Spring 5-17-2013

# A Comparative Look at Entity Framework Code First

Casey Griffin
*La Salle University*, griffinc4@student.lasalle.edu

# A Comparative Look at Entity Framework Code First

**Submitted to the**
**Department of Computer Information Science**

**By:**
**Casey Griffin**

**Date:**
**May 1, 2013**

## APPROVALS

| | |
|---|---|
| Agree to Advise: | Stephen A. Longo, Ph.D. (Project Advisor) |
| Date Submitted: | May 1st, 2013 |
| Approved by: | Stephen A. Longo, Ph.D. (MS in CS Committee) |
| Date Approved: | May 3rd, 2013 |

## Contents

## 1. Abstract

The motivation behind this project is to examine what "Entity Framework Code First" is bringing to the world of object relational mappers and data access and how it compares to the more traditional methods of the past. The problem is whether Entity Framework's high level of abstraction from the database schema is useful to the developers in reducing code development or if traditional approaches with their robust, custom data layers provide developers with an overall better performance. To analyze Entity Framework, a real-world business web application was developed implementing an Entity Framework Code First approach to data access. Using this implementation of a business application, comparisons were drawn between Entity Framework and more traditional data access techniques. The results of the research conclude that while it does have some criticisms, Entity Framework is an improvement upon traditional approaches. It greatly reduces the time spent writing code for the application's data access layer, makes managing database relationships and data objects easier, provides a level of abstraction to isolate the database from the developer's application, and translates queries at runtime allowing minimal code impact with regards to database storage changes.

## 2. Introduction

The course of computing technology is interesting in that the complex solutions of yesterday become the simplified, slick, and easier solutions for the user of tomorrow. The progress of technology is all too apparent in the world of web-based development where e-business is the driving force behind the rapid advancement of websites, web applications and web services. From static html pages to dynamic data driven websites, web development has come a long way. Credit to this can be given to user-friendly web development software such as Adobe Dreamweaver and Microsoft Visual Studios (VS), which allow their users access to diverse, powerful tools for web programming in an easy-to-learn environment (Adobe Dreamweaver CS6) (Visual Studio).

One of the newest technologies available in Visual Studios 2012 is the ADO.NET Entity Framework 5.0. Entity Framework (EF) allows ADO.NET users the ability to work with data-oriented applications at a higher level of abstraction. Developers no longer need to be confused by the complexities of entities, relationships, and business logic nor worry about creating the data engines used to retrieve and store data (Microsoft, 2013). Entity Framework saves developers from the headache of building data-oriented software applications by opting to provide more efficiently written and maintainable code for data accessing than traditional forms.

Entity Framework gives developers two approaches, Model First and Code First when constructing a suitable workflow for their applications projects. With both workflows, an application developer will first have to choose whether they will be working with creating a new database or an existing database. In Model First, the developer can create a new database by using the model designer and generate the database based on that model or they can reverse engineer an existing database in the model designer. With the Code First approach, the developer can create domain-name classes and mappings within their code and have the database generated accordingly or use reverse engineering tools to model an existing database and generate classes and mappings. This paper focuses on the latter approach, Code First and creates a new database.

The emphasis on using the Code First approach during this project was a combination of the usage of new technologies and concepts, and a personal preference towards coding rather than working with a model design. Code First in Entity Framework represents a relatively newer concept, in the sense that it wasn't introduced until .NET 4.0, where developers are able to code their object classes first without having to think about the database schema or how the objects would map to the relational database. The other reason as to why Code First was chosen was because when deciding on which development workflow to take, there was more familiarity with coding-centric workflow than the model approach.

Microsoft boasts that Entity Framework offers the opportunity to reduce the amount of code an application developer has to write when addressing impedance mismatch across various data representations as well as empowering the application developers to focus on the business goals of the applications rather than the complexities of bridging disparate data representations (Corporation, 2006). Yet an open mind is always needed to sustain an objective conclusion about a product which presents itself as "all too good to be true". If Entity Framework provides developers with such a powerful object relational mapper (ORM) tool which eliminates the code base programming between applications and their databases then why isn't everyone using it? What are the benefits and disadvantages of using Entity Framework over the traditional forms of data access? Are the Entity Framework features described by Microsoft as beneficial to developers as claimed?

There is obviously a great deal of analysis which needs to be considered before a conclusion can be drawn on whether Microsoft's Entity Framework achieves what it has intended to do. This paper will show the implementation of the Entity Framework using a Code First approach within a real world business application, its beneficial usages and features, and a comparison with traditional data access frameworks. Through the supportive implementation details of the paper and the comparison given, the ADO.NET Entity Framework Code First approach will prove to be the leading exemplary example of an ORM's ability to endow their application developers with a higher level of data access abstraction.

## 3. Background

Traditionally, when developers wanted to access information from a database, they would have to code for all the different types of data accessing needed. This would include writing out, sometimes lengthy, SQL statements that would be used to select, insert, modify, and delete data which was stored in the database. A traditional approach (Mitchell, 2006) to this was to create what was known as a Data Access Layer (DAL). The DAL would separate data access logic from the presentation layer which would include the web application or web page the user would see. The DAL would contain all classes that held the methods used for data access such as a “GetCategories()” method which returned all the categories in a Categories table. Typically, these DALs would be developed using manually derived SQL statements by the programmer and would require knowledge of the underlying database structure. This, in turn, meant that writing DALs would be a long, tedious and time consuming process that would require the developer to debug and tweak their code for efficient database performance.

These traditional approaches still exist but have since been built upon and advanced to where developers can use built-in application programming interfaces or APIs from vendors to achieve the data access they need. Such APIs includes the ones found in the ADO.NET Entity Framework as part of the Microsoft .NET Framework.

### 3.1 Microsoft .NET Framework and ADO.NET

Microsoft’s .NET Framework is a development platform primarily used in building Window’s applications. Microsoft began working on the .NET Framework in the late 1990s and is currently on its 4.5 generation (.NET Framework). The .NET Framework 4.5 comprises of “the common language runtime (CLR) and the .NET

Framework class library, which includes classes, interfaces, and value types that support an extensive range of technologies." (.NET Framework 4.5) The .NET Framework typically uses the Visual Basic and Visual C# programming languages and is integrated into Microsoft's development environment called Visual Studio. The current release, Visual Studio 2012 allows the development of console and graphical user interface applications such as websites, web applications and web services using the .NET Framework (What's New, 2013).

As part of the .NET Framework and Visual Studio integrated development environment (IDE), ADO.NET is a set of classes which provide programmers with means of accessing and modifying data stored in a data source. These data sources can range from relational databases found in database management systems (DBMS) such as SQL Server, MySQL, and SQL Express to XML documents (ADO.NET Overview). ADO.NET has two components, Data Providers and DataSets. The Data Providers allows interaction with the data source either by data manipulation or read only access. Data Providers build the access to complex databases such as Oracle, Microsoft SQL Server, MySQL, PostgreSQL, SQLite, DB2, and others (ADO.NET Data Provider). DateSet is used to access data that is independent from the data source. Thus it can be employed with differing data sources, XML data, or local application data (ADO.NET Architecture).

## 3.2 Entity Framework

One of the newest additions and enhancements built on to ADO.NET is the object relational mapper (ORM) Entity Framework, currently is in its 5.0 release. Figure 1 shows the current Entity Framework Architecture (Lokuge, 2012) and the interactions between architecture layers. By definition, an ORM is a “programming technique used to convert incompatible types into objects that can be used in programming languages.” (Object-relational mapping) The conversion of these incompatible database types into programming objects causes a problem area for developers known as impedance mismatch. Impedance mismatch refers to the differences between the relational database and object model in object relational mapping. Entity Framework reduces impedance mismatch by “mapping relational tables, columns, and foreign key constraints within the logical model to entities and relationships in the conceptual models” (Entity Framework Overview, 2012).

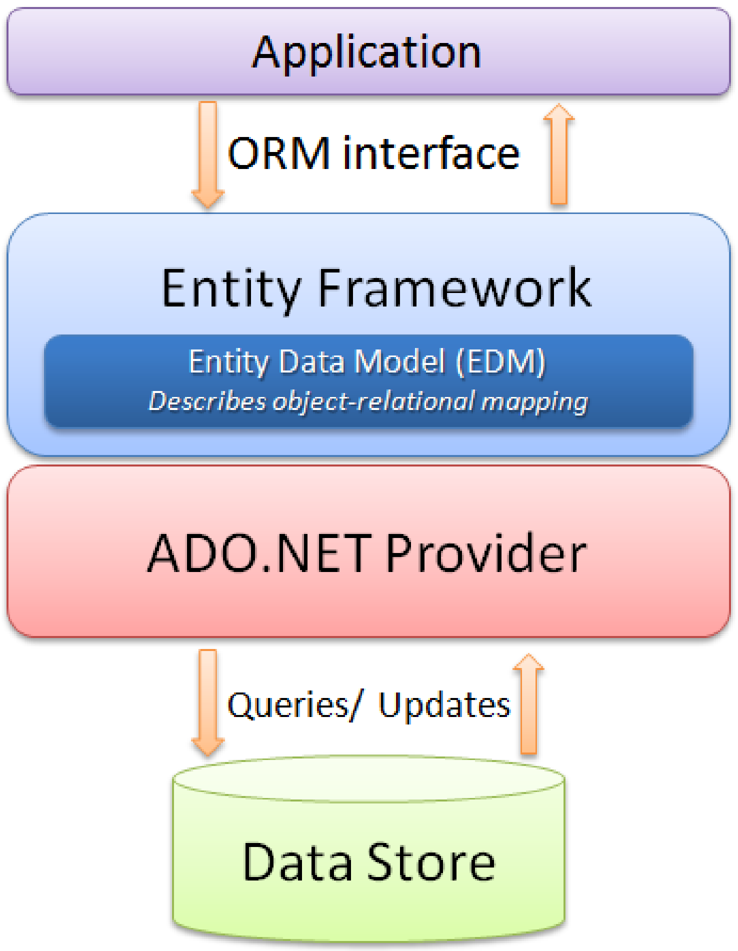


**Figure 1 Entity Framework Architecture**

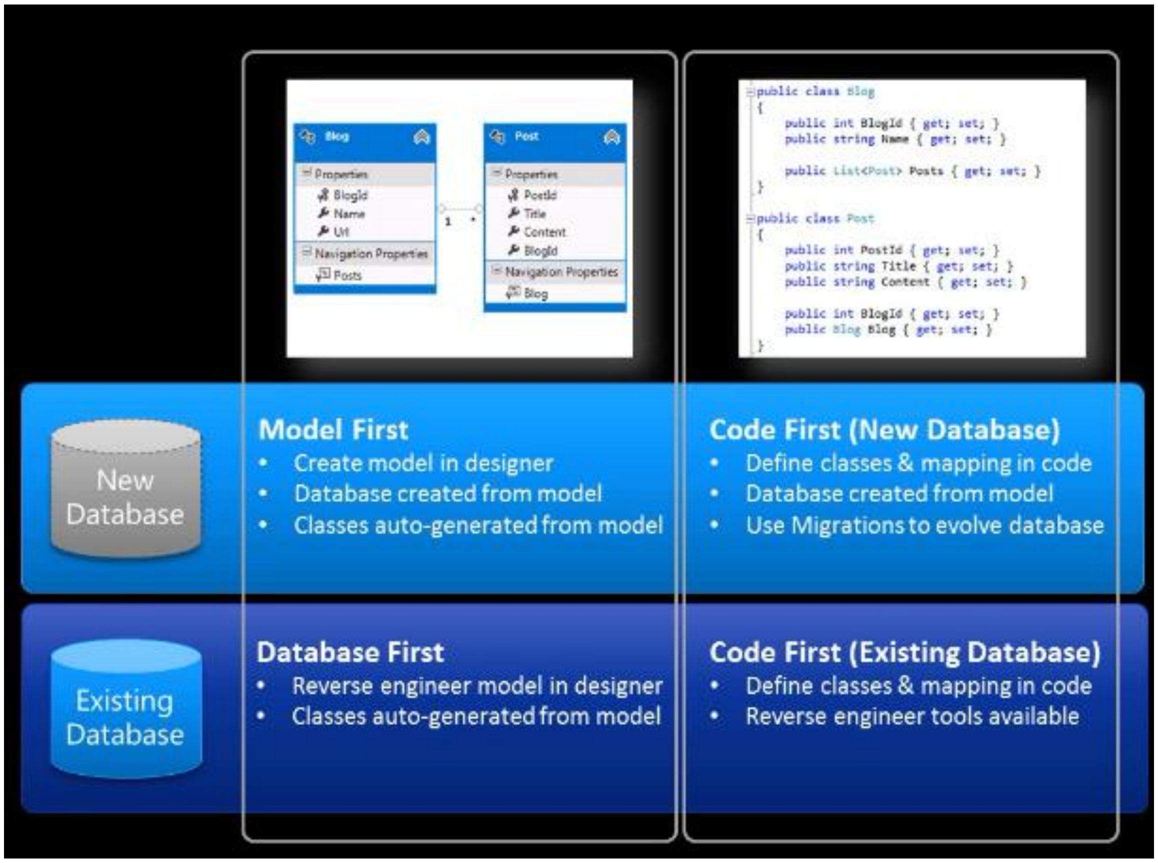
## 3.3 Conceptual Model

One of the primary goals of Entity Framework is to allow developers to query entities and relationships in the domain model or as it is known in Entity Framework, the conceptual model. This conceptual model will be mapped to the database schema which allows developers to work with relational data from the database using domain-specific objects (ADO.NET Entity Framework At-a-Glance). These domain-specific objects or entities are the converted incompatible types from the relational database which can be used in programming languages such as C# and Visual Basics. The conceptual model relies on Entity Framework to interpret queries on these entities and their relationships into data source-specific commands (Entity Framework Overview, 2012).

Building the conceptual model in Entity Framework depends on which development workflow is chosen.

## 3.4 Development Workflows

As previously mentioned, Entity Framework has essentially two workflows, Model First and Code First which can either be used with an existing database or the creation of a new database. Figure 2 shows the two Entity Framework development workflows, Model First and Code First (Miller, Entity Framework Development Workflows).



```
public class Blog
{
    public int BlogId { get; set; }
    public string Name { get; set; }

    public List<Post> Posts { get; set; }
}

public class Post
{
    public int PostId { get; set; }
    public string Title { get; set; }
    public string Content { get; set; }

    public int BlogId { get; set; }
    public Blog Blog { get; set; }
}
```

New Database

**Model First**
- Create model in designer
- Database created from model
- Classes auto-generated from model

**Code First (New Database)**
- Define classes & mapping in code
- Database created from model
- Use Migrations to evolve database

Existing Database

**Database First**
- Reverse engineer model in designer
- Classes auto-generated from model

**Code First (Existing Database)**
- Define classes & mapping in code
- Reverse engineer tools available

Figure 2 Entity Framework Development Workflows

In the Model First approach, a conceptual model can be created for a new database or generated from an existing database structure.  Generating the model based on an existing database is also known as the Database First approach.  In Model First, the data model represents the conceptual model or Entity Data Model (EDM). The mapping between the EDM and database structure or storage model is stored in the XML file. When working in Visual Studio, Entity Framework provides a visual designer in which developers can manipulate the EDM in a graphical representation. Once the EDM is completed, Entity Framework will auto generate the entity classes that developers can use in their programming.  All entity classes inherit the EntityObject class and as such, will always be tightly coupled to the Entity Framework.

In the Code First approach which was used in this paper, the conceptual model is inferred by classes and mapping defined by the developer's code and EF Code First conventions rather than auto generated from the EDM. At runtime, Entity Framework will generate the mapping metadata based on developer defined domain types and other configurations such as Code First Conventions, Data Annotations and the Fluent API. Like Model First, this approach can be used with either an existing database or the creation of a new database. When building a new database from scratch, Entity Framework employs Code First Migrations which are used to evolve the database as it is coded. Code First Migrations are explained in greater detail in the next section.

## 3.5 Code First Approach

Entity Framework's Code First development workflow with the creation of a new database was used in the project.

### 3.5.1 Plan Old CLR Objects (POCO) Classes

Starting out with Code First approach, developers generally will begin by defining domain name classes or Plain Old CLR Objects (POCO) classes. POCO classes are the C# equivalent to Plain Old Java Object classes and are basic object classes that follow no conventions or frameworks. POCO classes, unlike Model First's entity classes, do not inherit the EntityObject class and therefore are persistence-ignorant, which means they are independent of Entity Framework. These POCO classes are made up of scalar and navigational properties. Scalar properties are used to define the characteristics of the object such as ID and Name while navigational properties are used to define the relationships between classes. By using the keyword, "virtual" when defining a navigational property, programmers can take advantage of the Entity Framework feature, Lazy Loading. Lazy Loading allows for the contents of related entity defined in the code to be populated only when the relational property is queried. The opposite of Lazy Loading is Eager Loading which will load the related entities in query regardless of whether it is needed or not.

### 3.5.2 DbContext API

With the POCO classes defined, Entity Framework can be used to provide data access functionality through the DbContext API. This DbContext is derived from the System.Data.Entity.DbContext class within the Entity Framework. The DbContext represents a session with the database and allows a developer to query and save data (Miller, Entity Framework Code First to a New Database). In the DbContext, a DbSet property is defined for each POCO class, which permits developers to query and save instances of those classes.

Once the domain classes and the DbContext class have been written, a developer can then use the DbContext within their application. Entity Framework will automatically generate the database using Code First Conventions, Data Annotations, and the Fluent API to decipher what the conceptual model should be based on the domain classes and the DbContext.

### 3.5.3 Code First Conventions

Defining domain model in Code First is like describing to Entity Framework how to build the conceptual model. Entity Framework uses what is known as Code First Conventions to discover how the conceptual model should be built and mapped and in the case of a new database, how to create the database storage model. The Code First Conventions include Type Discovery, Primary Key, Relationship and Complex Types conventions. The concept of an entity type can be thought as the domain name classes and their properties. The type discovery convention determines the types that will be used in the conceptual model, which is based upon the types defined within the DbContext described in 3.5.2. The Primary Key convention simply states that a property with the name "ID" or the name of the class plus "ID" will act as the primary key for that class. The Relationship convention infers that navigational properties as well as foreign key properties will denote existing relationships between classes. The Complex Types convention indicates that where no primary key is defined for a class that that class should become a complex type. A complex type is basically an entity type without any identifier and therefore must be a property of other entity types.

### 3.5.4 Data Annotations and the Fluent API

In Code First, there exists two ways of configuring/overriding the Code First Conventions, Data Annotations and the Fluent API. Entity Framework uses the Data Annotations and the Fluent API to configure how Entity Framework builds the conceptual model. The Data Annotations are basically supplementing Entity Framework with properties about the model it will build. An example would be including a MaxLength 50 data annotation, Entity Framework will see this and work out that this property should have a max length of 50 when it builds its counterpart in the physical database. The Fluent API is Entity Framework's Fluent API. It is the programming interface developers can use to manipulate how Entity Framework builds the model.

When constructing domain name classes, a developer can define the Key (Primary Key) and Association (Foreign Key) values for the properties of classes by using Data

Annotations. Other data annotations, which can be used, include indicating a required property and/or max length restrictions on the property. These Data Annotations help customize how Entity Framework will generate the model by overwriting the Code First conventions. Examples of Data Annotations are shown later as they apply to the overall project.

Like Data Annotations, the Entity Framework's Fluent API is used to configure Code First conventions. Essentially, EF's Fluent API can perform any type of manipulation a developer would need when creating the model such as defining primary keys, properties, types, relationships, and table mappings. To use the Fluent API requires developers to override an OnModelCreating method inside the DbContext described in 3.5.2. Inside this method, programmers can configure how their domain name classes will translate into the conceptual model, map to the database and possibly how the database will be created.

### 3.5.5 Code First Migrations

As developers want to evolve their databases over time, they will need to change the conceptual schema which in turn will change their database structure. Code First Migrations are used to accomplish this goal. When enabled, Migrations will create a separate folder called Migrations in the Visual Studio project. This Migrations folder will hold a configuration file, an initial migration database if the database has already been created, and any sequential migrations that are created. The configuration file holds the settings for the DbContext and can be used to set up initial seed data for the database. With Migrations enabled, developers can make changes to their domain model of POCO classes and then use the command, Add-Migration. Adding a Migration will generate the code needed to make the database changes and place it in the Migrations folder as the user defined the Migration. The command, Update-Database, is then used to commit the Migration changes to the database.

## 4. Methodologies

In order to test and validate the claims Microsoft makes on their Entity Framework as well as comparing it against traditional forms of data access, an application would need to be developed which implements Entity Framework. The application chosen for this paper was a business application for the drywall company, Griffin Drywall & Insulation Inc. This business application took the form of an online e-portal for the Griffin Drywall which utilized Entity Framework to access their backend SQL server database. The idea of using a business application of this type to test Entity Framework's capabilities and features was to get a sense of how an Entity Framework Code First approach functions in a real world application.

### 4.1 Example: Griffin Drywall

The business application developed using Entity Framework Code First approach in this paper was an e-portal website designed for Griffin Drywall. Griffin Drywall & Insulation Inc. is a local family owned drywall and insulation company centrally located in Plumsteadville, Pennsylvania. Founded in 1980 by Carlson and Tim Griffin (father & son), Griffin Drywall & Insulation Inc. is now ran by Tim's sons, Clayton and Chris Griffin. The company is contracted to do commercial, industrial, residential, and municipal buildings throughout the Pennsylvania and New Jersey area. Services provided are metal framing, acoustic ceiling, drywall, insulation, and demolition.

Several other technologies were used in addition to Entity Framework 5.0, which Table 1 outlines with a description their part in the overall project design.

**Table 1 List of Technologies**

| Technology | Description | Version |
|---|---|---|
| **Visual Studio** | The IDE from Microsoft. This is the development tool used to code the project. | 2012 |
| **SQL Server 2012** | A relational database management system which was used to house and manage the development database. | 2012 |
| **Adobe Photoshop** | A graphics editing program. This was used to design the website's presentation layer. | CS6 |
| **Adobe Dreamweaver** | A web development application. This application was primarily used to convert the Photoshop website design into a Master webpage which would be used in Visual Studio. | CS6 |
| **Adobe Flash** | A multimedia and software platform, which used to create moving images on the website. | CS6 |
| **JQuery** | jQuery is a new kind of JavaScript Library. It was used to add some specific functionality within the project. | 1.9.1 |
| **Galleria.js** | Galleria is a JavaScript image gallery framework. This framework was used to design the project image galleries. | 1.2.9 |
| **Fire Bug** | A web development tool that was used to debug the cascade style sheet and html elements used on the page. | 1.11.2 |

## 4.2 Business Requirements

As defined in the proposal, the business application for Griffin Drywall was developed based on the functional requirements described in Table 2. These requirements satisfied the goal of developing a real-world application using Entity Framework Code First approach with regards to data access. Some functional abilities were added during the development of the application to take advantage of some of Code First's features as well as to showcase some of the comparisons that could be drawn between the traditional data access approach and the Code First's.

**Table 2 Functional Requirements**

| Functional Requirement | Test Plan: Test Cases |
|---|---|
| **1. Website has navigational panel to move between web pages.** | 1.1 User is able to traverse the website through the usage of the navigational panel. (Normal Use)<br><br>1.2 User encounters an error while moving between pages. (Abnormal Use)<br><br>1.3 User clicks on page link and is not moved to correct page. (Abnormal Use) |
| **2. Home Page: Provides general information about the company.** | 2.1 User is directed to the Home page and it is displayed correctly. (Normal Use)<br><br>2.2 User is directed to the Home page and it is not displayed as intended. (Abnormal Use)<br><br>2.3 User encounters an error on the Home page. (Abnormal Use) |
| **3. Services Page: Provides information on the services provided by the company.** | 3.1 User is directed to the Services page and it is displayed correctly. (Normal Use)<br><br>3.2 User is directed to the Services page and it is not displayed as intended. (Abnormal Use)<br><br>3.3 User encounters an error on the Services page. (Abnormal Use) |
| **4. Contact Us Page: Customers are able to submit information which will be forwarded to company email address.** | 4.1 User is directed to the Contact Us page and it is displayed correctly. (Normal Use)<br><br>4.2 User is directed to the Contact Us page and it is not displayed as intended. (Abnormal Use)<br><br>4.3 User encounters an error on the Contact Us page. (Abnormal Use) |
| **5. Professional design with company logo and official company colors.** | 5.1 Website uses the company logo and is designed around the company colors.<br><br>5.2 Website does not display either company logo and company colors correctly or as specified by the client. |
| **6. User Login and Password Management** | 6.1 User enters the correct login information and is able to login into the website. (Normal Use)<br><br>6.2 User enters incorrect login information. The user is alerted that the information is incorrect and access to any personalized account is denied. (Abnormal Use)<br><br>6.3 User forgets password and uses link to recovery password. Password is sent to assigned email address. (Abnormal Use) |
| **7. Scheduling Page: Page used to schedule jobs of current employees.** | 7.1 User is directed to the Scheduling page and it is displayed correctly. (Normal Use)<br><br>7.2 User is directed to the Scheduling page and it is not displayed as intended. (Abnormal Use)<br><br>7.3 User encounters an error on the Scheduling page. (Abnormal Use) |
| **8. Scheduling Page only accessible to authorized employees.** | 8.1 User enters correct login information to enter Scheduling page. (Normal Use)<br><br>8.2 User does not enter correct login information and is denied access. (Abnormal Use) |

| Functional Requirement | Test Plan: Test Cases |
|---|---|
| **9. Backend database with company information.** | 9.1 Transparent connection to backend database. (Normal Use)<br><br>9.2 Error(s) while connecting/working with backend database. (Abnormal Use) |
| **10. Scheduling Page should pull current data from database to populate the page.** | 10.1 Correct data is pulled from database and used on website. (Normal Use)<br><br>10.2 Incorrect data is used on the website from the database. (Abnormal Use) |
| **11. Maintain database through website interface.** | 11.1 Client is able to manage database through online interface. (Normal Use)<br><br>11.2 Client is unable to connect to database. (Abnormal case)<br>11.3 Client is unable to perform the maintenance required. (Abnormal Use)<br><br>11.4 Client encounters an error while working with the database. (Abnormal Use) |

## 4.3 EF Code First

As described in earlier sections, Development Workflows and Code First Approach, this project was designed to use Entity Framework Code First with development workflow surrounding the creation of a new database. The project used the general steps in the approach by creating domain name classes (POCO classes) for each business entity, constructing a DbContext for which a conceptual model could be built, and using Code First Migrations to develop and evolve the database structure.

### 4.3.1 POCO Classes

There were ten POCO classes involved in the Griffin Drywall application. They were Company, GImage, Job, Project, Request, Service and ServiceType, Subcontractor, Work, and WorkDetail. Table 3 has a description of each class as it relates to the business application while Appendix A holds the source code for the classes used.

Table 3 POCO Classes

| Class | Description |
|---|---|
| **Company** | This class serves to provide information about a company, which would include Griffin Drywall as well as the companies it is associated with. |
| **Job** | This class has the properties associated with a Griffin Drywall job. |
| **GImage** | This class holds the addresses for Gallery Images that are mapped to a specific Project. |
| **Project** | This class stores the information all Griffin Drywall Projects. |
| **Request** | This class has information on client requests (bids) for services provided by Griffin Drywall. |
| **Service** | This class holds the properties of the services which Griffin Drywall offers. |
| **ServiceType** | This class holds the information on types of services Griffin Drywall has. |
| **Subcontractor** | This class has the information pertaining to individual subcontractors that work for Griffin Drywall. |
| **Work** | This class acts as the many-to-many connection between Job and Subcontractor. |
| **WorkDetails** | This class holds the details for any work done on a job by a subcontractor. |

Each class was associated with a business entity for Griffin Drywall. As well, each class held their own unique properties for the entity they define and their relationship with other classes. Each class used Data Annotations in order to set certain predefined restraints and other guidelines, which would help/overwrite the Code First conventions when generating the database. The two figures, Figure 3 and Figure 4 show the graphical model of each POCO class as it would be seen from a Model First approach.

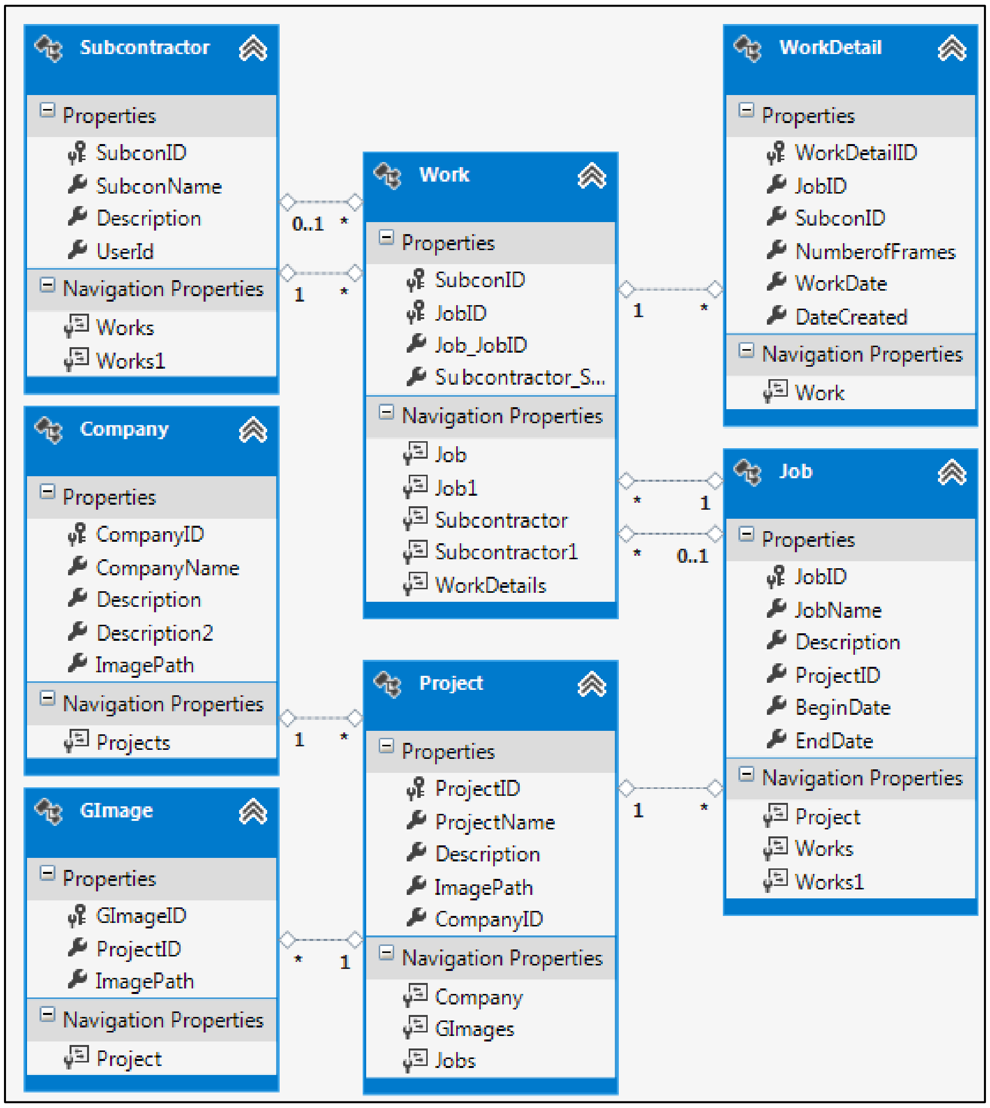


**Figure 3 Griffin Drywall Model Part 1**

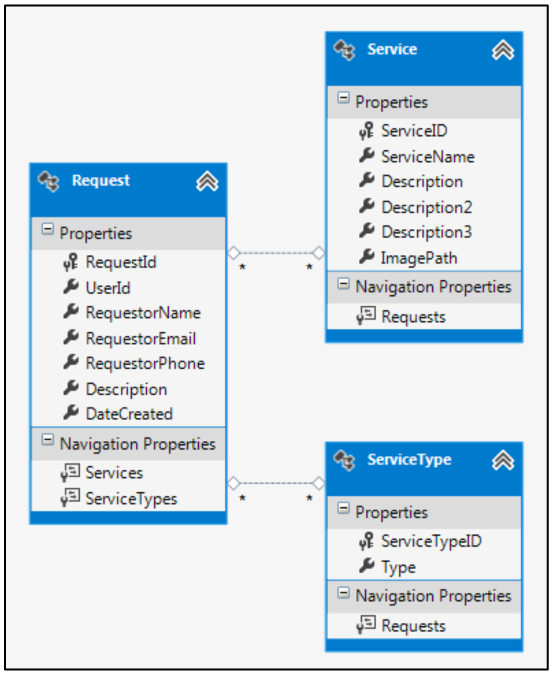


Figure 4 Griffin Drywall Model Part 2

### 4.3.2 DbContext API

The DbContext for the project was designed to use the domain name classes described in POCO Classes. Each class was given a DbSet property to allow querying and database save changes for instances of those classes as they appeared in the code. The Code First's Fluent API was applied to ensure that the mapping between the many-to-many relationships of Request/Services and Request/ServiceType were configured correctly. The source code for the Griffin Drywall DbContext can be found in Appendix B.

### 4.3.3 Code First Migrations

Code First Migrations was used in the design for the purpose of updating the database as the domain classes were manipulated and tailored to the business needs. There were many Migrations used in the development of the application, with some more meaningful than others, such as the Migration, Add-Request shown in its entirety in

Appendix C. Note that while the majority of these migrations used the Code First conventions to deduce the conceptual model and thus the database structure, the classes also used Data Annotations and the Fluent API to overwrite these some conventions. Such is the case in Add-Request where Code First's Fluent API was used to overwrite how the model was built in regards to how the entities are mapped together. For example, the bridging entity, "RequestServices" which was created in the Add-Request Migration by Code First convention may have been given another name but was overwritten in the DbContext by the following line of code: `m.ToTable("RequestServices");`

## 4.4 The Website

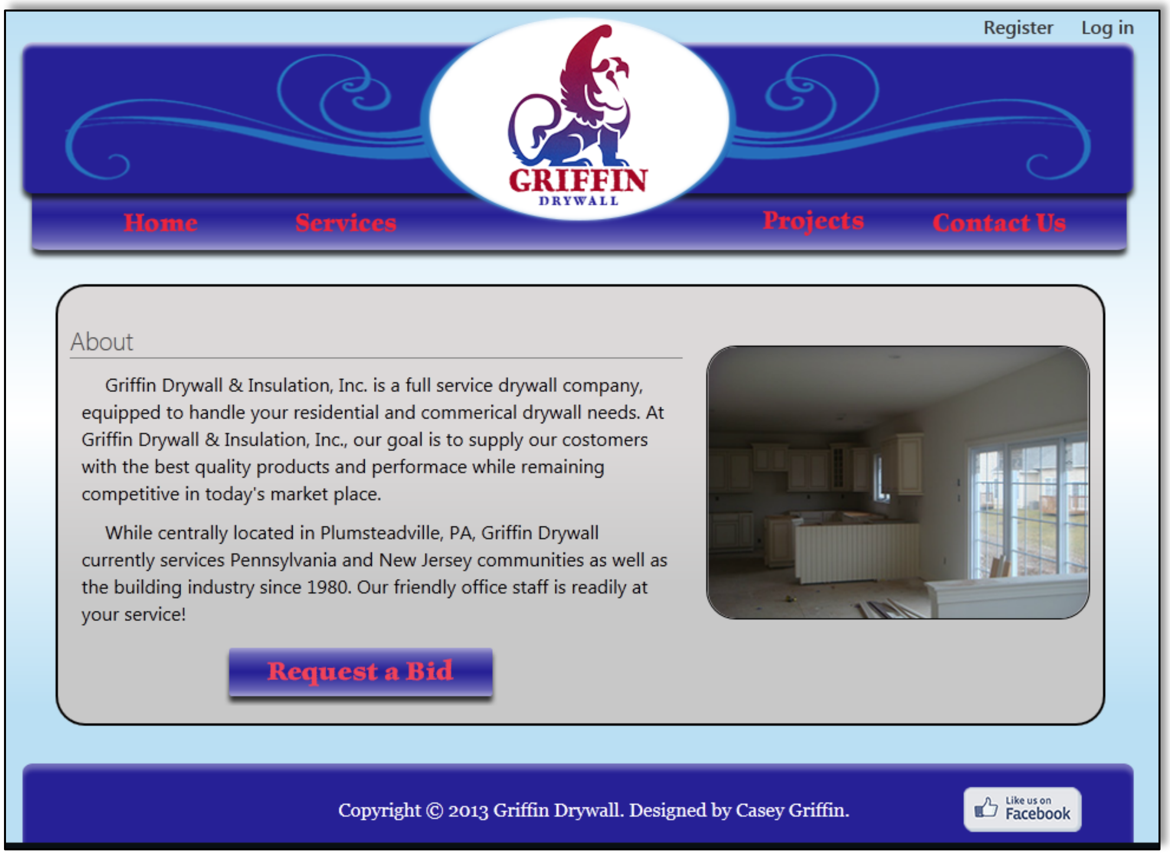
The business application, designed to implement Entity Framework Code First and validate its superiority over the traditional approaches of data access, was the Griffin Drywall and Insulation Inc. website. This data-driven website was constructed to use the Code First approach to successfully access and modify the backend database which would correlate to data and elements that appeared on the webpages of the site. The site is made up of a presentation layer of the home, request bid, services, projects, and contact us pages. There are also account management and administrator services which include pages to manage registered user accounts and the interface to interact with the backend database. The source code for the web pages can be found in Appendix D.

### 4.4.1 Master Page

The Master page contained the overall concept design for the Griffin Drywall website. It was designed to meet the business requirements #1, Navigational Panel and #5, Professional Design defined in Table 2. Each page of the site inherited this design and used content containers for their own individual subject matter. The Master page was made up of the Company logo, the navigation bar, account links, and footer which includes a Facebook link. In addition depending on the user's credentials, the administration tab became visible and the user could access the administration services.

### 4.4.2 Home Page

The Home page (Figure 5) was designated as the default page of the site and would be the first page users see when interacting with the site. It was intended to meet the business requirement #2, Home Page defined in Table 2. In terms of Entity Framework, this page was not developed to interact with the database. The contents of the Home page were the About article for the Griffin Drywall company, a Flash slide show gallery, and the Request Bid link.



About

Griffin Drywall & Insulation, Inc. is a full service drywall company, equipped to handle your residential and commerical drywall needs. At Griffin Drywall & Insulation, Inc., our goal is to supply our costomers with the best quality products and performace while remaining competitive in today's market place.

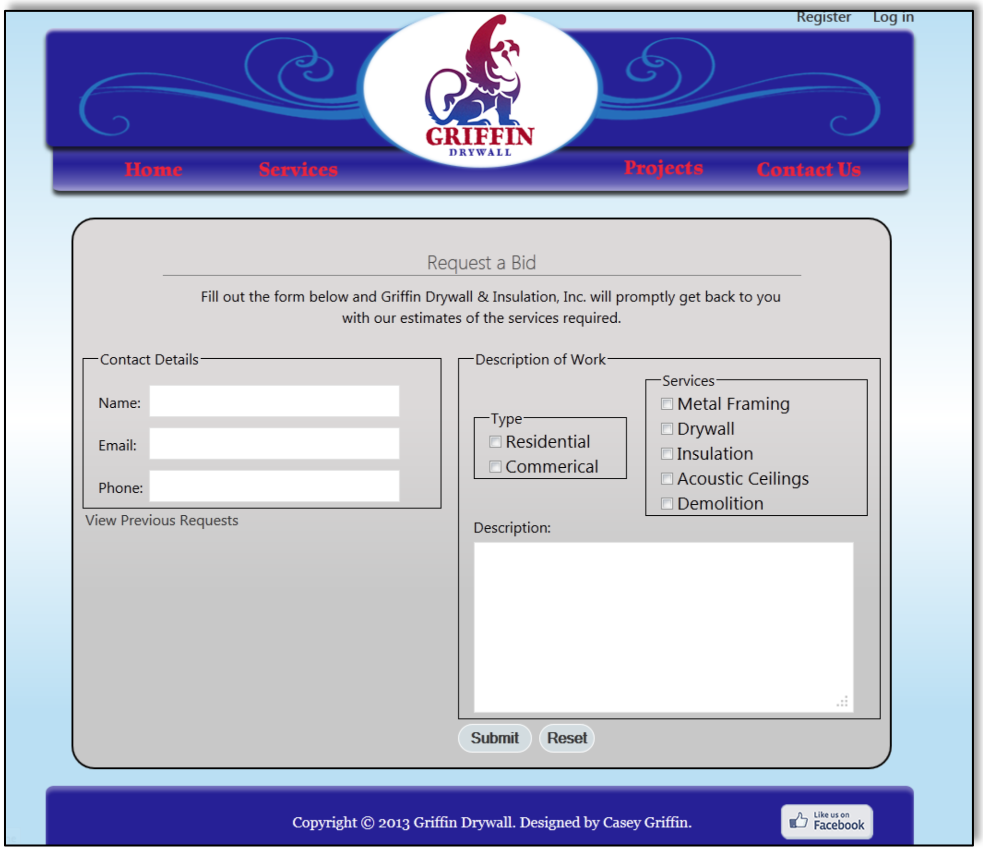
While centrally located in Plumsteadville, PA, Griffin Drywall currently services Pennsylvania and New Jersey communities as well as the building industry since 1980. Our friendly office staff is readily at your service!

Copyright © 2013 Griffin Drywall. Designed by Casey Griffin.

Figure 5 Home

### 4.4.3 Request Bid Pages

There were two pages associated with the requesting a bid, RequestBid (Figure 6) and PrevReq (Figure 7). Both pages used the Entity Framework to some degree to access the database. On the RequestBid page, EF allowed the checkbox content for the Service Type and Services to be dynamically generated. The issued submit command with the required information inserted a new entry into the Request table. This request was directly associated with a login user or a generated globally unique identifier (GUID) for anonymous users. Users who have already made previous requests can view those requests in the PrevReq page. This page was designed to use EF to populate a Gridview with the user's previous requests.



Request a Bid

Fill out the form below and Griffin Drywall & Insulation, Inc. will promptly get back to you with our estimates of the services required.

Contact Details

Name:

Email:

Phone:

View Previous Requests

Description of Work

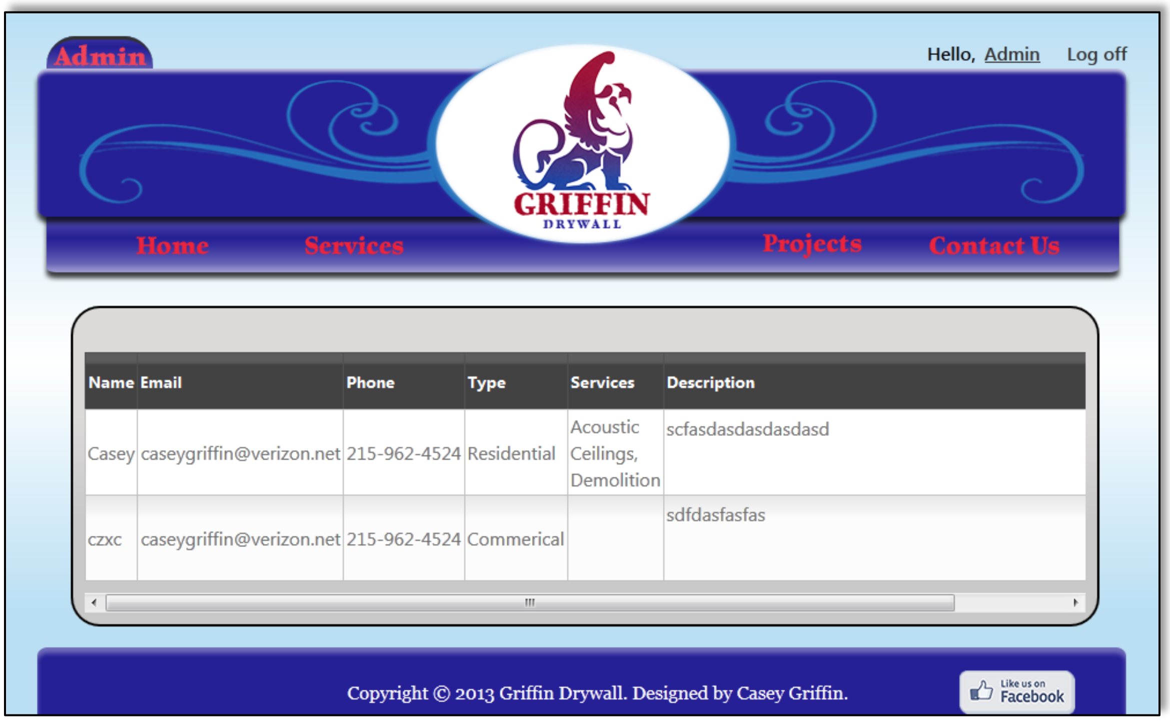
Type
- ☐ Residential
- ☐ Commerical

Services
- ☐ Metal Framing
- ☐ Drywall
- ☐ Insulation
- ☐ Acoustic Ceilings
- ☐ Demolition

Description:

Copyright © 2013 Griffin Drywall. Designed by Casey Griffin.

**Figure 6 RequestBid**



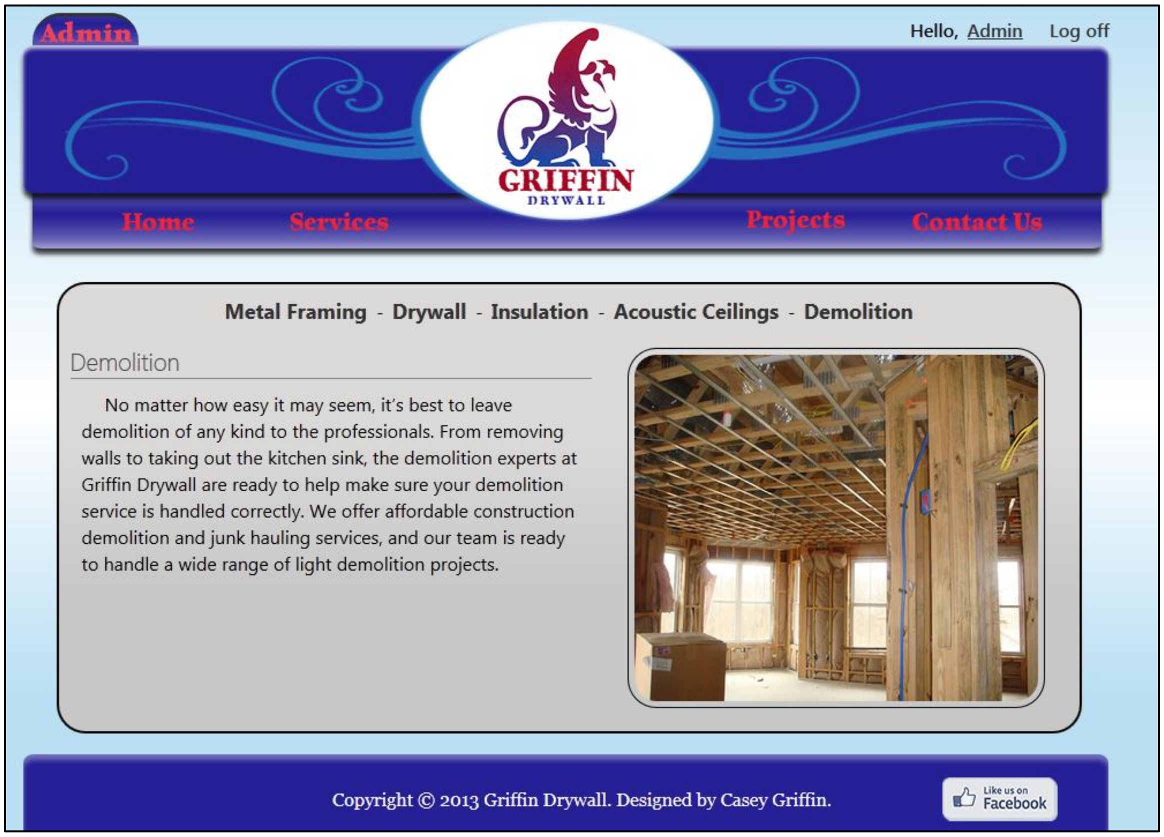
| Name | Email | Phone | Type | Services | Description |
|---|---|---|---|---|---|
| Casey | caseygriffin@verizon.net | 215-962-4524 | Residential | Acoustic Ceilings, Demolition | scfasdasdasdasdasd |
| czxc | caseygriffin@verizon.net | 215-962-4524 | Commerical | | sdfdasfasfas |

Copyright © 2013 Griffin Drywall. Designed by Casey Griffin.

Figure 7 PrevReq

### 4.4.4 Services Page

Entity Framework was used in the Services page (Figure 8) to access the database and pull the information on the different services offered by Griffin Drywall. The Service page accomplished the business requirement #3, Service Page defined in Table 2. The pulled database information generated the navigation bar of services for the user by displaying the name of the service in an unordered navigational link menu. When a link was clicked, a post back to the server occurred and the rest of the page was generated to display the information about the service and the image associated with it. This created a data-driven page which used Entity Framework to fetch the information about the services. These services can be managed by the website administrator through the interface in the Admin Services.



Demolition

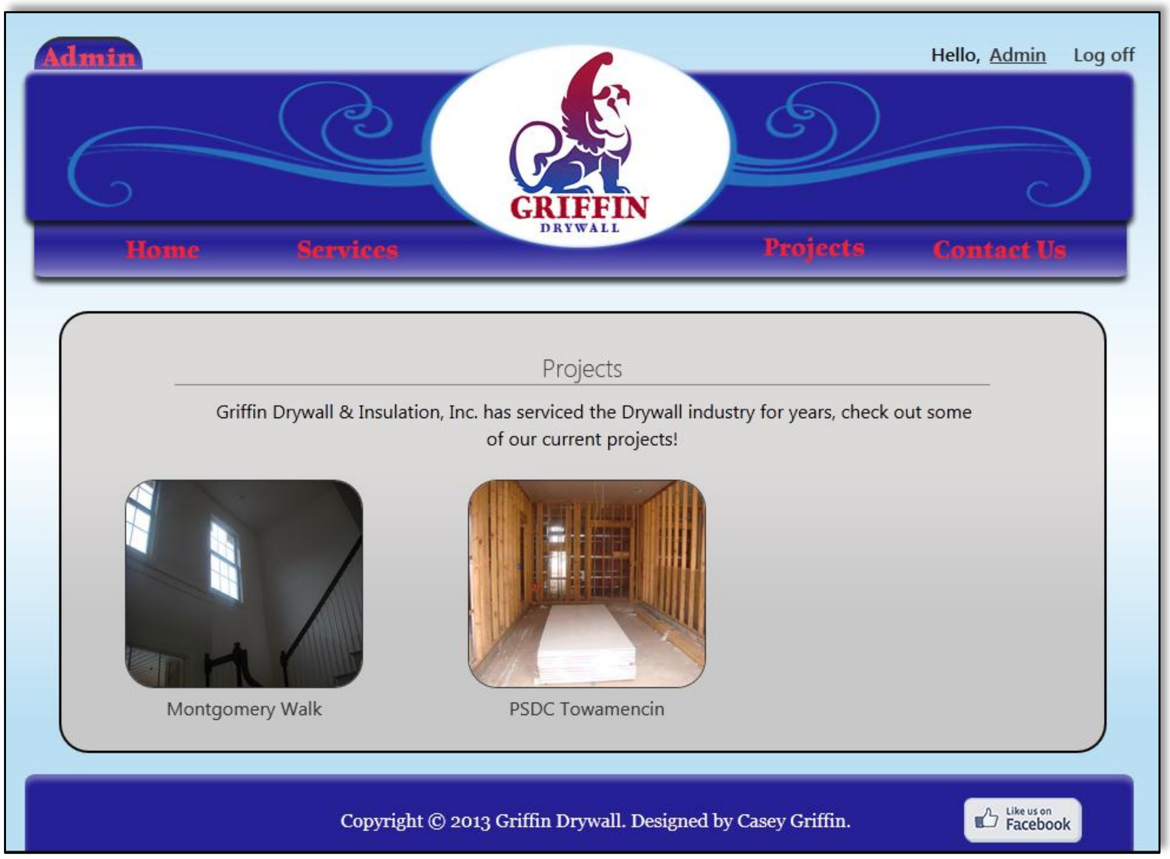
No matter how easy it may seem, it's best to leave demolition of any kind to the professionals. From removing walls to taking out the kitchen sink, the demolition experts at Griffin Drywall are ready to help make sure your demolition service is handled correctly. We offer affordable construction demolition and junk hauling services, and our team is ready to handle a wide range of light demolition projects.

Copyright © 2013 Griffin Drywall. Designed by Casey Griffin.

Figure 8 Services

### 4.4.5 Project Pages

Similar to the Services page, there were two Project pages used in this application, Project Overview (Figure 9) and Project Details (Figure 10). Both pages used Entity Framework to some degree to access the database and was an added business requirement. The Project Overview page extracted the name and thumbnail image for projects listed in the database and displayed them in a fashion so that users could click them and be linked to the Project Detail page for that project. The Project Details page took the Project Id sent in the URL to display the properties about the Project. The Galleria javascript image gallery on this page used Entity Framework to obtain the image paths for each of the project images within the relationship between the entities, Project and GImage. The gallery was then dynamically populated with the image paths so that the correct images were shown for the project.



Projects

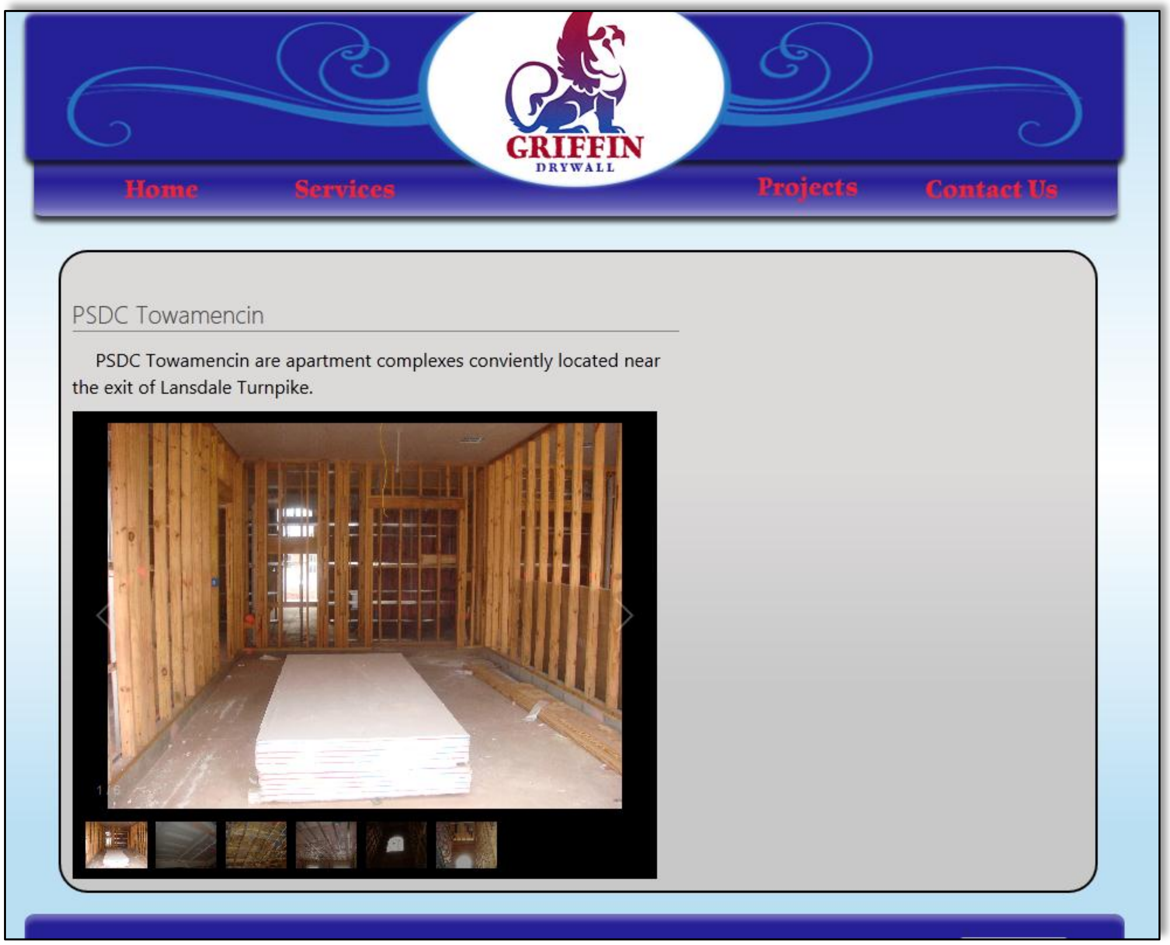
Griffin Drywall & Insulation, Inc. has serviced the Drywall industry for years, check out some of our current projects!

Montgomery Walk

PSDC Towamencin

Copyright © 2013 Griffin Drywall. Designed by Casey Griffin.

Figure 9 Project Overview



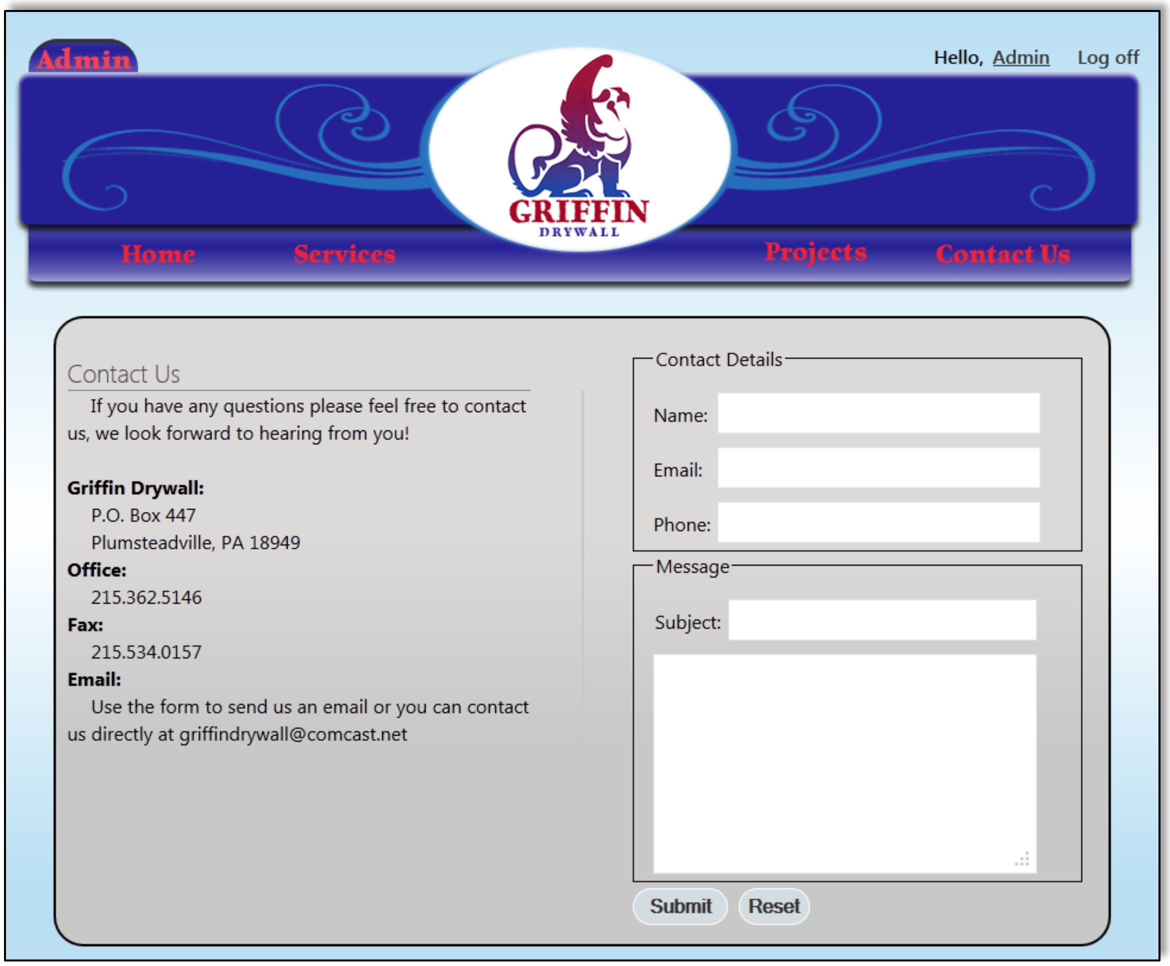
PSDC Towamencin

PSDC Towamencin are apartment complexes conviently located near the exit of Lansdale Turnpike.

Figure 10 Project Details

### 4.4.6 Contact Us Page

The Contact Us page (Figure 11) served as a means for prospective clientele of Griffin Drywall and Insulation Inc. to reach out to them. This page filled the business requirement #4, Contact Us Page in Table 2 but has no use of the Entity Framework. The Contact Us page code used SmtpClient to send an email to Griffin Drywall with the client’s entries on the form.



Contact Us

If you have any questions please feel free to contact us, we look forward to hearing from you!

**Griffin Drywall:**
P.O. Box 447
Plumsteadville, PA 18949
**Office:**
215.362.5146
**Fax:**
215.534.0157
**Email:**
Use the form to send us an email or you can contact us directly at griffindrywall@comcast.net

Figure 11 Contact Us

### 4.4.7 Account Management

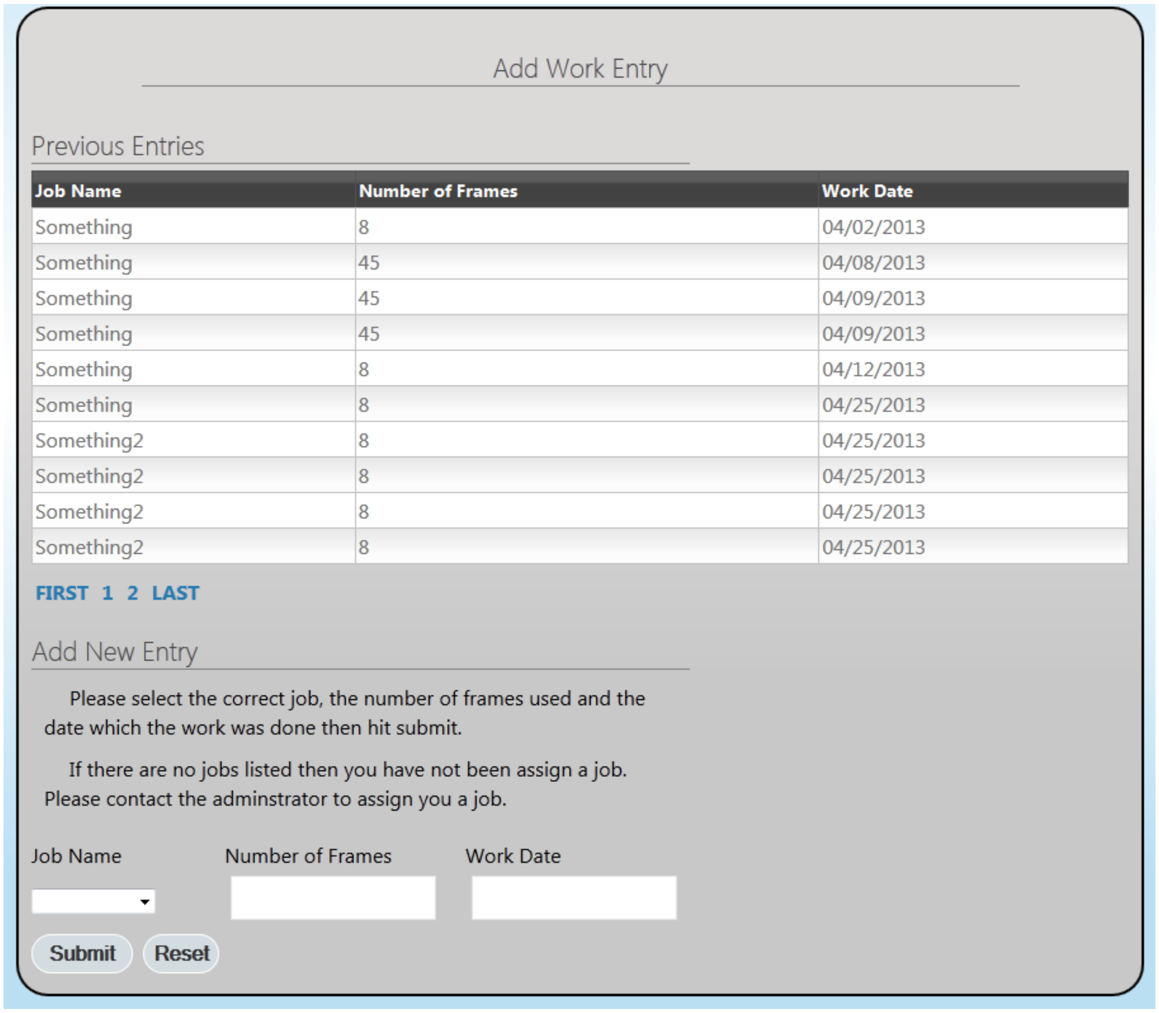
The account management for the Griffin Drywall site utilized .NET Framework's ASP.NET Membership to handle their user account, roles, and memberships. Standard with any default Visual Studio web application template, ASP.NET Membership is a built-in way to validate and store user credentials (Introduction to Membership). The layouts for these pages which included Login, Manage, Register, and Register External Login were all modified to fit the overall site design. ASP.NET Membership was used to meet the functional requirement #6, Account Management in Table 2.

### 4.4.8 Admin Services

Requiring admin authorization, the admin services provided the Griffin Drywall and Insulation Inc. with the ability to add, modify and delete data within their database. It served to meet the function requirement #11, Database Maintenance as outlined in Table 2. Interacting with the database would have a direct impact on the content presented to the users on the website. The Entity Framework Code First approach was used in all pages within this area to access instances of domain classes and save changes. Besides the created POCO classes, EF was custom fitted around the usage of ASP.NET Membership, which creates a default database structure used in managing and assigning users' roles and memberships. This database was integrated into the Griffin Drywall database to add and edit current user accounts and assigning roles to those users. The admin services' pages also fulfill the functional requirements #7, #8, and #10 which deal with the scheduling of jobs for subcontractors.

### 4.4.9 Subcontractor Page

The Subcontractor page (Figure 12) was designed to only be accessible to those users assigned with a Subcontractor role. It, essentially, allowed subcontractors to login and update work details about the jobs they had been assigned to.



Add Work Entry

Previous Entries

| Job Name | Number of Frames | Work Date |
|---|---|---|
| Something | 8 | 04/02/2013 |
| Something | 45 | 04/08/2013 |
| Something | 45 | 04/09/2013 |
| Something | 45 | 04/09/2013 |
| Something | 8 | 04/12/2013 |
| Something | 8 | 04/25/2013 |
| Something2 | 8 | 04/25/2013 |
| Something2 | 8 | 04/25/2013 |
| Something2 | 8 | 04/25/2013 |
| Something2 | 8 | 04/25/2013 |

FIRST 1 2 LAST

Add New Entry

Please select the correct job, the number of frames used and the date which the work was done then hit submit.

If there are no jobs listed then you have not been assign a job. Please contact the adminstrator to assign you a job.

Figure 12 Subcontractor

## 5. Analysis

The analysis of Entity Framework Code First approach used within a business application will show that the ORM's abilities and features provide a superior development environment with regards to data access than more traditional methods. These abilities and features will be examined and related to the implementation process of the project to confirm these advantages. A direct comparison of Entity Framework with the traditional data access approach will demonstration how much Entity Framework is bringing in terms of capabilities for application developers.

Some of the key developmental advantages that Entity Framework offers are:

- Reducing development time by using the conceptual model to generate code and EF data accessing capabilities.
- The conceptual model allows developed applications to be independent of their data engine or physical model, which means that they can use different RDBMs without modifying the code or being tied down with SQL dependencies.
- Supports the usage of LINQ that allows for compile-time syntax validation when writing queries against the conceptual model.
- With respect to Code First, developers can create persistence ignorant domain name classes, which can be reused with other frameworks if needed.

### 5.1 Entity Framework Features

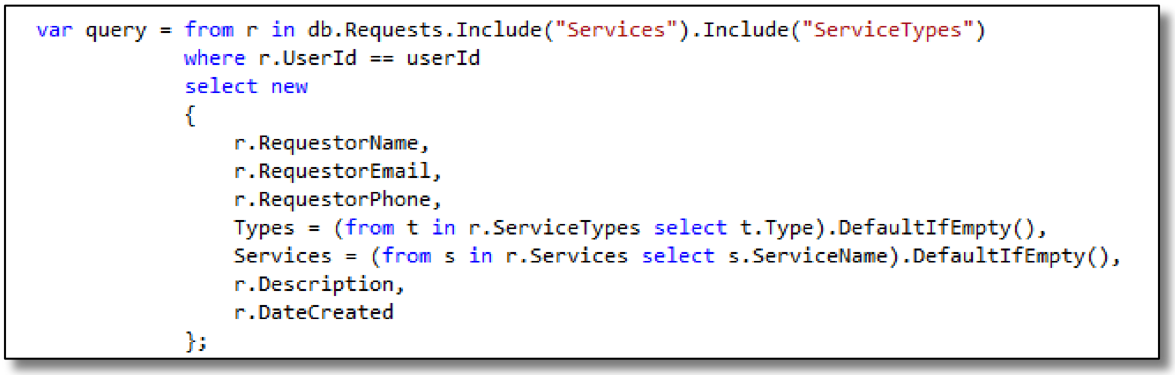
There a several features that Entity Framework has introduced which make it comparatively better than traditional data access with self-written code. One of the most prominent features is its ability as an ORM to make a conceptual model that overlays the database schema, adding a layer of abstraction for developers. This conceptual model allows developers to work with the entities as they appear from a functional or business standpoint rather than how they are structured within the database. Also the conceptual model allows developers to generate code needed to interact with the database whether using a Model First or Code First approach.

Entity Framework encapsulated the Lazy Loading feature into their 4.0 release. This feature allows entities with relationships to other entities to have those entities loaded on demand when the former entity's navigational property is accessed. For an example, an entity, OrderDetails using Lazy Loading with a relationship to Order will have its data populated only during the first time OrderDetails is accessed through an Order query. Developers using this feature can then designate that the automatic loading of an entity only take place when the entity is needed. On the other hand, using Eager Loading in the example above would load OrderDetails whenever the Order entity was queried. This improves the performance of relational queries because data will only be initialized when needed.

### 5.1.1 LINQ

Another feature which EF has over other older approaches is that it utilizes the querying language, LINQ. In EF, developers can write their queries against the conceptual model created rather than the physical database schema. This offers developers with a great deal of scalability as they are able to essentially move from one database provider (SQL Server, Oracle, etc…) to another without any modification to their existing code and queries. The EF framework is built on LINQ-to-Entities, a LINQ provider that will build the SQL query expressed in the LINQ. This is a great benefit because LINQ to SQL passes its data via SQL parameters, which helps prevent most common forms of SQL Injection.

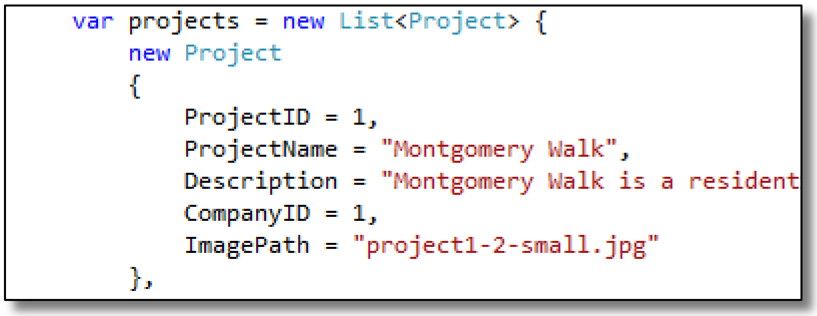
To support LINQ, a number of new language constructs needed to be added to the .NET languages, C# and VB. These enhancements include query expressions, implicitly typed variables, anonymous types, object/collection initializers, auto-implemented properties, lambda expressions, and extension methods (C# 3.0 Features That Support LINQ, 2008).



```
var query = from r in db.Requests.Include("Services").Include("ServiceTypes")
            where r.UserId == userId
            select new
            {
                r.RequestorName,
                r.RequestorEmail,
                r.RequestorPhone,
                Types = (from t in r.ServiceTypes select t.Type).DefaultIfEmpty(),
                Services = (from s in r.Services select s.ServiceName).DefaultIfEmpty(),
                r.Description,
                r.DateCreated
            };
```

Figure 13 Request Query

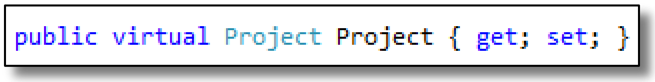
In the Figure 13 example above, the query expression is formatted in a similar SQL query (from, where, select). At compile time, the query syntax will be translated by the LINQ provider into a SQL statement to the database. The implicitly typed variable enhancement can be seen in the above example with the usage of "var query." Using the var modifier, the compiler will infer the type of the variable. This makes creating anonymous types possible like "select new {…}." These anonymous types allow properties such as ReqestorName, RequestorEmail, and etc… to be encapsulated into a single object without being explicitly defined.



```
    var projects = new List<Project> {
        new Project
        {
            ProjectID = 1,
            ProjectName = "Montgomery Walk",
            Description = "Montgomery Walk is a resident
            CompanyID = 1,
            ImagePath = "project1-2-small.jpg"
        },
```

Figure 14 Database Seeding

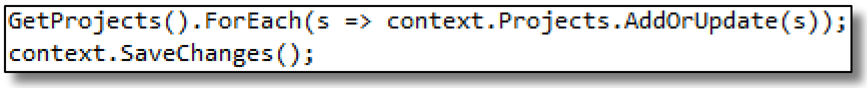
Object and collection initializers, seen above in Figure 14, let developers initialize objects without explicitly calling a constructor. Seen below, the auto-implemented properties for Project make quick property accessors (get and set) that use the private, anonymous backing field created by the compiler.



```
public virtual Project Project { get; set; }
```

Figure 15 Property Accessors

Lambda expressions are inline functions that use the "=>" operator as depicted in the code below for Adding and Updating the Project table during the seed. The left side of the operator defines the input parameters while the right side has the expression or statement block. Lambda expressions are used as arguments when direct methods calls are made to standard query operators.



```
GetProjects().ForEach(s => context.Projects.AddOrUpdate(s));
context.SaveChanges();
```

Figure 16 Lambda Expressions

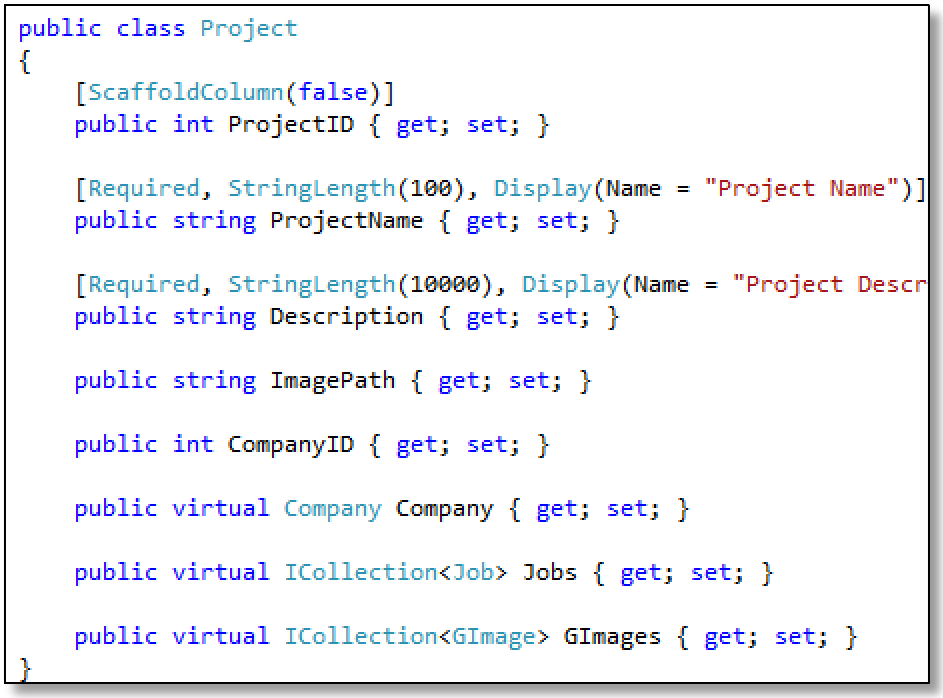
Lastly, extension methods are static methods which can be called as if they were an instance method of a type. Their use is to add new methods to existing types without modifying the type (C# 3.0 Features That Support LINQ, 2008).

## 5.2 Code First Implementation

The project used an EF Code First approach to data access with the creation of a new database. This section will highlight the results of the methodologies found in this implementation and how it supports the thesis of the paper.

### 5.2.1 POCO Classes

One great thing about EF Code First development is being able to write persistence ignorant POCO classes. This means that no matter what data access type is being used, theoretically the same POCO classes can be applied without change. For an example in Figure 17, the Project model does not contain any methods or constructors. This is because methods and constructors could possibly tie the class down to single data accessing framework which is not persistent ignorant. Methods can be defined in the DbContext.

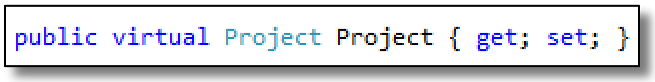


```
public class Project
{
    [ScaffoldColumn(false)]
    public int ProjectID { get; set; }

    [Required, StringLength(100), Display(Name = "Project Name")]
    public string ProjectName { get; set; }

    [Required, StringLength(10000), Display(Name = "Project Descr
    public string Description { get; set; }

    public string ImagePath { get; set; }

    public int CompanyID { get; set; }

    public virtual Company Company { get; set; }

    public virtual ICollection<Job> Jobs { get; set; }

    public virtual ICollection<GImage> GImages { get; set; }
}
```

Figure 17 Persistence Ignorant POCO Class

Also, the Project class does not have dependency on the data layer which is being accessed if the Data Annotations were removed.  This would improve the Griffin Drywall's business application scalability for further development down the line should the need arise to switch from Entity Framework to some other ORM.

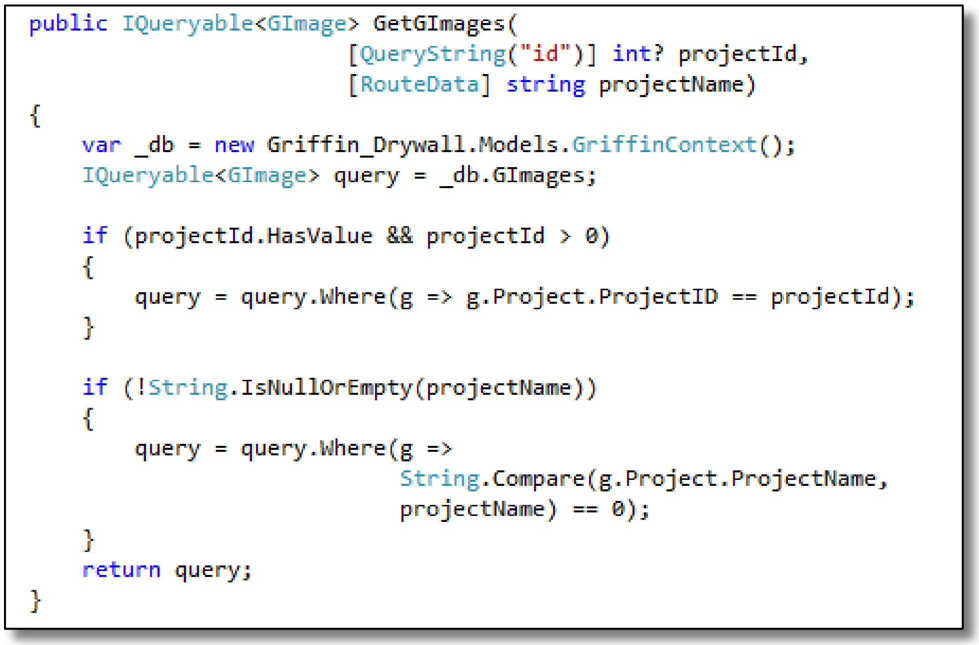
As stated earlier, one of the benefits to using EF is the ability to use Lazy Loading.  In Code First, Lazy Loading can be applied to a relationship by using the keyword, "virtual" when defining the navigational property.  The model, GImage uses the virtual keyword with its relationship with Project as seen below.



```
public virtual Project Project { get; set; }
```

**Figure 18 Declaring Lazy Loading**

This navigational property allowed for the creation of the query method (Figure 19) used in the ProjectDetails page to retrieve the gallery images for the project based on the project id or name.  In the code shown, GetGImages uses "int?" with projectId which specifies that projectId is a nullable type.  Nullable types allow developers to use methods such as HasValue to determine whether the property has a value or is null. These nullable types were introduced to work with variables which could have null values with regards to database concepts.

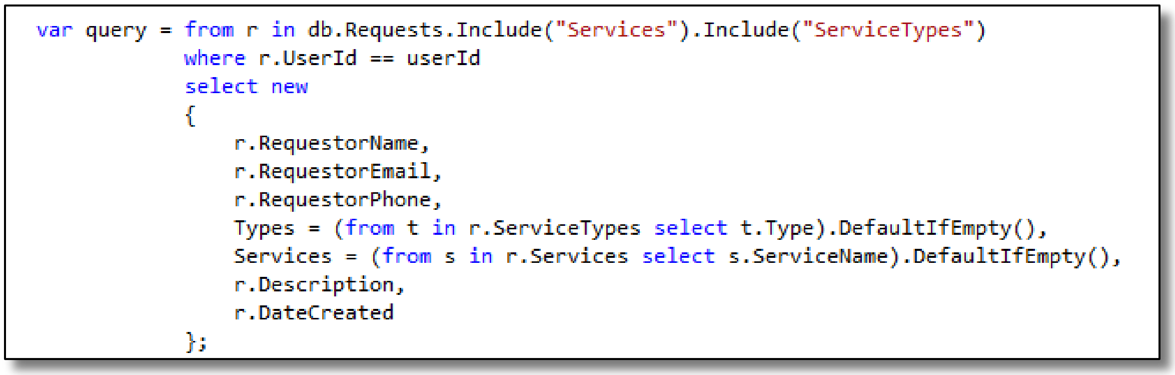


```
public IQueryable<GImage> GetGImages(
                        [QueryString("id")] int? projectId,
                        [RouteData] string projectName)
{
    var _db = new Griffin_Drywall.Models.GriffinContext();
    IQueryable<GImage> query = _db.GImages;

    if (projectId.HasValue && projectId > 0)
    {
        query = query.Where(g => g.Project.ProjectID == projectId);
    }

    if (!String.IsNullOrEmpty(projectName))
    {
        query = query.Where(g =>
                        String.Compare(g.Project.ProjectName,
                        projectName) == 0);
    }
    return query;
}
```

**Figure 19 Lazy Loading Query Example**

Since the GImage class is defined to use Lazy Loading with regards to the Project class, the related Project entity is only loaded the first time the navigational property through GImage is accessed during this query.  This means that Project entity is not loaded until we iterate through the query and access the Project entity through the navigational property (i.e. g.Project.ProjectID).  This is useful to developers who wish to defer loading a database table’s content into memory until it is actually being used.

Debate will rise over the question of Lazy Loading vs. Eager Loading as was the case in the PrevReq page where Eager Loading was better suited for the query.

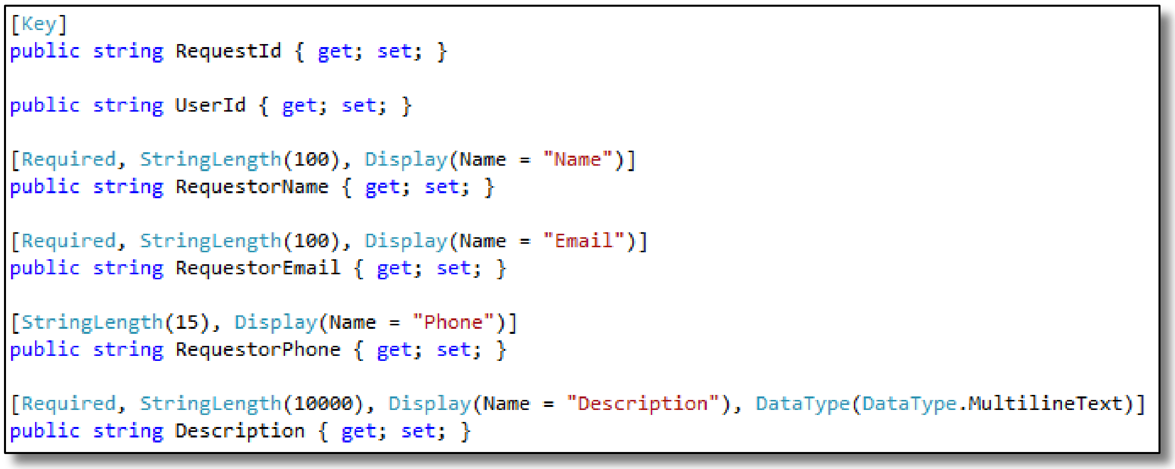


```
var query = from r in db.Requests.Include("Services").Include("ServiceTypes")
            where r.UserId == userId
            select new
            {
                r.RequestorName,
                r.RequestorEmail,
                r.RequestorPhone,
                Types = (from t in r.ServiceTypes select t.Type).DefaultIfEmpty(),
                Services = (from s in r.Services select s.ServiceName).DefaultIfEmpty(),
                r.Description,
                r.DateCreated
            };
```

Figure 20 Request Query Example

Including the entities, Service and ServiceType by extending the query with the extension, Include incurs Eager Loading.  Eager Loading allows the developer to load table contents whenever the query is used. This is useful if it is already known that data of the table is needed during query's runtime like Figure 20's query where Service and ServiceType will need to be accessed to retrieve the Services and Types respectively.

Another improvement about with the POCO classes is the use of Data Annotations.



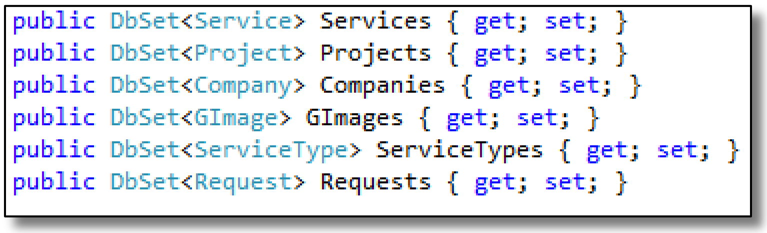
```
[Key]
public string RequestId { get; set; }

public string UserId { get; set; }

[Required, StringLength(100), Display(Name = "Name")]
public string RequestorName { get; set; }

[Required, StringLength(100), Display(Name = "Email")]
public string RequestorEmail { get; set; }

[StringLength(15), Display(Name = "Phone")]
public string RequestorPhone { get; set; }

[Required, StringLength(10000), Display(Name = "Description"), DataType(DataType.MultilineText)]
public string Description { get; set; }
```

Figure 21 Data Annotations

In the example above, the Data Annotations can be used to set certain validation attributes such as requiring the field, RequestorName and having a max length of 100 characters. Adding Data Annotations however, does impede upon the persistence ignorance of the POCO class as it exposes some of the information about the underlying infrastructure of the database. To avoid this, EF Code First allows the usage of the Fluent API in the Database Context which can do everything that Data Annotations can do within the POCO classes.

### 5.2.2 DbContext API

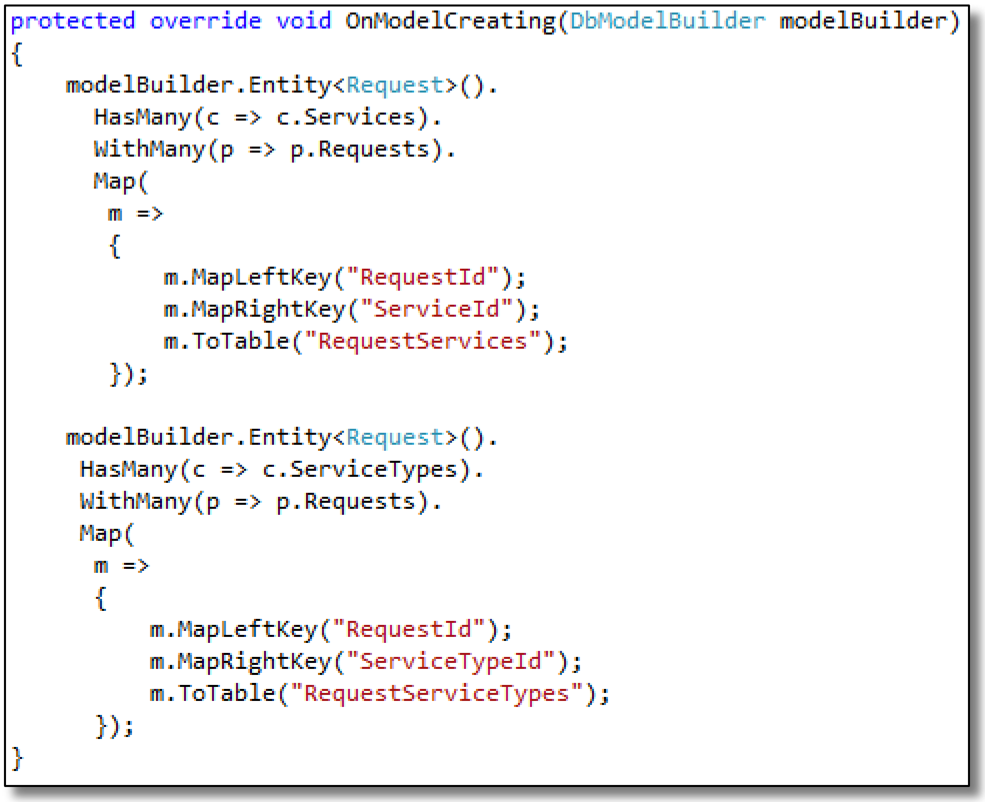
The DbContext of the Griffin Drywall site was fairly straightforward with its implementation. Each POCO class design was given a DbSet member to serve as its basic repository for materializing and saving instances of the class.



```
public DbSet<Service> Services { get; set; }
public DbSet<Project> Projects { get; set; }
public DbSet<Company> Companies { get; set; }
public DbSet<GImage> GImages { get; set; }
public DbSet<ServiceType> ServiceTypes { get; set; }
public DbSet<Request> Requests { get; set; }
```

Figure 22 DbContext Example

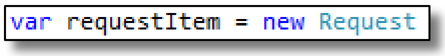
As mentioned (0), the Griffin Drywall DbContext used the Fluent API to overwrite the conventions for building the model of the many-to-many relationships between Request/Services and Request/ServiceTypes.



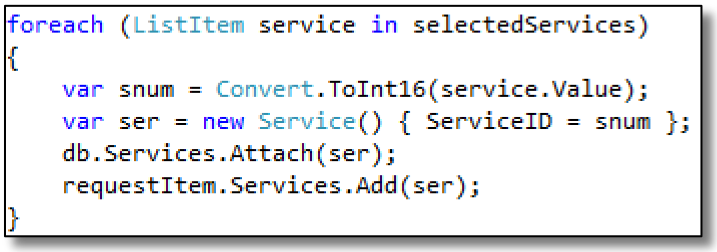
```
protected override void OnModelCreating(DbModelBuilder modelBuilder)
{
    modelBuilder.Entity<Request>().
      HasMany(c => c.Services).
      WithMany(p => p.Requests).
      Map(
       m =>
       {
           m.MapLeftKey("RequestId");
           m.MapRightKey("ServiceId");
           m.ToTable("RequestServices");
       });

    modelBuilder.Entity<Request>().
     HasMany(c => c.ServiceTypes).
     WithMany(p => p.Requests).
     Map(
      m =>
      {
          m.MapLeftKey("RequestId");
          m.MapRightKey("ServiceTypeId");
          m.ToTable("RequestServiceTypes");
      });
}
```

Figure 23 Fluent API Example

The interesting thing here is that unlike traditional approaches, the bridging table between Request/Services and Request/ServiceTypes is being generated automatically. Entity Framework will then be able to self-manage this table on its own without the developer having to write code to insert linkage data into the bridge table. The code below (Figure 24) is used in the Request page is an illustration of how this is done.



```
var requestItem = new Request
```



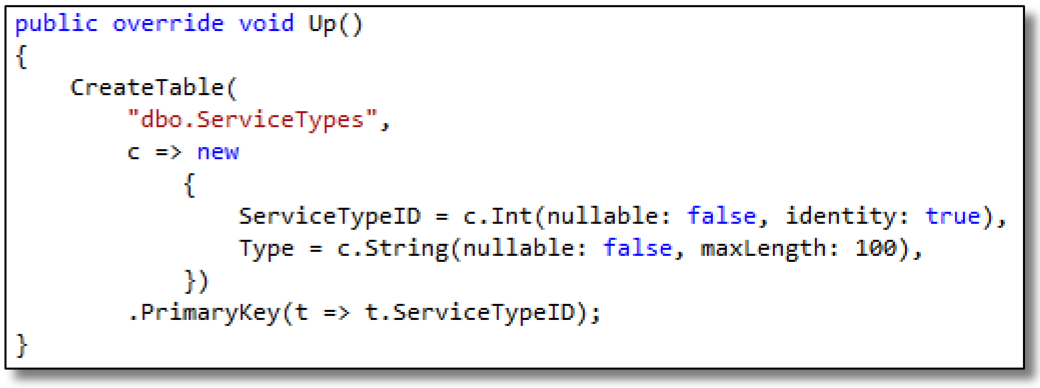
```
foreach (ListItem service in selectedServices)
{
    var snum = Convert.ToInt16(service.Value);
    var ser = new Service() { ServiceID = snum };
    db.Services.Attach(ser);
    requestItem.Services.Add(ser);
}
```

Figure 24 EF Many-to-Many Example

Here, the requestItem is a new Request entity and when Services are added the appropriate values will be inserted into the bridging table. This is how Entity Framework handles many-to-many relationships.

### 5.2.3 Code First Migrations

Code First Migrations gives developers a certain level of scalability when evolving their database. As the work grows, Entity Framework handles it in a capable manner by allowing developers to code their domain classes as a representation of the business changes being applied. Using the Package Manager Console in VS, developers can enable migrations to make these changes to the database schema. The generated SQL for the migrations can be customized further before applying it to the database schema with the Update-Database command.



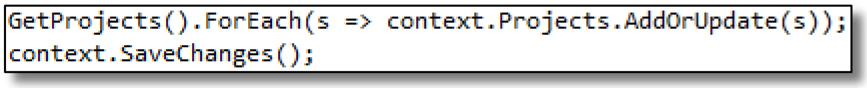
```
public override void Up()
{
    CreateTable(
        "dbo.ServiceTypes",
        c => new
            {
                ServiceTypeID = c.Int(nullable: false, identity: true),
                Type = c.String(nullable: false, maxLength: 100),
            })
        .PrimaryKey(t => t.ServiceTypeID);
}
```

Figure 25 Generated Migration Code

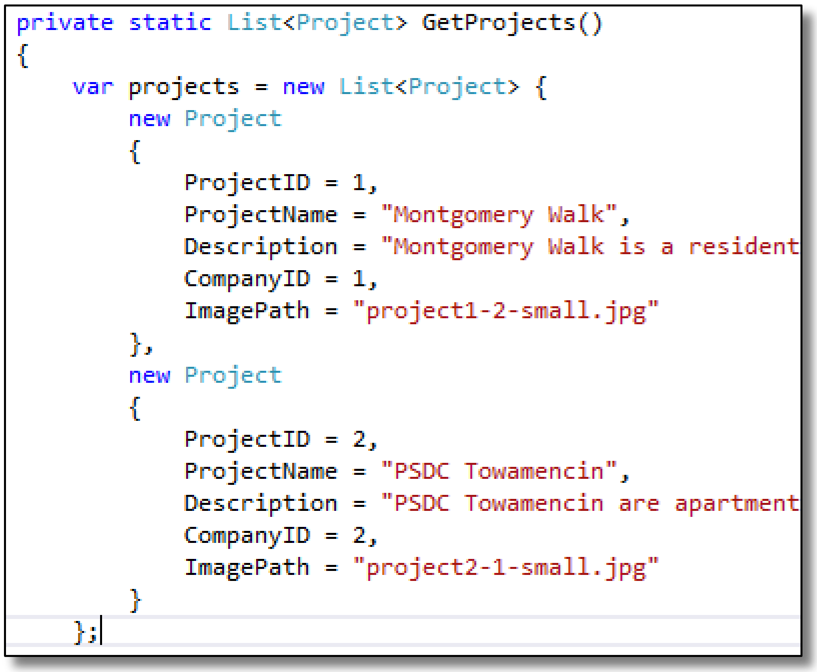
The example above shows an excerpt from the Add-ServiceType migration. This is the generated code of the scaffold created by EF's conventions as well as the Data Annotations within the ServiceType model.

Code First Migrations also allows developers to target specific migrations to update the database. This feature not only gives developers the power to update to a specific migration version but also to downgrade the database to an earlier version. This is ideal for database evolution since changes can be tracked and backed out should implementation problems arise.

Another interesting aspect about Code First Migrations is the configuration file, which permits data seeding when the initial database is made as well as further changes to the conceptual model. The code example (Figure 26) shows the seed data for the Project entity.



```
GetProjects().ForEach(s => context.Projects.AddOrUpdate(s));
context.SaveChanges();
```



```
private static List<Project> GetProjects()
{
    var projects = new List<Project> {
        new Project
        {
            ProjectID = 1,
            ProjectName = "Montgomery Walk",
            Description = "Montgomery Walk is a resident
            CompanyID = 1,
            ImagePath = "project1-2-small.jpg"
        },
        new Project
        {
            ProjectID = 2,
            ProjectName = "PSDC Towamencin",
            Description = "PSDC Towamencin are apartment
            CompanyID = 2,
            ImagePath = "project2-1-small.jpg"
        }
    };
```

Figure 26 Configuration File

While not ideal on a large scale due to size and complexity, seeding the database with data at the start of the application gives developers data to work with while testing out their applications. Seeding the database could be done manually by having the developer use a DBMS to insert the rows for the existing table. However using EF

Migrations method, the configuration file could be shared between developers working on the same database project or create the database from scratch again if needed. The goal in using the seed method in the configuration file is to reduce the time spent manually entering the seed data.

## 5.3 Traditional Approach Comparison

Forming a comparison between Entity Framework and traditional techniques for data access is somewhat difficult in that Entity Framework encapsulates much of the traditional approaches but with greater ease to the developer. The most obvious advantage EF has over traditional approaches is its ability as an ORM to give developers that higher abstraction level while working with database access. With a focus on the Code First approach, developers can build their domain classes from a business entity perspective and let EF's conventions figure out how it maps to the underlying database schema. If the conventions aren't able to infer the mapping between the domain model and the database schema correctly then developers can use Data Annotations or the Fluent API to override the conventions to adjust the mapping. Then with the DbContext, developers can access the collections of entities or an instance of an entity to manipulate data and save changes. Doing so greatly reduces the amount of code a developer would have to write if using a traditional approach of tailoring their own data access layer. This layer of abstraction also promotes scalability as queries are programmed against the conceptual model which allows queries to be translated by Entity Framework for different data sources such as SQL Server or Oracle.

### 5.3.1 Criticism

Critics of ORMs have claimed it “is the Vietnam of Computer Science” (Neward, 2006) or ORMs such as Entity Framework are considered “bloatware” (Fowler, 2012). In the article (Neward, 2006), the analogy between Vietnam and ORMs is used to describe the response to the problem of object to relational data mapping, the solution of creating ORMs, and the problematic outcome of this solution. The article feels that like the US government, developers are unwilling to back out of ORMs (Vietnam) even when they see diminishing returns and more and more overhead. The article uses the term, “quagmire” to define the situation of ORMs where they start out good but get more complicated and burdensome as time proceeds.

Basically, the comparison is that the United States government went into Vietnam wanting to prevent Communism and in the beginning it looked as if they had done just that. However as time went on the US government saw more and more losses without seeing any rewards i.e. diminishing returns. Yet, the mentality stood that they had committed this far so shouldn’t they see it through even though it continued to get worse. Ultimately the US government back out of Vietnam and the war was recorded as one of the most profound defeats ever for the US. Neward makes his case that ORMs fall to the same concerns. Developers wanted to solve the problem of object to relational data mapping and did so by developing object relational mappers. In the beginning, developers observed the benefits of using ORMs but as time proceeded, it became apparent that supporting them created more overhead than the original problem. Yet because most developers were/are so invested in ORMs, they refuse to back out of using them. Neward feels that ORM developers will see the same end that the US government did with the Vietnam War.

The majority of the Neward's argument revolves around impedance mismatch and the ability of an ORM to map the object data model with that of the relational data model. While Entity Framework does greatly reduce impedance mismatch (Entity Framework), it does not completely eliminate it. However as Fowler states, an ORM framework such as EF that allows for developers to avoid 80% of the "repetitive, boiler-plate code" is worth it. Critics in support of traditional approaches will claim that it is better to build their own framework up around their applications with manually coded stored procedures and written SQL statements which avoids impedance mismatch completely.

### 5.3.2 Entity Framework Advantages

The traditional approaches, which create a manually written DAL with either ad hoc or stored procedures, present another advantage of EF in the form of code maintenance (Campbell, 2012). When using stored procedures, if there is a change in the database structure then all queries pertaining to the change will need to be checked and possibly updated. As noted (Campbell, 2012), stored procedures are tedious to manage, requiring developers to "define parameter after parameter" using a "copy, paste, tweak" methodology. This can lead to many bugs and costly errors which necessitate a great deal of troubleshooting. With EF, adding structural changes on the fly is easy and queries will be updated automatically with the change thus saving the developer the time spent maintaining the code for all existing queries.

An example of such a database change would be if there was a database table, Customer with a "CustomerName" field that needed to be changed to "Name." Within a business enterprise, this change may need the analysis of hundreds of stored procedures to examine that there is no reference to "CustomerName" and if so, it is changed to

"Name." With Entity Framework, a change like this would only require an alteration to the mapping from entity, Customer in the conceptual model and the database table, Customer. Afterwards all queries would function properly as they were programmed against the conceptual model and not the relational model as in the case with stored procedures.

Two other areas of comparison are how Entity Framework handles concurrency and caching. By default EF implements an optimistic concurrency model which is similar to the typical approach used in traditional data access. Yet EF Code First also allows the developer to specify a concurrency check attribute with Data Annotations or use the Fluent API to initiate concurrency checking on a property. This property is normally some type of Timestamp, which EF will use to check whether the data has been modified before that change is saved. These concurrency check options ensure that a concurrency exception will be thrown should two users try to update the same row of data. This is different from a traditional standpoint because Entity Framework is doing all the concurrency checking for the developer without much code written.

Caching in Entity Framework makes a difference in performance when running continuous queries over the same data. While caching is definitely seen in most traditional approaches to data access, the Entity Framework applies it in such a way that it remains transparent to the developer and without the developer implementing a caching system on their own. In respect to LINQ-to-Entities, Entity Framework will create a hash of the query being executed and store the generated SQL and the Hash in cache. If the same hash is found in existence then Entity Framework will pull the pre-generated, cached SQL. This proves to be a great performance boost to queries as seen in Sneak

Preview: Entity Framework 5.0 Performance Improvements (dpblogs, 2012) and in Figure 27 .NET 4.0 vs .NET 4.5, which is taken from the article's finding. Figure 27 represents the "relative time spent in the execution of a query that retrieves an entity by its key" (dpblogs, 2012). Implementing the caching of pre-complied queries in .NET 4.5 can be seen to improve, by about 600%, the performance in LINQ-to-Entities.

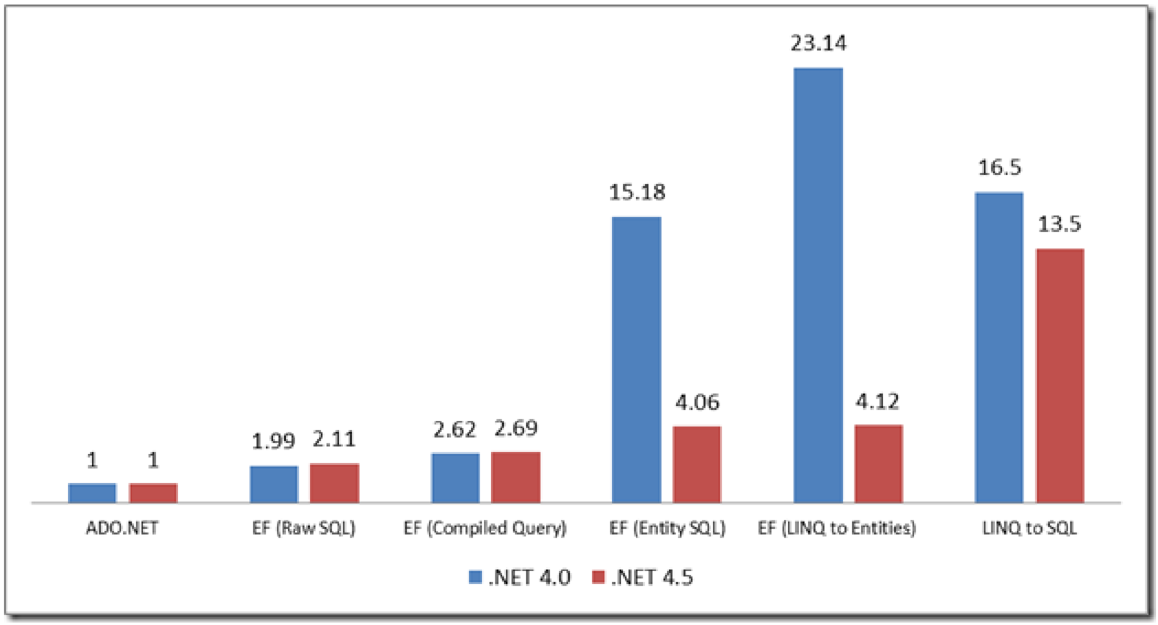
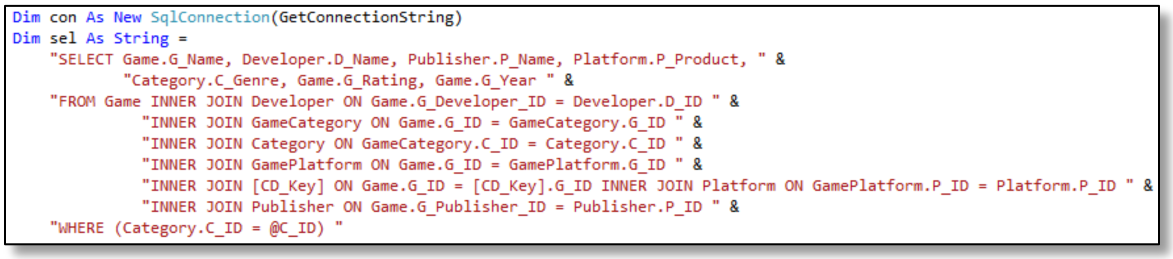


Figure 27 .NET 4.0 vs .NET 4.5

## 5.4 Griffin Drywall Development

During the development of the Griffin Drywall application, there were several noticeable improvements in using the ORM, Entity Framework over the traditional data access approaches like those used in the CIS 623 final project submission (Griffin, 2011). This CIS 623 project (Griffin, 2011) centered on using ADO.NET's SQLDataSource provider and parameterized SQL statements to perform database accessing. The CIS 623 final project submission makes for a comparative example for the improvements observed during the Entity Framework development experience.

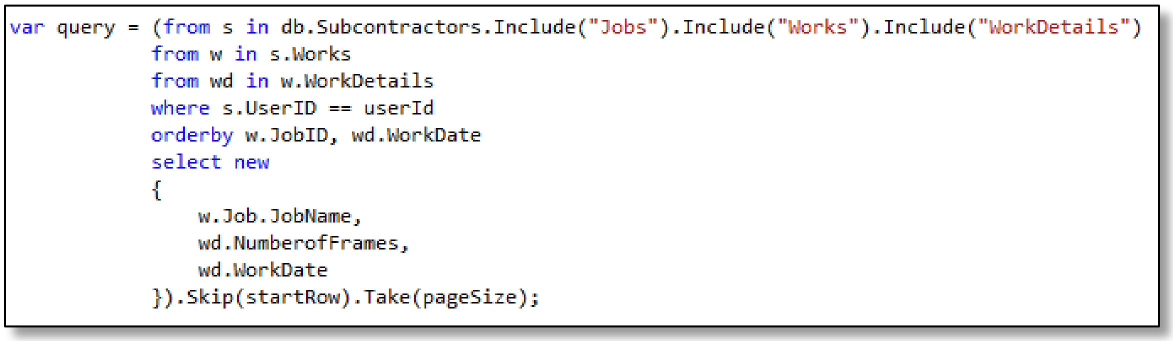
One of the first improvements perceived in the EF development process was the usage of navigational properties with regards to how relationships between entities were used in LINQ queries.  The Figure 28 shows a SQL query, from the CIS 623 project, used to return a number of game titles based on their category ID in the relational database.



```
Dim con As New SqlConnection(GetConnectionString)
Dim sel As String =
    "SELECT Game.G_Name, Developer.D_Name, Publisher.P_Name, Platform.P_Product, " &
            "Category.C_Genre, Game.G_Rating, Game.G_Year " &
    "FROM Game INNER JOIN Developer ON Game.G_Developer_ID = Developer.D_ID " &
              "INNER JOIN GameCategory ON Game.G_ID = GameCategory.G_ID " &
              "INNER JOIN Category ON GameCategory.C_ID = Category.C_ID " &
              "INNER JOIN GamePlatform ON Game.G_ID = GamePlatform.G_ID " &
              "INNER JOIN [CD_Key] ON Game.G_ID = [CD_Key].G_ID INNER JOIN Platform ON GamePlatform.P_ID = Platform.P_ID " &
              "INNER JOIN Publisher ON Game.G_Publisher_ID = Publisher.P_ID " &
    "WHERE (Category.C_ID = @C_ID) "
```

Figure 28 SQL Query Example

While the LINQ query, in Figure 29, is similarly used to return the Work Details information from the Griffin Drywall relational database.



```
var query = (from s in db.Subcontractors.Include("Jobs").Include("Works").Include("WorkDetails")
             from w in s.Works
             from wd in w.WorkDetails
             where s.UserID == userId
             orderby w.JobID, wd.WorkDate
             select new
             {
                 w.Job.JobName,
                 wd.NumberofFrames,
                 wd.WorkDate
             }).Skip(startRow).Take(pageSize);
```

Figure 29 LINQ Query Example

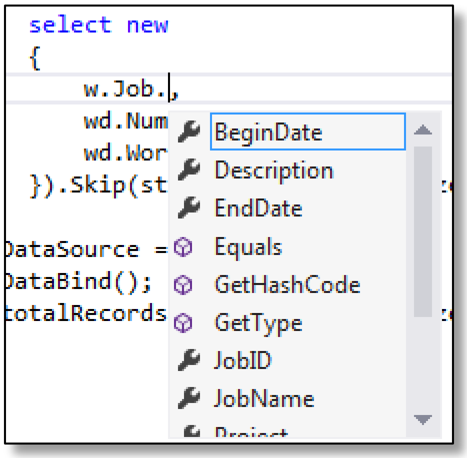
Both queries need to select data found in other relational database tables but only the SQL query needs to know the underlying database structure and make several table joins to return the correct data.  On the other hand, the LINQ query uses the navigational properties within the types being used (Subcontractors, Works, and WorkDetails) to pull the required data.  These navigations are derived from the domain classes' navigational properties. For example "w.Job.JobName" in Figure 29, the "w" is a Works type as define in the Griffin Drywall DbContext and as such has access to the navigational property, Job. The Job navigational property allows the developer to pull properties from the Job entity such as JobName.

For the sake of comparison, the translated SQL version of the LINQ query in Figure 29 is presented:

```
Dim select As String =
        "SELECT Job.JobName, WorkDetail.NumberofFrames, WorkDetail.WorkDate " &
        "FROM Subcontractors AS Subcon " &
        "CROSS JOIN Work AS Work " &
                "CROSS JOIN WorkDetails AS WorkDetail " &
                 "LEFT OUTER JOIN Jobs AS Job ON Job.JobID = Work.JobID " &
        "WHERE Subcon.SubconID = @userID AND Work.SubconID = Subcon.SubconID " &
        "AND WorkDetail.JobID = Work.JobID " &
        "ORDERBY Work.JobID, WorkDetail.WorkDate"
```

Here it should be noted how this query is designed against the relational model of the database. Developers need to know the database schema in order to formulate the correct SQL select statement with relative joins made on the primary keys of those tables. If the primary key should change for one of these tables, this query will need to be updated to reflect the change and keep the joins intact. Such is not the case in the EF LINQ query where the query's joins will be dynamically updated because the query is made against the conceptual model.

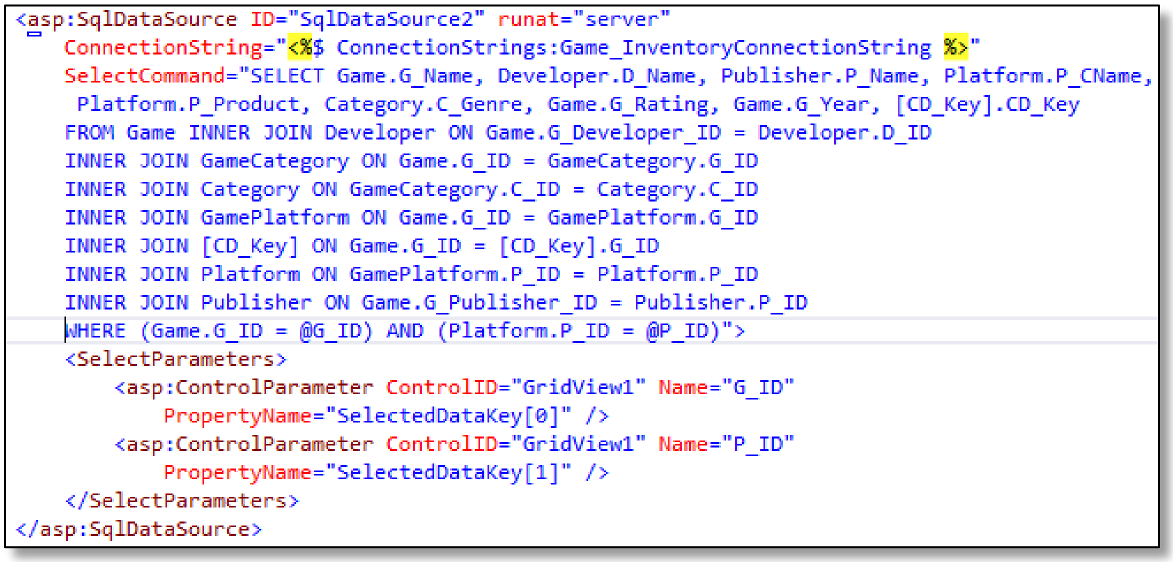
Furthermore, Entity Framework is fully integrated with Visual Studio's IntelliSense, which helps developers reduce programming errors from things like typos and commonly made mistakes. Using the same example from Figure 28 and Figure 29, the written SQL statement is of type string. Any problems within this SQL would not appear until the program's execution of the SQL statement. If there was a typo or a misunderstanding of the database structure, the developer would need to rewrite the SQL statement until it was correct. In the LINQ example, since Entity Framework uses VS's IntelliSense, it will identify whether the query is wrong within the Visual Studio before execution for the majority of developer prone mistakes. Also another advantage here is that IntelliSense will autocomplete for both scalar and navigational properties of an entity. Figure 30 shows an example of this where IntelliSense will show all the available properties and method extensions for Job.



```
select new
{
      w.Job.,
      wd.Num
      wd.Wor
}).Skip(st

DataSource =
DataBind();
totalRecords
```

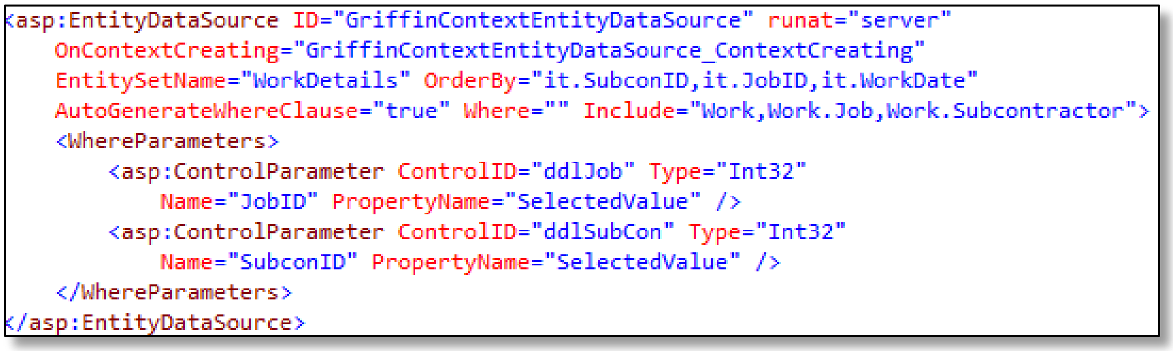
Figure 30 IntelliSense AutoComplete

Another comparative benefit to using Entity Framework can be seen in the contrast between the SqlDataSource, used in the CIS 623 project to display all games in the database and the EntityDataSource, used in the Griffin Drywall application to view all work. The figures, Figure 31 and Figure 32, demonstrate the implementation of the two data sources as they appear in their respectful code.



```
<asp:SqlDataSource ID="SqlDataSource2" runat="server"
    ConnectionString="<%$ ConnectionStrings:Game_InventoryConnectionString %>"
    SelectCommand="SELECT Game.G_Name, Developer.D_Name, Publisher.P_Name, Platform.P_CName,
     Platform.P_Product, Category.C_Genre, Game.G_Rating, Game.G_Year, [CD_Key].CD_Key
    FROM Game INNER JOIN Developer ON Game.G_Developer_ID = Developer.D_ID
    INNER JOIN GameCategory ON Game.G_ID = GameCategory.G_ID
    INNER JOIN Category ON GameCategory.C_ID = Category.C_ID
    INNER JOIN GamePlatform ON Game.G_ID = GamePlatform.G_ID
    INNER JOIN [CD_Key] ON Game.G_ID = [CD_Key].G_ID
    INNER JOIN Platform ON GamePlatform.P_ID = Platform.P_ID
    INNER JOIN Publisher ON Game.G_Publisher_ID = Publisher.P_ID
    WHERE (Game.G_ID = @G_ID) AND (Platform.P_ID = @P_ID)">
    <SelectParameters>
        <asp:ControlParameter ControlID="GridView1" Name="G_ID"
            PropertyName="SelectedDataKey[0]" />
        <asp:ControlParameter ControlID="GridView1" Name="P_ID"
            PropertyName="SelectedDataKey[1]" />
    </SelectParameters>
</asp:SqlDataSource>
```

Figure 31 SqlDataSource Example

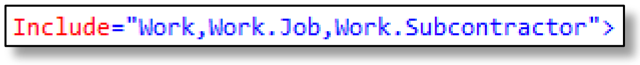


```
<asp:EntityDataSource ID="GriffinContextEntityDataSource" runat="server"
    OnContextCreating="GriffinContextEntityDataSource_ContextCreating"
    EntitySetName="WorkDetails" OrderBy="it.SubconID,it.JobID,it.WorkDate"
    AutoGenerateWhereClause="true" Where="" Include="Work,Work.Job,Work.Subcontractor">
    <WhereParameters>
        <asp:ControlParameter ControlID="ddlJob" Type="Int32"
            Name="JobID" PropertyName="SelectedValue" />
        <asp:ControlParameter ControlID="ddlSubCon" Type="Int32"
            Name="SubconID" PropertyName="SelectedValue" />
    </WhereParameters>
</asp:EntityDataSource>
```

Figure 32 EntityDataSource

In the example figures, the most noticeable change is the SqlDataSource has a huge SQL select statement packaged into it. If a developer wishes to change the database properties within, for example, the Game table, they are going to have to check and possibly update every single query which references the Game table to ensure that the change will not affect the query's execution.  This is not like the EntityDataSource where the entities, WorkDetails, Work, Job and Subcontractor are being used.  In Entity Framework, changes made to the structure of database via Code First Migrations will not affect how this data source functions since the query generated will automatically be updated by Entity Framework.  So if the developer wants to change the Job property, JobName to Name instead, they can do so and not have to change anything in the EntityDataSource.

Emphasizing Entity Framework's inference of written code in data accessing, there are several instances of using Join in Figure 31 to explain the relationship between database tables however in EntityDataSouce, there is only the statement to include these entities in Figure 33.



```
Include="Work,Work.Job,Work.Subcontractor">
```

Figure 33 Include Example

Again, navigational properties demonstrate a unique benefit over using the traditional approach with written SQL.  This example shows that the EntityDataSource will include the entity types and their relationships as they are defined in the conceptual model.  Then Entity Framework will automatically generate the SQL statement needed to perform the requested operation.  Entity Framework generates these SQL statements by using expression trees, which are tree-like data representations of the original requested operation.  This is of great profit to the developers because they no longer need to spend time figuring out the correct joins and relationships needed to retrieve the data from the relational database.

## 6. Conclusion

Microsoft's ADO.NET Entity Framework is "bloatware" but that doesn't necessarily mean it is a bad thing (Fowler, 2012). There is a lot that Entity Framework does to empower the developers with the means to access relational data in a simplified manner using domain specific classes. This paper only covers the implementation of a Code First approach with the creation of a new database, which is only one of the four workflows.  It examines the key features that Entity Framework provides business application development. Yet, this is only the tip of the iceberg for what Entity Framework has to offer.

Though this paper and the development of the Entity Framework business application for Griffin Drywall, a constructive and informative presentation has been given as to the benefits of using the EF ORM with a real-world environment.  It has shown that Entity Framework allows developers to work with data-oriented applications with less code written than traditional approaches by using a higher level of abstraction. Comparatively, everything traditional data access techniques can do Entity Framework was examined to do just as well if not better and without the complexity of robust code development.  Based on the research, this factor is largely because Entity Framework was built-in to Microsoft's existing data access technology, ADO.NET.  It is bloatware.  It was designed to interface with just about every form of data access to give developers the scalability of not being tied down to one form or another.

This is not to say Entity Framework is the best and should be used in the development of all business applications hence forth.  Far from it, though Microsoft would prefer it.  As stated Entity Framework is going to take care of 80-90% of the coding of all queries and mappings a business application is ever going to use. That other 10-20% is where critics are really going to evaluate Entity Framework's effectiveness. The general opinion amongst developers about the choice of whether to use Entity Framework or the traditional methods is that it is very scenario dependent. Meaning that it is best practice to look at what type of business application is being developed and weigh out the tradeoffs of each approach.

This paper has displayed beneficial features of Entity Framework Code First within a business web application as well as a comparison between it and traditional data access. It has demonstrated that Entity Framework as an ORM has ability to endow their application developers with a higher level of data access abstraction.

## 6.1 Further Research

There are many interesting places to pursue in this area of research. First and foremost, an examination into Model First approach would be a good start. This paper focuses on the Code First approach and while the two approaches are largely similar there were some differences between the two. In my opinion, Model First looked to be a more graphical way of designing the database so a good comparison between how Entity Framework conventions translates the model into generated code versus how Code First generates the model would expand this research. Also an understanding of how EF's XML language and structure is formed for all three models (conceptual, storage, and mapping) would be another topic of interest.

Another good area to expand this topic would be to compare Entity Framework against the current generation ORMs such as Hibernate. Though it might not be "the" leading example in the ORM department, Entity Framework is making large strides with each new release and would be surprising if Microsoft did not improve it in the next couple of years to surpass most ORM technology on the current market.  There have already been a few papers and articles published regarding which of these two is the better, as well as countless online forum discussions/arguments.  Expanding or duplicating the research using a developed benchmark application and testing the performance of each ORMs' ability to accomplish certain tasks would prove to be a very provocative read.  Though quite frankly, it would have to be a very well developed implementation of both ORMs because publishing those results, given the heated disputes that have arisen, could cause some criticism.

# 7. Appendix A

The source code for the domain name POCO classes for Griffin Drywall.

## 7.1 Company

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.ComponentModel.DataAnnotations;

namespace Griffin_Drywall.Models
{
    public class Company
    {
        [ScaffoldColumn(false)]
        public int CompanyID { get; set; }

        [Required, StringLength(100), Display(Name = "Name")]
        public string CompanyName { get; set; }

        [Required, StringLength(10000), Display(Name = "Company Description"),
DataType(DataType.MultilineText)]
        public string Description { get; set; }

        [StringLength(10000), Display(Name = "Company Description 2nd"),
DataType(DataType.MultilineText)]
        public string Description2 { get; set; }

        public virtual ICollection<Project> Projects { get; set; }

        public string ImagePath { get; set; }
    }
}
```

## 7.2 GImage

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.ComponentModel.DataAnnotations;

namespace Griffin_Drywall.Models
{
    public class GImage
    {
        [ScaffoldColumn(false)]
        public int GImageID { get; set; }
        public int ProjectID { get; set; }

        public string ImagePath { get; set; }

        public virtual Project Project { get; set; }
    }
}
```

## 7.3 Job

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.ComponentModel.DataAnnotations;

namespace Griffin_Drywall.Models
{
    public class Job
    {
        [ScaffoldColumn(false)]
        public int JobID { get; set; }

        [Required, StringLength(100), Display(Name = "Name")]
        public string JobName { get; set; }

        [StringLength(10000), Display(Name = "Job Description"),
DataType(DataType.MultilineText)]
        public string Description { get; set; }

        public DateTime BeginDate { get; set; }

        public DateTime EndDate { get; set; }

        public int ProjectID { get; set; }

        public virtual Project Project { get; set; }

        public virtual ICollection<Work> Works { get; set; }
    }
}
```

## 7.4 Project

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.ComponentModel.DataAnnotations;

namespace Griffin_Drywall.Models
{
    public class Project
    {
        [ScaffoldColumn(false)]
        public int ProjectID { get; set; }

        [Required, StringLength(100), Display(Name = "Project Name")]
        public string ProjectName { get; set; }

        [Required, StringLength(10000), Display(Name = "Project Description"),
DataType(DataType.MultilineText)]
        public string Description { get; set; }

        public string ImagePath { get; set; }

        public int CompanyID { get; set; }

        public virtual Company Company { get; set; }

        public virtual ICollection<Job> Jobs { get; set; }

        public virtual ICollection<GImage> GImages { get; set; }
    }
}
```

## 7.5 Request

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.ComponentModel.DataAnnotations;

namespace Griffin_Drywall.Models
{
    public class Request
    {
        [Key]
        public string RequestId { get; set; }

        public string UserId { get; set; }

        [Required, StringLength(100), Display(Name = "Name")]
        public string RequestorName { get; set; }

        [Required, StringLength(100), Display(Name = "Email")]
        public string RequestorEmail { get; set; }

        [StringLength(15), Display(Name = "Phone")]
        public string RequestorPhone { get; set; }

        [Required, StringLength(10000), Display(Name = "Description"),
DataType(DataType.MultilineText)]
```

```
        public string Description { get; set; }

        public System.DateTime DateCreated { get; set; }

        public virtual ICollection<Service> Services { get; set; }

        public virtual ICollection<ServiceType> ServiceTypes { get; set; }

        public Request()
        {
            Services = new HashSet<Service>();
            ServiceTypes = new HashSet<ServiceType>();
        }

    }
}
```

## 7.6 Service

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.ComponentModel.DataAnnotations;

namespace Griffin_Drywall.Models
{
    public class Service
    {
        [ScaffoldColumn(false)]
        public int ServiceID { get; set; }

        [Required, StringLength(100), Display(Name = "Name")]
        public string ServiceName { get; set; }

        [Required, StringLength(10000), Display(Name = "Service Description"),
DataType(DataType.MultilineText)]
        public string Description { get; set; }

        [StringLength(10000), Display(Name = "Service Description 2nd"),
DataType(DataType.MultilineText)]
        public string Description2 { get; set; }

        [StringLength(10000), Display(Name = "Service Description 3rd"),
DataType(DataType.MultilineText)]
        public string Description3 { get; set; }

        public string ImagePath { get; set; }

        public ICollection<Request> Requests { get; set; }
        public Service()
        {
            Requests = new HashSet<Request>();
        }
    }
}
```

## 7.7 ServiceType

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.ComponentModel.DataAnnotations;

namespace Griffin_Drywall.Models
{
    public class ServiceType
    {
        [ScaffoldColumn(false)]
        public int ServiceTypeID { get; set; }

        [Required, StringLength(100), Display(Name = "Type")]
        public string Type { get; set; }

        public ICollection<Request> Requests { get; set; }

        public ServiceType()
        {
            Requests = new HashSet<Request>();
        }
    }
}
```

## 7.8 Subcontractor

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.ComponentModel.DataAnnotations;

namespace Griffin_Drywall.Models
{
    public class Subcontractor
    {
        [Key, ScaffoldColumn(false)]
        public int SubconID { get; set; }

        [Required, StringLength(100), Display(Name = "Name")]
        public string SubconName { get; set; }
        [StringLength(10000), Display(Name = "Subcontractor Description"),
DataType(DataType.MultilineText)]
        public string Description { get; set; }

        [Required, Display(Name = "UserID")]
        public Guid UserID { get; set; }

        public virtual ICollection<Work> Works { get; set; }
    }
}
```

## 7.9 Work

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.ComponentModel.DataAnnotations;

namespace Griffin_Drywall.Models
{
    public class Work
    {
        public int JobID { get; set; }
        public int SubconID { get; set; }
        public virtual Job Job { get; set; }
        public virtual Subcontractor Subcontractor{ get; set; }
        public virtual ICollection<WorkDetail> WorkDetails { get; set; }
    }
}
```

## 7.10 WorkDetails

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.ComponentModel.DataAnnotations;

namespace Griffin_Drywall.Models
{
    public class WorkDetail
    {
        [Key]
        public int WorkDetailID { get; set; }

        public int JobID { get; set; }
        public int SubconID { get; set; }

        [Required]
        public int NumberofFrames { get; set; }

        [Required]
        public DateTime WorkDate { get; set; }

        public System.DateTime DateCreated { get; set; }
        public virtual Work Work { get; set; }
    }
}
```

# 8. Appendix B

## 8.1 GriffinContext DbContext

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Data.Entity;
using Griffin_Drywall.Models;

namespace Griffin_Drywall.Models
{
    public class GriffinContext : DbContext
    {
        public GriffinContext()
            : base("Griffin_Drywall")
        { }
        public DbSet<Service> Services { get; set; }
        public DbSet<Project> Projects { get; set; }
        public DbSet<Company> Companies { get; set; }
        public DbSet<GImage> GImages { get; set; }
        public DbSet<ServiceType> ServiceTypes { get; set; }
        public DbSet<Request> Requests { get; set; }
        public DbSet<Job> Jobs { get; set; }
        public DbSet<Subcontractor> Subcontractors { get; set; }
        public DbSet<Work> Works { get; set; }
        public DbSet<WorkDetail> WorkDetails { get; set; }

        protected override void OnModelCreating(DbModelBuilder modelBuilder)
        {
            modelBuilder.Entity<Request>().
              HasMany(c => c.Services).
              WithMany(p => p.Requests).
              Map(
               m =>
               {
                   m.MapLeftKey("RequestId");
                   m.MapRightKey("ServiceId");
                   m.ToTable("RequestServices");
               });

            modelBuilder.Entity<Request>().
             HasMany(c => c.ServiceTypes).
             WithMany(p => p.Requests).
             Map(
              m =>
              {
                  m.MapLeftKey("RequestId");
                  m.MapRightKey("ServiceTypeId");
                  m.ToTable("RequestServiceTypes");
              });

            modelBuilder.Entity<Work>()
               .HasKey(w => new { w.SubconID, w.JobID });

            modelBuilder.Entity<Subcontractor>()
```

```
                        .HasMany(s => s.Works)
                        .WithRequired()
                        .HasForeignKey(w => w.SubconID);

            modelBuilder.Entity<Job>()
                        .HasMany(j => j.Works)
                        .WithRequired()
                        .HasForeignKey(w => w.JobID);
        }
    }
}
```

# 9. Appendix C

## 9.1 Initial Migration (First)

```
namespace Griffin_Drywall.Migrations
{
    using System;
    using System.Data.Entity.Migrations;

    public partial class First : DbMigration
    {
        public override void Up()
        {
            CreateTable(
                "dbo.Services",
                c => new
                    {
                        ServiceID = c.Int(nullable: false, identity: true),
                        ServiceName = c.String(nullable: false, maxLength: 100),
                        Description = c.String(nullable: false),
                        Description2 = c.String(),
                        Description3 = c.String(),
                        ImagePath = c.String(),
                    })
                .PrimaryKey(t => t.ServiceID);

            CreateTable(
                "dbo.Projects",
                c => new
                    {
                        ProjectID = c.Int(nullable: false, identity: true),
                        ProjectName = c.String(nullable: false, maxLength: 100),
                        Description = c.String(nullable: false),
                        ImagePath = c.String(),
                        CompanyID = c.Int(nullable: false),
                    })
                .PrimaryKey(t => t.ProjectID)
                .ForeignKey("dbo.Companies", t => t.CompanyID, cascadeDelete:
true)
                .Index(t => t.CompanyID);

            CreateTable(
                "dbo.Companies",
                c => new
                    {
                        CompanyID = c.Int(nullable: false, identity: true),
                        CompanyName = c.String(nullable: false, maxLength: 100),
                        Description = c.String(nullable: false),
                        Description2 = c.String(),
                        ImagePath = c.String(),
                    })
                .PrimaryKey(t => t.CompanyID);

            CreateTable(
                "dbo.GImages",
                c => new
                    {
                        GImageID = c.Int(nullable: false, identity: true),
                        ProjectID = c.Int(nullable: false),
                        ImagePath = c.String(),
                    })
                .PrimaryKey(t => t.GImageID)
                .ForeignKey("dbo.Projects", t => t.ProjectID, cascadeDelete: true)
                .Index(t => t.ProjectID);

        }

        public override void Down()
        {
            DropIndex("dbo.GImages", new[] { "ProjectID" });
            DropIndex("dbo.Projects", new[] { "CompanyID" });
            DropForeignKey("dbo.GImages", "ProjectID", "dbo.Projects");
            DropForeignKey("dbo.Projects", "CompanyID", "dbo.Companies");
            DropTable("dbo.GImages");
            DropTable("dbo.Companies");
            DropTable("dbo.Projects");
            DropTable("dbo.Services");
        }
    }
}
```

## 9.2 Add-ServiceType

```
namespace Griffin_Drywall.Migrations
{
    using System;
    using System.Data.Entity.Migrations;

    public partial class AddServiceType : DbMigration
    {
        public override void Up()
        {
            CreateTable(
                "dbo.ServiceTypes",
                c => new
                    {
                        ServiceTypeID = c.Int(nullable: false, identity: true),
                        Type = c.String(nullable: false, maxLength: 100),
                    })
                .PrimaryKey(t => t.ServiceTypeID);

        }

        public override void Down()
        {
            DropTable("dbo.ServiceTypes");
        }
    }
}
```

## 9.3 Add-Request

```
namespace Griffin_Drywall.Migrations
{
```

```
using System;
using System.Data.Entity.Migrations;

public partial class AddRequest : DbMigration
{
    public override void Up()
    {
        CreateTable(
            "dbo.Requests",
            c => new
                {
                    RequestId = c.String(nullable: false, maxLength: 128),
                    UserId = c.String(),
                    RequestorName = c.String(nullable: false, maxLength: 100),
                    RequestorEmail = c.String(nullable: false, maxLength: 100),
                    RequestorPhone = c.String(maxLength: 15),
                    Description = c.String(nullable: false),
                    DateCreated = c.DateTime(nullable: false),
                })
            .PrimaryKey(t => t.RequestId);

        CreateTable(
            "dbo.RequestServices",
            c => new
                {
                    RequestId = c.String(nullable: false, maxLength: 128),
                    ServiceId = c.Int(nullable: false),
                })
            .PrimaryKey(t => new { t.RequestId, t.ServiceId })
            .ForeignKey("dbo.Requests", t => t.RequestId, cascadeDelete: true)
            .ForeignKey("dbo.Services", t => t.ServiceId, cascadeDelete: true)
            .Index(t => t.RequestId)
            .Index(t => t.ServiceId);

        CreateTable(
            "dbo.RequestServiceTypes",
            c => new
                {
                    RequestId = c.String(nullable: false, maxLength: 128),
                    ServiceTypeId = c.Int(nullable: false),
                })
            .PrimaryKey(t => new { t.RequestId, t.ServiceTypeId })
            .ForeignKey("dbo.Requests", t => t.RequestId, cascadeDelete: true)
            .ForeignKey("dbo.ServiceTypes", t => t.ServiceTypeId, cascadeDelete: true)
            .Index(t => t.RequestId)
            .Index(t => t.ServiceTypeId);

    }

    public override void Down()
    {
            DropIndex("dbo.RequestServiceTypes", new[] { "ServiceTypeId" });
            DropIndex("dbo.RequestServiceTypes", new[] { "RequestId" });
            DropIndex("dbo.RequestServices", new[] { "ServiceId" });
            DropIndex("dbo.RequestServices", new[] { "RequestId" });
            DropForeignKey("dbo.RequestServiceTypes", "ServiceTypeId", "dbo.ServiceTypes");
            DropForeignKey("dbo.RequestServiceTypes", "RequestId", "dbo.Requests");
            DropForeignKey("dbo.RequestServices", "ServiceId", "dbo.Services");
            DropForeignKey("dbo.RequestServices", "RequestId", "dbo.Requests");
            DropTable("dbo.RequestServiceTypes");
            DropTable("dbo.RequestServices");
            DropTable("dbo.Requests");
        }
    }
}
```

## 9.4 Add-Job

```
namespace Griffin_Drywall.Migrations
{
    using System;
    using System.Data.Entity.Migrations;
    
    public partial class AddJob : DbMigration
    {
        public override void Up()
        {
            CreateTable(
                "dbo.Jobs",
                c => new
                    {
                        JobID = c.Int(nullable: false, identity: true),
                        JobName = c.String(nullable: false, maxLength: 100),
                        Description = c.String(nullable: false),
                        ProjectID = c.Int(nullable: false),
                    })
                .PrimaryKey(t => t.JobID)
                .ForeignKey("dbo.Projects", t => t.ProjectID, cascadeDelete: true)
                .Index(t => t.ProjectID);
            
        }
        
        public override void Down()
        {
            DropIndex("dbo.Jobs", new[] { "ProjectID" });
            DropForeignKey("dbo.Jobs", "ProjectID", "dbo.Projects");
            DropTable("dbo.Jobs");
        }
    }
}
```

## 9.5 Add-Subcon

```
namespace Griffin_Drywall.Migrations
{
    using System;
    using System.Data.Entity.Migrations;

    public partial class AddSubcon : DbMigration
    {
        public override void Up()
        {
            CreateTable(
                "dbo.Subcontractors",
                c => new
                    {
                        SubconID = c.Int(nullable: false, identity: true),
                        SubconName = c.String(nullable: false, maxLength: 100),
                        Description = c.String(),
                        UserId = c.String(nullable: false),
                        UserName = c.String(nullable: false),
                    })
                .PrimaryKey(t => t.SubconID);

            AddColumn("dbo.Jobs", "BeginDate", c => c.DateTime(nullable: false));
            AddColumn("dbo.Jobs", "EndDate", c => c.DateTime(nullable: false));
            AlterColumn("dbo.Jobs", "Description", c => c.String());
        }

        public override void Down()
        {
            AlterColumn("dbo.Jobs", "Description", c => c.String(nullable:
false));
            DropColumn("dbo.Jobs", "EndDate");
            DropColumn("dbo.Jobs", "BeginDate");
            DropTable("dbo.Subcontractors");
        }
    }
}
```

## 9.6 Add-Work

```
namespace Griffin_Drywall.Migrations
{
    using System;
    using System.Data.Entity.Migrations;

    public partial class AddWork : DbMigration
    {
        public override void Up()
        {
            CreateTable(
                "dbo.Works",
                c => new
                    {
                        SubconID = c.Int(nullable: false),
                        JobID = c.Int(nullable: false),
                        NumberofFrames = c.Int(nullable: false),
                        WorkDate = c.DateTime(nullable: false),
                        DateCreated = c.DateTime(nullable: false),
                        Job_JobID = c.Int(),
                        Subcontractor_SubconID = c.Int(),
                    })
                .PrimaryKey(t => new { t.SubconID, t.JobID })
                .ForeignKey("dbo.Jobs", t => t.Job_JobID)
                .ForeignKey("dbo.Subcontractors", t => t.SubconID, cascadeDelete:
true)
                .ForeignKey("dbo.Subcontractors", t => t.Subcontractor_SubconID)
                .ForeignKey("dbo.Jobs", t => t.JobID, cascadeDelete: true)
                .Index(t => t.Job_JobID)
                .Index(t => t.SubconID)
                .Index(t => t.Subcontractor_SubconID)
                .Index(t => t.JobID);

        }

        public override void Down()
        {
            DropIndex("dbo.Works", new[] { "JobID" });
            DropIndex("dbo.Works", new[] { "Subcontractor_SubconID" });
            DropIndex("dbo.Works", new[] { "SubconID" });
            DropIndex("dbo.Works", new[] { "Job_JobID" });
            DropForeignKey("dbo.Works", "JobID", "dbo.Jobs");
            DropForeignKey("dbo.Works", "Subcontractor_SubconID",
"dbo.Subcontractors");
            DropForeignKey("dbo.Works", "SubconID", "dbo.Subcontractors");
            DropForeignKey("dbo.Works", "Job_JobID", "dbo.Jobs");
            DropTable("dbo.Works");
        }
    }
}
```

## 9.7 SubCon-Change

```
namespace Griffin_Drywall.Migrations
{
    using System;
    using System.Data.Entity.Migrations;

    public partial class SubConChange : DbMigration
    {
        public override void Up()
        {
            AlterColumn("dbo.Subcontractors", "UserID", c => c.String(nullable:
false));
            DropColumn("dbo.Subcontractors", "UserName");
        }

        public override void Down()
        {
            AddColumn("dbo.Subcontractors", "UserName", c => c.String(nullable:
false));
            AlterColumn("dbo.Subcontractors", "UserId", c => c.String(nullable:
false));
        }
    }
}
```

## 9.8 SubCon-Change2

```
namespace Griffin_Drywall.Migrations
{
    using System;
    using System.Data.Entity.Migrations;

    public partial class SubConChange2 : DbMigration
    {
        public override void Up()
        {
            AlterColumn("dbo.Subcontractors", "UserID", c => c.Int(nullable:
false));
        }

        public override void Down()
        {
            AlterColumn("dbo.Subcontractors", "UserID", c => c.String(nullable:
false));
        }
    }
}
```

## 9.9 SubCon-Change3

```
namespace Griffin_Drywall.Migrations
{
    using System;
    using System.Data.Entity.Migrations;

    public partial class SubConChange3 : DbMigration
    {
        public override void Up()
        {
            AlterColumn("dbo.Subcontractors", "UserID", c => c.String(nullable:
false));
        }

        public override void Down()
        {
            AlterColumn("dbo.Subcontractors", "UserID", c => c.Int(nullable:
false));
        }
    }
}
```

## 9.10 SubCon-Change4

```
namespace Griffin_Drywall.Migrations
{
    using System;
    using System.Data.Entity.Migrations;

    public partial class SubConChange4 : DbMigration
    {
        public override void Up()
        {
            AlterColumn("dbo.Subcontractors", "UserID", c => c.Guid(nullable: 
false));
        }
        
        public override void Down()
        {
            AlterColumn("dbo.Subcontractors", "UserID", c => c.String(nullable: 
false));
        }
    }
}
```

## 9.11 Work-Change2

```
namespace Griffin_Drywall.Migrations
{
    using System;
    using System.Data.Entity.Migrations;
    
    public partial class WorkChange2 : DbMigration
    {
        public override void Up()
        {
            DropPrimaryKey("dbo.Works", new[] { "SubconID", "JobID" });
            AddPrimaryKey("dbo.Works", new[] { "SubconID", "JobID", "WorkDate" });
        }
        
        public override void Down()
        {
            DropPrimaryKey("dbo.Works", new[] { "SubconID", "JobID", "WorkDate" 
});
            AddPrimaryKey("dbo.Works", new[] { "SubconID", "JobID" });
        }
    }
}
```

## 9.12 Add-WorkDetails

```
namespace Griffin_Drywall.Migrations
{
    using System;
    using System.Data.Entity.Migrations;
    
    public partial class AddWorkDetails : DbMigration
    {
        public override void Up()
        {
            CreateTable(
                "dbo.WorkDetails",
                c => new
                    {
                        WorkDetailID = c.Int(nullable: false, identity: true),
                        JobID = c.Int(nullable: false),
                        SubconID = c.Int(nullable: false),
                        NumberofFrames = c.Int(nullable: false),
                        WorkDate = c.DateTime(nullable: false),
                        DateCreated = c.DateTime(nullable: false),
                    })
                .PrimaryKey(t => t.WorkDetailID);

            AddColumn("dbo.Works", "WorkDetail_WorkDetailID", c => c.Int());
            DropPrimaryKey("dbo.Works", new[] { "SubconID", "JobID", "WorkDate"
});
            AddPrimaryKey("dbo.Works", new[] { "SubconID", "JobID" });
            AddForeignKey("dbo.Works", "WorkDetail_WorkDetailID",
"dbo.WorkDetails", "WorkDetailID");
            CreateIndex("dbo.Works", "WorkDetail_WorkDetailID");
            DropColumn("dbo.Works", "WorkDate");
            DropColumn("dbo.Works", "NumberofFrames");
            DropColumn("dbo.Works", "DateCreated");
        }

        public override void Down()
        {
            AddColumn("dbo.Works", "DateCreated", c => c.DateTime(nullable:
false));
            AddColumn("dbo.Works", "NumberofFrames", c => c.Int(nullable: false));
            AddColumn("dbo.Works", "WorkDate", c => c.DateTime(nullable: false));
            DropIndex("dbo.Works", new[] { "WorkDetail_WorkDetailID" });
            DropForeignKey("dbo.Works", "WorkDetail_WorkDetailID",
"dbo.WorkDetails");
            DropPrimaryKey("dbo.Works", new[] { "SubconID", "JobID" });
            AddPrimaryKey("dbo.Works", new[] { "SubconID", "JobID", "WorkDate" });
            DropColumn("dbo.Works", "WorkDetail_WorkDetailID");
            DropTable("dbo.WorkDetails");
        }
    }
}
```

## 9.13 Change-WorkDetails

```
namespace Griffin_Drywall.Migrations
{
    using System;
    using System.Data.Entity.Migrations;

    public partial class ChangeWorkDetails1 : DbMigration
    {
        public override void Up()
        {
            DropForeignKey("dbo.Works", "WorkDetail_WorkDetailID",
"dbo.WorkDetails");
            DropIndex("dbo.Works", new[] { "WorkDetail_WorkDetailID" });
            AddForeignKey("dbo.WorkDetails", new[] { "SubconID", "JobID" },
"dbo.Works", new[] { "SubconID", "JobID" }, cascadeDelete: true);
            CreateIndex("dbo.WorkDetails", new[] { "SubconID", "JobID" });
            DropColumn("dbo.Works", "WorkDetail_WorkDetailID");
        }

        public override void Down()
        {
            AddColumn("dbo.Works", "WorkDetail_WorkDetailID", c => c.Int());
            DropIndex("dbo.WorkDetails", new[] { "SubconID", "JobID" });
            DropForeignKey("dbo.WorkDetails", new[] { "SubconID", "JobID" },
"dbo.Works");
            CreateIndex("dbo.Works", "WorkDetail_WorkDetailID");
            AddForeignKey("dbo.Works", "WorkDetail_WorkDetailID",
"dbo.WorkDetails", "WorkDetailID");
        }
    }
}
```

# 10. Appendix D

## 10.1 Main Pages

### 10.1.1 Contact.aspx

```
<%@ Page Title="Griffin Drywall | Contact" Language="C#"
MasterPageFile="~/Site.Master" AutoEventWireup="true" CodeBehind="Contact.aspx.cs"
Inherits="Griffin_Drywall.Contact" %>

<asp:Content runat="server" ID="BodyContent" ContentPlaceHolderID="MainContent">
    <div class="float-left contactUs">
    <article class="Ctitle">
        <h1>Contact Us</h1>
    </article>
    <article class="contactUs2">
        <div id="contactUsForm">
            <p>If you have any questions please feel free to contact us, we look
forward to hearing from you! </p>
            <br />
            <header>Griffin Drywall:</header>
            <p>
                <span>P.O. Box 447</span>
            </p>
            <p>
                <span>Plumsteadville, PA 18949</span>
            </p>
            <header>Office:</header>
            <p>
                <span>215.362.5146</span>
            </p>
            <header>Fax:</header>
            <p>
                <span>215.534.0157</span>
            </p>
            <header>Email:</header>
            <p>
                <span>Use the form to send us an email or you can contact us
directly at griffindrywall@comcast.net</span>
            </p>
        </div>
    </article>
    </div>
    <asp:Panel ID="emailPanel" runat="server">
    <article id="contactUsEmail">
        <div class="contactUsEmail">
            <table>
                <tr>
                    <td>
                        <fieldset>
                            <legend>Contact Details</legend>
                            <table>
                                <tr>

                                    <td>Name:
                                    </td>
                                    <td class="contactPad">
```

```
                                        <asp:TextBox class="contactTxtBox" 
ID="nameBox" runat="server" />
                                         <asp:RequiredFieldValidator 
ID="nameReqVal" ControlToValidate="nameBox"
                                            ErrorMessage="Enter Name" 
CssClass="field-validation-error" runat="server">*</asp:RequiredFieldValidator>
                                    </td>
                                </tr>
                                <tr>
                                    <td >Email:
                                    </td>
                                    <td class="contactPad">
                                        <asp:TextBox class="contactTxtBox" 
ID="emailBox" runat="server" />
                                        <asp:RequiredFieldValidator 
ID="emailReqVal" ControlToValidate="emailBox"
                                            ErrorMessage="Enter Email" 
CssClass="field-validation-error" runat="server">*</asp:RequiredFieldValidator>
                                        <asp:RegularExpressionValidator 
ID="emailRegVal"
                                            
                                            ValidationExpression="\w+([-
+.]\w+)*@\w+([-.]\w+)*\.\w+([-.]\w+)*"
                                            ControlToValidate="emailBox"
                                            ErrorMessage="Enter Valid Email 
Address"  CssClass="field-validation-error" 
                                            
runat="server">*</asp:RegularExpressionValidator>
                                    </td>
                                </tr>
                                <tr>
                                    <td>Phone:
                                    </td>
                                    <td class="contactPad">
                                        <asp:TextBox class="contactTxtBox" 
ID="phoneBox" runat="server" />
                                    </td>
                                </tr>
                            </table>
                        </fieldset>
                    </td>
                </tr>
                <tr>
                    <td>

                        <fieldset>
                            <legend>Message</legend>
                            <table >
                                <tr>
                                    <td align="center">Subject:
                                    </td>
                                    <td class="contactPad">
                                        <asp:TextBox Width="248" ID="subjectBox" 
runat="server" />
                                        <asp:RequiredFieldValidator 
ID="subjectReqVal" ControlToValidate="subjectBox"
                                            ErrorMessage="Enter Subject" 
CssClass="field-validation-error" runat="server">*</asp:RequiredFieldValidator>
                                    </td>
                                </tr>
                                <tr>
                                    <td colspan="2" " >
                                        <asp:TextBox width="311" ID="messageBox"
TextMode="MultiLine"
                                            Rows="6" runat="server"></asp:TextBox>
                                        <asp:RequiredFieldValidator
ID="messageReqVal" ControlToValidate="messageBox"
                                            ErrorMessage="Enter Message"
CssClass="field-validation-error" runat="server">*</asp:RequiredFieldValidator>
                                    </td>
                                </tr>
                            </table>
                        </fieldset>
                    </td>
                </tr>
                <tr>
                    <td>
                        <asp:Button ID="submitBtn" Text="Submit" Width="80"
runat="server" OnClick="SubmitBtn_Click" />
                        <asp:Button ID="resetBtn" Text="Reset" Width="60"
CausesValidation="false"
                            OnClick="ResetBtn_Click" runat="server" />
                    </td>
                </tr>
                <tr>
                    <td>
                        <asp:ValidationSummary CssClass="field-validation-error"
ID="valSum" runat="server" />
                    </td>
                </tr>
            </table>
        </div>
    </article>
    </asp:Panel>
    <div id="contactMSG">
    <asp:Label class="contactMsg" ID="lblMsg" runat="server" Text=""></asp:Label>
        </div>
</asp:Content>
```

### 10.1.2 Contact.aspx.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Data;
using System.Configuration;
using System.Collections;
using System.Net.Mail;
using System.Web.Configuration;
using System.Web.Security;
using System.Web.UI.WebControls.WebParts;
using System.Web.UI.HtmlControls;

namespace Griffin_Drywall
{
    public partial class Contact : Page
    {

        protected void Page_Load(object sender, EventArgs e)
        {
            if (!Page.IsPostBack)
            {

            }
        }

        protected void SubmitBtn_Click(object sender, EventArgs e)
        {
            if (Page.IsValid)
            {
                System.Configuration.Configuration config
                    =
WebConfigurationManager.OpenWebConfiguration(base.Request.ApplicationPath);
                AppSettingsSection appSettings =
(AppSettingsSection)config.GetSection("appSettings");

                string emailHost = appSettings.Settings["EmailHost"].Value;
                string fromAddress = appSettings.Settings["FromEmailAddr"].Value;
                string toAddress = appSettings.Settings["ToEmailAddr"].Value;

                SmtpClient smtpClient = new SmtpClient();
                smtpClient.EnableSsl = true;

                // Default in IIS will be localhost
                //smtpClient.Host = "localhost";

                //Default port will be 25
                //smtpClient.Port = 578;

                MailMessage message = new MailMessage();
                message.IsBodyHtml = false;
                message.Priority = MailPriority.High;
                message.DeliveryNotificationOptions =
DeliveryNotificationOptions.OnFailure;

                try
                {
                    message.Subject = "Griffin Drywall: " + this.subjectBox.Text;
                    message.Body = "Sender: " + nameBox.Text.Trim() + "\n";
                    message.Body += "Email: " + emailBox.Text.Trim() + "\n";
                    message.Body += "Phone: " + phoneBox.Text.Trim() + "\n";
                    message.Body += "\n\n" + this.messageBox.Text + "\n\n";

                    smtpClient.Send(fromAddress, toAddress, message.Subject,
message.Body);
                }
                catch (Exception ex)
                {
                    // Display error panel
                    lblMsg.Text = "We seem to be experiencing some problems with
our contact form. We apologize for this inconvenience and suggest that you try
again later.";
```

```
                    // Log error
                    LogError(ex);
                }
                emailPanel.Visible = false;
                lblMsg.Text = "Thank you for your inquiry. We will get back to you 
as soon as possible.";
            }
        }

        protected void ResetBtn_Click(object sender, EventArgs e)
        {
            nameBox.Text = "";
            phoneBox.Text = "";
            emailBox.Text = "";
            subjectBox.Text = "";
            messageBox.Text = "";
        }

        private void LogError(Exception error)
        {
            LogError(error.Message, error.StackTrace);
        }


        private void LogError(string errMsg, string errTrace)
        {
            // Get programatic access to the email information stored in the 
web.config file
            Configuration config = 
WebConfigurationManager.OpenWebConfiguration(Request.ApplicationPath);
            AppSettingsSection appSettings = 
(AppSettingsSection)config.GetSection("appSettings");

            // Extract the values based on their key names
            string toAddress = appSettings.Settings["MyEmailAddr"].Value;
            string fromAddress = appSettings.Settings["ToEmailAddr"].Value;
            string emailHost = appSettings.Settings["EmailHost"].Value;
            string mailSubject = "Error in Customer Site: Contact.aspx.cs";
            string mailBody = errMsg + errTrace;

            // Create error notification email
            SmtpClient smtpClient = new SmtpClient(emailHost);
            MailMessage message = new MailMessage(fromAddress, toAddress, 
mailSubject, mailBody);
            message.IsBodyHtml = false;
            smtpClient.Send(message);
        }
    }
}
```

### 10.1.3 Default.aspx

```
<%@ Page Title="Griffin Drywall | Home" Language="C#" 
MasterPageFile="~/Site.Master" AutoEventWireup="true" CodeBehind="Default.aspx.cs" 
Inherits="Griffin_Drywall._Default" %>

<asp:Content runat="server" ID="FeaturedContent" 
ContentPlaceHolderID="FeaturedContent">
    <section class="featured">
```

```
        <div class="content-wrapper">

        </div>
    </section>
</asp:Content>
<asp:Content runat="server" ID="BodyContent" ContentPlaceHolderID="MainContent">

    <div id="left">
   <article>
       <h1>About</h1>
        <p>
           Griffin Drywall & Insulation, Inc. is a full service drywall company, 
equipped to handle your 
             residential and commerical drywall needs.  At Griffin Drywall & 
Insulation, Inc., our goal is to supply our costomers 
            with the best quality products and performace while remaining 
competitive in today's market place.
        </p>

        <p>
           While centrally located in Plumsteadville, PA, Griffin Drywall 
currently services Pennsylvania and New Jersey communities 
            as well as the building industry since 1980.  Our friendly office 
staff is readily at your service!
        </p>
   </article>
        <object id="griffin-xml" class="round-img" width="325" height="300" 
name="griffin-xml" classid="clsid:D27CDB6E-AE6D-11cf-96B8-444553540000" 
style="visibility: visible;">
            <param value="griffin-xml.swf" name="movie">
            <param value="high" name="quality">
            <param value="#ffffff" name="bgcolor">
            <param value="true" name="play">
            <param value="true" name="loop">
            <param value="transparent" name="wmode">
            <param value="showall" name="scale">
            <param value="true" name="menu">
            <param value="false" name="devicefont">
            <param value="" name="salign">
            <param value="sameDomain" name="allowScriptAccess">
            <object width="325"  height="232" class="round-img2" 
data="/Flash/griffin-xml.swf" type="application/x-shockwave-flash">
                <param value="true" name="allowfullscreen">
                <param value="transparent" name="wmode">
                <param value="high" name="quality">
                <param value="#ffffff" name="bgcolor">
                <param value="true" name="play">
                <param value="true" name="loop">
            </object>
        </object>
    </div>
    <div style="clear: both"></div>
    <div id="request-btn">
        <a id="A6" href="<%: GetRouteUrl("RequestBidRoute", null) %>" 
onmouseover="document.rollover6.src=requestOver.src" 
onmouseout="document.rollover6.src=request.src">
            <img id="Img2" runat="server" src="/Images/btnRequest.png" border="0" 
name="rollover6"></a>
```

```
    </div>
</asp:Content>
```

### 10.1.4 Default.aspx.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;

namespace Griffin_Drywall
{
    public partial class _Default : Page
    {
        protected void Page_Load(object sender, EventArgs e)
        {

        }
    }
}
```

### 10.1.5 Global.asax

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.Optimization;
using System.Web.Routing;
using System.Web.Security;
using Griffin_Drywall;
using System.Data.Entity;
using Griffin_Drywall.Models;

namespace Griffin_Drywall
{
    public class Global : HttpApplication
    {
        void Application_Start(object sender, EventArgs e)
        {
            // Code that runs on application startup
            BundleConfig.RegisterBundles(BundleTable.Bundles);
            AuthConfig.RegisterOpenAuth();

            // Add Administrator.
            if (!Roles.RoleExists("Administrator"))
            {
                Roles.CreateRole("Administrator");
            }
            if (!Roles.RoleExists("SubContractor"))
            {
                Roles.CreateRole("SubContractor");
            }
            if (Membership.GetUser("Admin") == null)
            {
                Membership.CreateUser("Admin", "Pa$$word", "Admin@griffindw.com");
                Roles.AddUserToRole("Admin", "Administrator");
            }

            // Add Routes
            RegisterRoutes(RouteTable.Routes);

        }

        void RegisterRoutes(RouteCollection routes)
        {
            routes.MapPageRoute(
                "HomeRoute",
                "Home",
                "~/Default.aspx"
            );

            routes.MapPageRoute(
                "ServicesRoute",
                "Services",
                "~/Services.aspx"
            );

            routes.MapPageRoute(
                "ProjectsRoute",
                "Projects",
                "~/Projects.aspx"
            );

            routes.MapPageRoute(
                "ContactRoute",
                "Contact",
                "~/Contact.aspx"
            );

            routes.MapPageRoute(
                "LoginRoute",
                "Login",
                "~/Account/Login.aspx"
            );

            routes.MapPageRoute(
                "RegisterRoute",
                "Register",
                "~/Account/Register.aspx"
            );

            routes.MapPageRoute(
                "ManageRoute",
                "Manage",
                "~/Account/Manage.aspx"
            );

            routes.MapPageRoute(
                "AddUserRoute",
                "AddUser",
                "~/Admin/AddUser.aspx"
            );

            routes.MapPageRoute(
                "RegisterExternalRoute",
                "RegesterExternalLogin",
                "~/Account/RegisterExternalLogin.aspx"
            );

            routes.MapPageRoute(
                "AdminRoute",
                "Admin",
                "~/Admin/AdminPage.aspx"
            );

            routes.MapPageRoute(
                "MgnUsersRoute",
                "MgnUsers",
                "~/Admin/MgnUsers.aspx"
            );

            routes.MapPageRoute(
                "EditServiceRoute",
                "MgnServices",
                "~/Admin/EditService.aspx"
            );

            routes.MapPageRoute(
                "EditServiceTypeRoute",
                "MgnServiceTypes",
                "~/Admin/EditServiceType.aspx"
            );

            routes.MapPageRoute(
                "EditProjectRoute",
                "MgnProjects",
                "~/Admin/EditProjects.aspx"
            );

            routes.MapPageRoute(
                "EditPGalleryRoute",
                "MgnGalleries",
                "~/Admin/EditPGallery.aspx"
            );

            routes.MapPageRoute(
                "EditJobRoute",
                "MgnJobs",
                "~/Admin/EditJob.aspx"
            );

            routes.MapPageRoute(
                "EditSubConRoute",
                "MgnSubCons",
                "~/Admin/EditSubCon.aspx"
            );

            routes.MapPageRoute(
                "EditWorkRoute",
                "MgnWork",
                "~/Admin/EditWork.aspx"
            );
            routes.MapPageRoute(
                "ViewWorkRoute",
                "ViewWork",
                "~/Admin/ViewWork.aspx"
            );

            routes.MapPageRoute(
                "ServicesByNameRoute",
                "Services/{serviceName}",
                "~/Services.aspx"
            );

            routes.MapPageRoute(
                "ProjectsByNameRoute",
                "Projects/{projectName}",
                "~/ProjectDetails.aspx"
            );

            routes.MapPageRoute(
                "PrevReqRoute",
                "PrevReq",
                "~/PrevReq.aspx"
            );

            routes.MapPageRoute(
                "RequestBidRoute",
                "RequestBid",
                "~/RequestBid.aspx"
            );
        }

        void Application_End(object sender, EventArgs e)
        {
            //  Code that runs on application shutdown

        }

        void Application_Error(object sender, EventArgs e)
        {
            // Code that runs when an unhandled error occurs

        }
    }
}
```

### 10.1.6 PrevReq.aspx

```
<%@ Page Title="" Language="C#" MasterPageFile="~/Site.Master"
AutoEventWireup="true" CodeBehind="PrevReq.aspx.cs"
Inherits="Griffin_Drywall.PrevReq" %>
<asp:Content ID="Content1" ContentPlaceHolderID="HeadContent" runat="server">
</asp:Content>
<asp:Content ID="Content2" ContentPlaceHolderID="FeaturedContent" runat="server">
</asp:Content>
<asp:Content ID="Content3" ContentPlaceHolderID="MainContent" runat="server">
    <br />
    <div style="vertical-align: top; left: auto; overflow: auto; width: auto;">
        <asp:GridView ID="GridView1" runat="server" AutoGenerateColumns="false"
```

```
            CssClass="mGrid" AlternatingRowStyle-CssClass="alt"
            PagerStyle-CssClass="pgr">
            <Columns>
                <asp:TemplateField>
                    <HeaderTemplate>
                        <asp:Label ID="Label4" runat="server"
Text="Name"></asp:Label>
                    </HeaderTemplate>
                    <ItemTemplate>
                        <asp:Label ID="Label1" runat="server" Text='<%#
Eval("RequestorName") %>'></asp:Label>
                    </ItemTemplate>
                </asp:TemplateField>
                <asp:TemplateField>
                    <HeaderTemplate>
                        <asp:Label ID="Label6" runat="server"
Text="Email"></asp:Label>
                    </HeaderTemplate>
                    <ItemTemplate>
                        <asp:Label ID="Label2" runat="server" Text='<%#
Eval("RequestorEmail") %>'></asp:Label>
                    </ItemTemplate>
                </asp:TemplateField>
                <asp:TemplateField>
                    <HeaderTemplate>
                        <asp:Label ID="Label7" runat="server"
Text="Phone"></asp:Label>
                    </HeaderTemplate>
                    <ItemTemplate>
                        <asp:Label ID="Label3" runat="server" Text='<%#
Eval("RequestorPhone") %>'></asp:Label>
                    </ItemTemplate>
                </asp:TemplateField>
                <asp:TemplateField>
                    <HeaderTemplate>
                        <asp:Label ID="Label12" runat="server"
Text="Type"></asp:Label>
                    </HeaderTemplate>
                    <ItemTemplate>
                        <asp:Label ID="Label13" runat="server" Text='<%#
Eval("Types") %>'></asp:Label>
                    </ItemTemplate>
                </asp:TemplateField>
                <asp:TemplateField>
                    <HeaderTemplate>
                        <asp:Label ID="Label10" runat="server"
Text="Services"></asp:Label>
                    </HeaderTemplate>
                    <ItemTemplate>
                        <asp:Label ID="Label11" runat="server" Text='<%#
Eval("Services") %>'></asp:Label>
                    </ItemTemplate>
                </asp:TemplateField>
                <asp:TemplateField>
                    <HeaderTemplate>
                        <asp:Label ID="Label8" runat="server"
Text="Description"></asp:Label>
                    </HeaderTemplate>
                    <ItemTemplate>
                        <div style="height: 65px; width: 370px; overflow: 
auto;"><%# Eval("Description") %></div>
                    </ItemTemplate>
                </asp:TemplateField>
                <asp:TemplateField>
                    <HeaderTemplate>
                        <asp:Label ID="Label9" runat="server" Text="Date 
Created"></asp:Label>
                    </HeaderTemplate>
                    <ItemTemplate>
                        <asp:Label ID="Label5" runat="server" Text='<%# 
Eval("DateCreated") %>'></asp:Label>
                    </ItemTemplate>
                </asp:TemplateField>
            </Columns>
            <EmptyDataTemplate>
                <div>There are no requests for this user.  Please ensure that you 
are logged in or register with the site. Thank you.</div>
            </EmptyDataTemplate>
        </asp:GridView>
    </div>
</asp:Content>
```

### 10.1.7 PrevReq.aspx.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
using Griffin_Drywall.Models;
using System.Web.ModelBinding;

namespace Griffin_Drywall
{
    public partial class PrevReq : System.Web.UI.Page
    {
        public string userId;

        protected void Page_Load(object sender, EventArgs e)
        {
            Griffin_Drywall.RequestBid users = new Griffin_Drywall.RequestBid();
            userId = users.GetUserId();

            using (var db = new Griffin_Drywall.Models.GriffinContext())
            {
                var query = from r in 
db.Requests.Include("Services").Include("ServiceTypes")
                            where r.UserId == userId
                            select new
                            {
                                r.RequestorName,
                                r.RequestorEmail,
                                r.RequestorPhone,
                                Types = (from t in r.ServiceTypes select 
t.Type).DefaultIfEmpty(),
                                Services = (from s in r.Services select
s.ServiceName).DefaultIfEmpty(),
                                r.Description,
                                r.DateCreated
                            };
                var listed = query.ToList();
                var commaListed = listed.Select(a => new
                {
                    a.RequestorName,
                    a.RequestorEmail,
                    a.RequestorPhone,
                    Types = a.Types.Aggregate((s1, s2) => s1 + ", " + s2),
                    Services = a.Services.Aggregate((s1, s2) => s1 + ", " + s2),
                    a.Description,
                    a.DateCreated
                });
                GridView1.DataSource = commaListed.ToList();
                GridView1.DataBind();
            }
        }

        protected override void OnPreRender(EventArgs e)
        {
            base.OnPreRender(e);

            if (GridView1.Rows.Count == 0)
                GridView1.CssClass = "empty";
        }
    }
}
```

### 10.1.8 ProjectDetails.aspx

```
<%@ Page Title="" Language="C#" MasterPageFile="~/Site.Master"
AutoEventWireup="true" CodeBehind="ProjectDetails.aspx.cs"
Inherits="Griffin_Drywall.ProjectDetails" %>
<asp:Content ID="Content1" ContentPlaceHolderID="HeadContent" runat="server">
</asp:Content>
<asp:Content ID="Content2" ContentPlaceHolderID="FeaturedContent" runat="server">
</asp:Content>
<asp:Content ID="Content3" ContentPlaceHolderID="MainContent" runat="server">
    <section id="projectDetailsForm">
        <asp:FormView ID="projectDetails" runat="server"
ItemType="Griffin_Drywall.Models.Project" SelectMethod="GetProjects"
RenderOuterTable="false">
            <ItemTemplate>
                <article>
                <div>
                    <h1><%#:Item.ProjectName%></h1>
                </div>

                <table>
                    <tr>
                        <td style="vertical-align: top">
                            <p style="margin:0">
                            <%#:Item.Description%>
                            </p>
                        </td>
                    </tr>
                </table>
                </article>
            </ItemTemplate>
        </asp:FormView>
    </section>
    <asp:ListView runat="server" ID="lvw" DataKeyNames="GImageID"
ItemType="Griffin_Drywall.Models.GImage" SelectMethod="GetGImages">
        <LayoutTemplate>
            <div id="galleria">
                <asp:PlaceHolder ID="itemPlaceholder"
runat="server"></asp:PlaceHolder>
            </div>
        </LayoutTemplate>
        <ItemTemplate>
            <img alt="" src="/Images/Projects/PGallery/<%#:Item.ImagePath %>"
                id="" class="full" />
        </ItemTemplate>
    </asp:ListView>
        <script>

Galleria.loadTheme('../Scripts/themes/classic/galleria.classic.min.js');
            Galleria.run('#galleria');
        </script>
</asp:Content>
```

### 10.1.9 ProjectDetails.aspx.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
using Griffin_Drywall.Models;
using System.Web.ModelBinding;

namespace Griffin_Drywall
{
    public partial class ProjectDetails : System.Web.UI.Page
    {
        protected void Page_Load(object sender, EventArgs e)
        {

        }

        public IQueryable<Project> GetProjects(
                                [QueryString("id")] int? projectId,
                                [RouteData] string projectName)
        {
            var _db = new Griffin_Drywall.Models.GriffinContext();
            IQueryable<Project> query = _db.Projects;

            if (projectId.HasValue && projectId > 0)
            {
                query = query.Where(p => p.ProjectID == projectId);
            }

            if (!String.IsNullOrEmpty(projectName))
            {
                query = query.Where(p =>
                                    String.Compare(p.ProjectName,
                                    projectName) == 0);
            }
            return query;
        }

        public IQueryable<GImage> GetGImages(
                                [QueryString("id")] int? projectId,
                                [RouteData] string projectName)
        {
            var _db = new Griffin_Drywall.Models.GriffinContext();
            IQueryable<GImage> query = _db.GImages;

            if (projectId.HasValue && projectId > 0)
            {
                query = query.Where(g => g.Project.ProjectID == projectId);
            }

            if (!String.IsNullOrEmpty(projectName))
            {
                query = query.Where(g =>
                                    String.Compare(g.Project.ProjectName,
                                    projectName) == 0);
            }
            return query;
        }
    }
}
```

### 10.1.10 Projects.aspx

```
<%@ Page Title="Griffin Drywall | Projects" Language="C#"
MasterPageFile="~/Site.Master" AutoEventWireup="true"
CodeBehind="Projects.aspx.cs" Inherits="Griffin_Drywall.Projects" %>
<asp:Content ID="Content1" ContentPlaceHolderID="HeadContent" runat="server">
</asp:Content>
<asp:Content ID="Content2" ContentPlaceHolderID="FeaturedContent" runat="server">
</asp:Content>
<asp:Content ID="Content3" ContentPlaceHolderID="MainContent" runat="server">
    <article class="projects">
       <h1>Projects</h1>
        <p>
           Griffin Drywall & Insulation, Inc. has serviced the Drywall industry
for years, check out some of our current projects!
        </p>
   </article>
    <asp:ListView ID="projectList" runat="server"
                            DataKeyNames="ProjectID"
                            GroupItemCount="3"
ItemType="Griffin_Drywall.Models.Project" SelectMethod="GetProjects">
                            <EmptyDataTemplate>
                                <table id="Table1" runat="server">
                                    <tr>
                                        <td>No data was returned.</td>
                                    </tr>
                                </table>
                            </EmptyDataTemplate>
                            <EmptyItemTemplate>
```

```
                                    <td id="Td1" runat="server" />
                                </EmptyItemTemplate>
                                <GroupTemplate>
                                    <tr ID="itemPlaceholderContainer" runat="server">
                                        <td ID="itemPlaceholder" runat="server"></td>
                                    </tr>
                                </GroupTemplate>
                                <ItemTemplate>
                                    <td id="Td2" runat="server">
                                        <table class="tbProjects">
                                            <tr>
                                                <td>
                                                    <a href="<%#:
GetRouteUrl("ProjectsByNameRoute", new {projectName = Item.ProjectName}) %>">
                                                        <image
src='/Images/Projects/Thumbs/<%#:Item.ImagePath%>'
                                                            width="200" height="175"
border="1" />
                                                    </a>
                                                </td>
                                            </tr>
                                            <tr >
                                                <td class="tbPad">
                                                    <a href="<%#:
GetRouteUrl("ProjectsByNameRoute", new {projectName = Item.ProjectName}) %>">
                                                        <%#:Item.ProjectName%>
                                                    </a>
                                                </td>
                                                </tr>
                                        </table>
                                    </td>
                                </ItemTemplate>
                                <LayoutTemplate>
                                    <table id="Table2" runat="server">
                                        <tr id="Tr1" runat="server">
                                            <td id="Td3" runat="server">
                                                <table ID="groupPlaceholderContainer"
runat="server">

                                                    <tr ID="groupPlaceholder"

runat="server"></tr>

                                                </table>
                                            </td>
                                        </tr>
                                        <tr id="Tr2" runat="server"><td id="Td4"
runat="server"></td></tr>
                                    </table>
                                </LayoutTemplate>
                            </asp:ListView>
</asp:Content>
```

### 10.1.11 Projects.aspx.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
using Griffin_Drywall.Models;
using System.Web.ModelBinding;

namespace Griffin_Drywall
{
    public partial class Projects : System.Web.UI.Page
    {
        protected void Page_Load(object sender, EventArgs e)
        {

        }
        public IQueryable<Project> GetProjects(
                                [QueryString("id")] int? projectId,
                                [RouteData] string projectName)
        {
            var _db = new Griffin_Drywall.Models.GriffinContext();
            IQueryable<Project> query = _db.Projects;

            if (projectId.HasValue && projectId > 0)
            {
                query = query.Where(p => p.ProjectID == projectId);
            }

            if (!String.IsNullOrEmpty(projectName))
            {
                query = query.Where(p =>
                                    String.Compare(p.ProjectName,
                                    projectName) == 0);
            }
            return query;
        }

    }
}
```

### 10.1.12 RequestBid.aspx

```
<%@ Page Title="" Language="C#" MasterPageFile="~/Site.Master"
AutoEventWireup="true" CodeBehind="RequestBid.aspx.cs"
Inherits="Griffin_Drywall.RequestBid" %>
<asp:Content ID="Content1" ContentPlaceHolderID="HeadContent" runat="server">
</asp:Content>
<asp:Content ID="Content2" ContentPlaceHolderID="FeaturedContent" runat="server">
</asp:Content>
<asp:Content ID="Content3" ContentPlaceHolderID="MainContent" runat="server">
    <article class="reqBid">
       <h1>Request a Bid</h1>
        <p>
           Fill out the form below and Griffin Drywall & Insulation, Inc. will
promptly get back to you with our estimates of the services required.
        </p>
   </article>
   <asp:Panel ID="emailPanel" runat="server">
    <article id="requestBidEmail">
        <div class="requestBidEmail">
            <table id="contactDetails">
                <tr>
                    <td>
                        <fieldset>
                            <legend>Contact Details</legend>
                            <table>
                                <tr>

                                    <td>Name:
                                    </td>
                                    <td class="requestPad">
                                        <asp:TextBox class="requestTxtBox"
ID="nameBox" runat="server" />
                                         <asp:RequiredFieldValidator
ID="nameReqVal" ControlToValidate="nameBox"
                                            ErrorMessage="Enter Name"
CssClass="field-validation-error" runat="server">*</asp:RequiredFieldValidator>
                                    </td>
                                </tr>
                                <tr>
                                    <td >Email:
                                    </td>
                                    <td class="requestPad">
                                        <asp:TextBox class="requestTxtBox"
ID="emailBox" runat="server" />
                                        <asp:RequiredFieldValidator
ID="emailReqVal" ControlToValidate="emailBox"
                                            ErrorMessage="Enter Email"
CssClass="field-validation-error" runat="server">*</asp:RequiredFieldValidator>
                                        <asp:RegularExpressionValidator
ID="emailRegVal"
                                            ValidationExpression="\w+([-
+.]\w+)*@\w+([-.]\w+)*\.\w+([-.]\w+)*"
                                            ControlToValidate="emailBox"
                                            ErrorMessage="Enter Valid Email
Address"  CssClass="field-validation-error"

runat="server">*</asp:RegularExpressionValidator>
                                    </td>
                                </tr>
                                <tr>
                                    <td>Phone:
                                    </td>
                                    <td class="requestPad">
                                        <asp:TextBox class="requestTxtBox"
ID="phoneBox" runat="server" />
                                    </td>
                                </tr>
                            </table>
                        </fieldset>
                    </td>
                </tr>
                <tr>
                    <td><a href="<%= GetRouteUrl("PrevReqRoute", null) %>">View
Previous Requests</a></td>
                    <td>
                        <asp:ValidationSummary CssClass="field-validation-error"
ID="valSum" runat="server" />
                    </td>
                </tr>
                </table>
               <table id="descWork">
                <tr>
```

```
                    <td>

                        <fieldset>
                            <legend>Description of Work</legend>
                            <table >
                                <tr>
                                    <td>
                                        <fieldset class="serviceType">
                                        <legend>Type</legend>
                                        <asp:CheckBoxList ID="ServiceType"
class="serviceChkBox" runat="server"></asp:CheckBoxList>
                                        </fieldset>
                                    </td>
                                    <td>
                                        <fieldset>
                                        <legend>Services</legend>
                                <asp:CheckBoxList ID="Services"
class="serviceChkBox" runat="server" ></asp:CheckBoxList>
                                            </fieldset>
                                        </td>
                                    </tr>
                                <tr>
                                    <td>Description:
                                    </td>
                                </tr>
                                <tr>
                                    <td colspan="2" " >
                                        <asp:TextBox width="400px" ID="messageBox"
TextMode="MultiLine"
                                            Rows="6" runat="server"></asp:TextBox>
                                        <asp:RequiredFieldValidator
ID="messageReqVal" ControlToValidate="messageBox"
                                            ErrorMessage="Enter Description"
CssClass="field-validation-error" runat="server">*</asp:RequiredFieldValidator>
                                    </td>
                                </tr>
                            </table>
                        </fieldset>
                    </td>
                </tr>
                <tr>
                    <td>
                        <asp:Button ID="submitBtn" Text="Submit" Width="80"
runat="server" OnClick="SubmitBtn_Click" />
                        <asp:Button ID="resetBtn" Text="Reset" Width="60"
CausesValidation="false"
                            OnClick="ResetBtn_Click" runat="server" />
                    </td>
                </tr>
            </table>
        </div>
    </article>
    </asp:Panel>
    <div id="requestMSG">
    <asp:Label class="requestMsg" ID="lblMsg" runat="server" Text=""></asp:Label>
        </div>
</asp:Content>
```

### 10.1.13 RequestBid.aspx.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Diagnostics;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Data;
using System.Data.Entity.Validation;
using System.Configuration;
using System.Collections;
using System.Net.Mail;
using System.Web.Configuration;
using System.Web.Security;
using System.Web.UI.WebControls.WebParts;
using System.Web.UI.HtmlControls;
using Griffin_Drywall.Models;
using System.Web.ModelBinding;


namespace Griffin_Drywall
{
    public partial class RequestBid : System.Web.UI.Page
    {

        public const string UserSessionKey = "UserId";

        private GriffinContext db = new GriffinContext();

        protected void Page_Load(object sender, EventArgs e)
        {
            if (!IsPostBack)
            {
                BindServices();
                BindServiceTypes();
            }
        }

        private void Page_PreRender()
        {

        }

        protected void SubmitBtn_Click(object sender, EventArgs e)
        {
            if (Page.IsValid)
            {
                System.Configuration.Configuration config
                    =
WebConfigurationManager.OpenWebConfiguration(base.Request.ApplicationPath);
                AppSettingsSection appSettings =
(AppSettingsSection)config.GetSection("appSettings");

                string emailHost = appSettings.Settings["EmailHost"].Value;
                string fromAddress = appSettings.Settings["FromEmailAddr"].Value;
                string toAddress = appSettings.Settings["ToEmailAddr"].Value;
                SmtpClient smtpClient = new SmtpClient();
                smtpClient.EnableSsl = true;

                // Default in IIS will be localhost
                //smtpClient.Host = "localhost";

                //Default port will be 25
                //smtpClient.Port = 578;

                MailMessage message = new MailMessage();
                message.IsBodyHtml = false;
                message.Priority = MailPriority.High;
                message.DeliveryNotificationOptions =
DeliveryNotificationOptions.OnFailure;

                var selectedServices = Services.Items.Cast<ListItem>().Where(x =>
x.Selected);
                var selectedServicesTypes =
ServiceType.Items.Cast<ListItem>().Where(x => x.Selected);

                try
                {
                    var id = Guid.NewGuid().ToString();
                    var requestItem = new Request
                    {
                        RequestId = id,
                        UserId = GetUserId(),
                        RequestorName = nameBox.Text.Trim(),
                        RequestorEmail = emailBox.Text.Trim(),
                        RequestorPhone = phoneBox.Text.Trim(),
                        Description = this.messageBox.Text,
                        DateCreated = DateTime.Now
                    };

                    db.Requests.Add(requestItem);

                    foreach (ListItem service in selectedServices)
                    {
                        var snum = Convert.ToInt16(service.Value);
                        var ser = new Service() { ServiceID = snum };
                        db.Services.Attach(ser);
                        requestItem.Services.Add(ser);
                    }

                    foreach (ListItem type in selectedServicesTypes)
                    {
                        var tnum = Convert.ToInt16(type.Value);
                        var typ = new ServiceType() { ServiceTypeID = tnum };
                        db.ServiceTypes.Attach(typ);
                        requestItem.ServiceTypes.Add(typ);
                    }

                    db.SaveChanges();

                }
                catch (DbEntityValidationException dbEx)
                {
                    foreach (var validationErrors in dbEx.EntityValidationErrors)
                    {
                        foreach (var validationError in 
validationErrors.ValidationErrors)
                        {
                            System.Diagnostics.Trace.TraceInformation("Property: 
{0} Error: {1}", validationError.PropertyName, validationError.ErrorMessage);
                        }
                    }
                }


                try
                {
                    message.Subject = "Griffin Drywall: Request for Bid";
                    message.Body = "Sender: " + nameBox.Text.Trim() + "\n";
                    message.Body += "Email: " + emailBox.Text.Trim() + "\n";
                    message.Body += "Phone: " + phoneBox.Text.Trim() + "\n";
                    message.Body += "Type: ";
                    foreach (ListItem type in selectedServicesTypes)
                    {
                        message.Body += type.Text + ", ";
                    }
                    message.Body += "\n";
                    message.Body += "Services: ";
                    foreach (ListItem service in selectedServices)
                    {
                        message.Body += service.Text + ", ";
                    }
                    message.Body += "\n";
                    message.Body += "\n\n" + this.messageBox.Text + "\n\n";

                    smtpClient.Send(fromAddress, toAddress, message.Subject, 
message.Body);
                }
                catch (Exception ex)
                {
                    // Display error panel
                    lblMsg.Text = "We seem to be experiencing some problems with 
our contact form. We apologize for this inconvenience and suggest that you try 
again later.";

                    // Log error
                    LogError(ex);
                }
                emailPanel.Visible = false;
                lblMsg.Text = "Thank you for your inquiry. We will get back to you 
as soon as possible.";
            }
        }

        protected void ResetBtn_Click(object sender, EventArgs e)
        {
            nameBox.Text = "";
            phoneBox.Text = "";
            emailBox.Text = "";
            messageBox.Text = "";
            Services.ClearSelection();
            ServiceType.ClearSelection();
        }

        protected void BindServices()
        {
            var services = from s in db.Services
                           select new { s.ServiceID, s.ServiceName };
            Services.DataSource = services.ToList();
            Services.DataTextField = "ServiceName";
            Services.DataValueField = "ServiceID";
            Services.DataBind();
        }

        protected void BindServiceTypes()
        {
            var servicesTypes = from s in db.ServiceTypes
                           select new { s.ServiceTypeID, s.Type };
            ServiceType.DataSource = servicesTypes.ToList();
            ServiceType.DataTextField = "Type";
            ServiceType.DataValueField = "ServiceTypeID";
            ServiceType.DataBind();
        }

        public string GetUserId()
        {
            if (HttpContext.Current.Session[UserSessionKey] == null)
            {
                if
(!string.IsNullOrWhiteSpace(HttpContext.Current.User.Identity.Name))
                {
                    HttpContext.Current.Session[UserSessionKey] =
HttpContext.Current.User.Identity.Name;
                }
                else
                {
                    // Generate a new random GUID using System.Guid class.
                    Guid tempUserId = Guid.NewGuid();
                    HttpContext.Current.Session[UserSessionKey] =
tempUserId.ToString();
                }
            }
            return HttpContext.Current.Session[UserSessionKey].ToString();
        }

        public void MigrateUser(string userId, string userName)
        {
            var user = db.Requests.Where(c => c.UserId == userId);
            foreach (Request item in user)
            {
                item.UserId = userName;
            }
            HttpContext.Current.Session[UserSessionKey] = userName;
            db.SaveChanges();
        }

        private void LogError(Exception error)
        {
            LogError(error.Message, error.StackTrace);
        }


        private void LogError(string errMsg, string errTrace)
        {
            // Get programatic access to the email information stored in the
web.config file
            Configuration config =
WebConfigurationManager.OpenWebConfiguration(Request.ApplicationPath);
            AppSettingsSection appSettings =
(AppSettingsSection)config.GetSection("appSettings");

            // Extract the values based on their key names
            string toAddress = appSettings.Settings["MyEmailAddr"].Value;
            string fromAddress = appSettings.Settings["ToEmailAddr"].Value;
            string emailHost = appSettings.Settings["EmailHost"].Value;
            string mailSubject = "Error in Customer Site: Contact.aspx.cs";
            string mailBody = errMsg + errTrace;

            // Create error notification email
            SmtpClient smtpClient = new SmtpClient(emailHost);
            MailMessage message = new MailMessage(fromAddress, toAddress,
mailSubject, mailBody);
            message.IsBodyHtml = false;
            smtpClient.Send(message);
        }
    }
}
```

### 10.1.14 Services.aspx

```
<%@ Page Title="Griffin Drywall | Services" Language="C#"
MasterPageFile="~/Site.Master" AutoEventWireup="true"
CodeBehind="Services.aspx.cs" Inherits="Griffin_Drywall.Services" %>
<asp:Content ID="Content1" ContentPlaceHolderID="HeadContent" runat="server">
</asp:Content>
<asp:Content ID="Content2" ContentPlaceHolderID="FeaturedContent" runat="server">
</asp:Content>
<asp:Content ID="Content3" ContentPlaceHolderID="MainContent" runat="server">
    <section class="serviceMenu" style="text-align: center;">
                <asp:ListView ID="serviceMenu"
                    ItemType="Griffin_Drywall.Models.Service"
                    runat="server"
                    SelectMethod="GetServices" >
                    <ItemTemplate>
                        <b style="font-size: large; font-style: normal">
                        <a href="<%#: GetRouteUrl("ServicesByNameRoute", new
{serviceName = Item.ServiceName}) %>">
                        <%#: Item.ServiceName %>
                        </a>
                        </b>
                    </ItemTemplate>
                    <ItemSeparatorTemplate> - </ItemSeparatorTemplate>
                </asp:ListView>
            </section>
     <asp:FormView ID="serviceDetails" CssClass="serviceDetails" runat="server"
ItemType="Griffin_Drywall.Models.Service" SelectMethod ="GetServiceById" >
        <ItemTemplate>
            <table>
```

```
                <tr>
                    <td style="vertical-align: top">
                        <div id="serviceDescription">
                            <h1><%#:Item.ServiceName %></h1>
                            <p><%#:Item.Description %></p>
                            <p><%#:Item.Description2 %></p>
                            <p><%#:Item.Description3 %></p>
                        </div>
                    </td>
                    <td>
                        <img src="/Images/Services/<%#:Item.ImagePath %>"
border="1" alt="<%#:Item.ServiceName %>" width="350" height="300" />
                    </td>
                </tr>
            </table>
        </ItemTemplate>
    </asp:FormView>

</asp:Content>
```

### 10.1.15 Services.aspx.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
using Griffin_Drywall.Models;
using System.Web.ModelBinding;


namespace Griffin_Drywall
{
    public partial class Services : System.Web.UI.Page
    {
        protected void Page_Load(object sender, EventArgs e)
        {

        }

        public IQueryable<Service> GetServices()
        {
            var db = new Griffin_Drywall.Models.GriffinContext();
            IQueryable<Service> query = db.Services;
            return query;
        }

        public IQueryable<Service> GetServiceById([QueryString("id")] int?
serviceId, [RouteData] string serviceName)
        {
            var _db = new Griffin_Drywall.Models.GriffinContext();
            IQueryable<Service> query = _db.Services;
            if (serviceId.HasValue && serviceId > 0)
            {
                query = query.Where(p => p.ServiceID == serviceId);
            }
            if (!String.IsNullOrEmpty(serviceName))
            {
                query = query.Where(p =>
                                    String.Compare(p.ServiceName,
                                    serviceName) == 0);
            }
            else
            {
                query = null;
            }
            return query;
        }

    }
}
```

### 10.1.16 Site.Master

```
<%@ Master Language="C#" AutoEventWireup="true" CodeBehind="Site.master.cs"
Inherits="Griffin_Drywall.SiteMaster" %>

<!DOCTYPE html>
<html lang="en">
<head runat="server">
    <meta charset="utf-8" />
    <title>Griffin Drywall</title>
    <asp:PlaceHolder runat="server">
          <%: Scripts.Render("~/bundles/modernizr") %>
    </asp:PlaceHolder>
    <webopt:BundleReference runat="server" Path="~/Content/css" />
    <link href="Images/griffin-drywall.ico" rel="shortcut icon" type="image/x-
generic">
    <meta name="viewport" content="width=device-width" />
    <asp:ContentPlaceHolder runat="server" ID="HeadContent" />
    <!--[if !IE 7]>
       <style type="text/css">
              #wrapper {display:table;height:100%}
       </style>
    <![endif]-->
    <script
src="http://ajax.googleapis.com/ajax/libs/jquery/1/jquery.js"></script>
    <script src="../Scripts/galleria-1.2.9.min.js"></script>

</head>
<script language = "javascript">
<!--
    if (document.images) {

        home = new Image
        homeOver = new Image

        services = new Image
        servicesOver = new Image

        projects = new Image
        projectsOver = new Image

        contactus = new Image
        contactusOver = new Image

        request = new Image
        requestOver = new Image

        request.src = '/Images/btnRequest.png'
        requestOver.src = '/Images/Rollover/btnRequest.png'

        home.src = '/Images/home_btn.png'
        homeOver.src = '/Images/Rollover/home_btn.png'

        services.src = '/Images/services_btn.png'
        servicesOver.src = '/Images/Rollover/services_btn.png'

        projects.src = '/Images/projects_btn.png'
        projectsOver.src = '/Images/Rollover/projects_btn.png'

        contactus.src = '/Images/contactus_btn.png'
        contactusOver.src = '/Images/Rollover/contactus_btn.png'
    }
    //--!>
</script>
<body>
    <form runat="server">

    <asp:ScriptManager runat="server">
        <Scripts>
            <%--Framework Scripts--%>

            <asp:ScriptReference Name="jquery" />
            <asp:ScriptReference Name="jquery.ui.combined" />
            <asp:ScriptReference Name="WebForms.js" Assembly="System.Web"
Path="~/Scripts/WebForms/WebForms.js" />
            <asp:ScriptReference Name="WebUIValidation.js" Assembly="System.Web"
Path="~/Scripts/WebForms/WebUIValidation.js" />
            <asp:ScriptReference Name="MenuStandards.js" Assembly="System.Web"
Path="~/Scripts/WebForms/MenuStandards.js" />
            <asp:ScriptReference Name="GridView.js" Assembly="System.Web"
Path="~/Scripts/WebForms/GridView.js" />
            <asp:ScriptReference Name="DetailsView.js" Assembly="System.Web"
Path="~/Scripts/WebForms/DetailsView.js" />
            <asp:ScriptReference Name="TreeView.js" Assembly="System.Web"
Path="~/Scripts/WebForms/TreeView.js" />
            <asp:ScriptReference Name="WebParts.js" Assembly="System.Web"
Path="~/Scripts/WebForms/WebParts.js" />
            <asp:ScriptReference Name="Focus.js" Assembly="System.Web"
Path="~/Scripts/WebForms/Focus.js" />
            <asp:ScriptReference Name="WebFormsBundle" />
            <%--Site Scripts--%>

        </Scripts>
    </asp:ScriptManager>
    <div id="wrapper">
        <header>
            <div class="content-wrapper">

                <div id="top">
                </div>
                <div id="logo">

                    <section id="login">
                    <div id="adhocMenu">
                        <a id="Admin1" class="adminBtn" visible="false" 
runat="server" href="~/Admin/AdminMenu"></a>
                    </div>
                    <div id="adhocMenu1">
                        <a id="Subcon" class="adminBtn" visible="false" 
runat="server" href="~/Subcon/AddWorkEntry"></a>
                    </div>
                    <asp:LoginView ID="LoginView1" runat="server" 
ViewStateMode="Disabled">
                        <AnonymousTemplate>
                            <ul>
                                <li><a id="registerLink" href="<%: 
GetRouteUrl("RegisterRoute", null) %>">Register</a></li>
                                <li><a id="loginLink" runat="server" 
href="~/Account/Login">Log in</a></li>
                            </ul>
                        </AnonymousTemplate>
                        <LoggedInTemplate>
                            <p>
                                Hello, <a id="A7" class="username" href="<%: 
GetRouteUrl("ManageRoute", null) %>" title="Manage your account">
                                    <asp:LoginName ID="LoginName1" runat="server" 
CssClass="username" /></a>
                                <asp:LoginStatus ID="LoginStatus1" runat="server" 
LogoutAction="Redirect" LogoutText="Log off" LogoutPageUrl="~/" />
                            </p>
                        </LoggedInTemplate>
                    </asp:LoginView>
                     </section>
                </div>
                <table id="navagtion-bar">
                    <tr>
                        <td>
                            <div id="Nav">
                            </div>
                        </td>
                        <td>
                            <div id="home-btn">
                                <a id="A1" class="navbtn" href="<%: 
GetRouteUrl("HomeRoute", null) %>" 
onmouseover="document.rollover1.src=homeOver.src" 
onmouseout="document.rollover1.src=home.src"><img runat="server" 
src="/Images/home_btn.png" border=0 name="rollover1"></a>
                            </div>
                        </td>
                        <td>
                            <div id="services-btn">
                                <a id="A2" class="navbtn" href="<%: 
GetRouteUrl("ServicesRoute", null) %>" 
onmouseover="document.rollover2.src=servicesOver.src" 
onmouseout="document.rollover2.src=services.src"><img runat="server" 
src="/Images/services_btn.png" border=0 name="rollover2"></a>
                            </div>
                        </td>
                        <td>
                            <div id="Nav008">
                            </div>
                        </td>
                        <td>
                            <div id="projects-btn">
                                <a id="A3" class="navbtn" href="<%:
GetRouteUrl("ProjectsRoute", null) %>"
onmouseover="document.rollover3.src=projectsOver.src"
onmouseout="document.rollover3.src=projects.src"><img runat="server"
src="/Images/projects_btn.png" border=0 name="rollover3"></a>
                            </div>
                        </td>
                        <td>
                            <div id="contactus-btn">
                                <a id="A4" class="navbtn" href="<%:
GetRouteUrl("ContactRoute", null) %>"
onmouseover="document.rollover4.src=contactusOver.src"
onmouseout="document.rollover4.src=contactus.src"><img runat="server"
src="/Images/contactus_btn.png" border=0 name="rollover4"></a>
                            </div>
                        </td>
                        <td>
                            <div id="Nav011">
                            </div>
                        </td>
                    </tr>
                </table>
                <div id="Nav-Bottom">
                </div>
            </div>
        </header>
    <div id="main">
        <asp:ContentPlaceHolder runat="server" ID="FeaturedContent" />
        <section class="content-wrapper2 main-content clear-fix">
            <asp:ContentPlaceHolder runat="server" ID="MainContent" />
        </section>
    </div>
    </div>
        <footer>
            <div id="footer-bottom">
                <div id="foot">
                    <div class="left footer-note">
                        <p>Copyright &copy; <%: DateTime.Now.Year %> Griffin
Drywall. Designed by Casey Griffin.</p>
                    </div>
                    <div class="right facebookicon">
                        <a id="A5" runat="server"
href="http://www.facebook.com/pages/Griffin-Drywall/149351565108396">
                            <img id="Img1" runat="server"
src="/Images/facebookicon.png" border="0"></a>
                    </div>
                </div>
            </div>
        </footer>
    </form>
</body>
</html>
```

### 10.1.17 Site.Master.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.Security;
using System.Web.UI;
using System.Web.UI.WebControls;
using Griffin_Drywall.Models;


namespace Griffin_Drywall
{
    public partial class SiteMaster : MasterPage
    {
        private const string AntiXsrfTokenKey = "__AntiXsrfToken";
        private const string AntiXsrfUserNameKey = "__AntiXsrfUserName";
        private string _antiXsrfTokenValue;

        protected void Page_Init(object sender, EventArgs e)
        {
            // The code below helps to protect against XSRF attacks
            var requestCookie = Request.Cookies[AntiXsrfTokenKey];
            Guid requestCookieGuidValue;
            if (requestCookie != null && Guid.TryParse(requestCookie.Value, out 
requestCookieGuidValue))
            {
                // Use the Anti-XSRF token from the cookie
                _antiXsrfTokenValue = requestCookie.Value;
                Page.ViewStateUserKey = _antiXsrfTokenValue;
            }
            else
            {
                // Generate a new Anti-XSRF token and save to the cookie
                _antiXsrfTokenValue = Guid.NewGuid().ToString("N");
                Page.ViewStateUserKey = _antiXsrfTokenValue;

                var responseCookie = new HttpCookie(AntiXsrfTokenKey)
                {
                    HttpOnly = true,
                    Value = _antiXsrfTokenValue
                };
                if (FormsAuthentication.RequireSSL && Request.IsSecureConnection)
                {
                    responseCookie.Secure = true;
                }
                Response.Cookies.Set(responseCookie);
            }

            Page.PreLoad += master_Page_PreLoad;
        }

        protected void master_Page_PreLoad(object sender, EventArgs e)
        {
            if (!IsPostBack)
            {
                // Set Anti-XSRF token
                ViewState[AntiXsrfTokenKey] = Page.ViewStateUserKey;
                ViewState[AntiXsrfUserNameKey] = Context.User.Identity.Name ??
String.Empty;
            }
            else
            {
                // Validate the Anti-XSRF token
                if ((string)ViewState[AntiXsrfTokenKey] != _antiXsrfTokenValue
                    || (string)ViewState[AntiXsrfUserNameKey] !=
(Context.User.Identity.Name ?? String.Empty))
                {
                    throw new InvalidOperationException("Validation of Anti-XSRF
token failed.");
                }
            }
        }

        protected void Page_Load(object sender, EventArgs e)
        {
            if (HttpContext.Current.User.IsInRole("Administrator"))
            {
                Admin1.Visible = true;
            }
            if (HttpContext.Current.User.IsInRole("SubContractor"))
            {
                Subcon.Visible = true;
            }
        }
    }
}
```

### 10.1.18 Web.config

```
<?xml version="1.0" encoding="utf-8"?>
<!--
  For more information on how to configure your ASP.NET application, please visit
  http://go.microsoft.com/fwlink/?LinkId=169433
  -->
<configuration>
  <configSections>
    <!-- For more information on Entity Framework configuration, visit
http://go.microsoft.com/fwlink/?LinkID=237468 -->
    <section name="entityFramework"
type="System.Data.Entity.Internal.ConfigFile.EntityFrameworkSection,
EntityFramework, Version=5.0.0.0, Culture=neutral,
PublicKeyToken=b77a5c561934e089" requirePermission="false" />
  </configSections>
  <connectionStrings>
    <add name="DefaultConnection" connectionString="Data Source=Nikki;Initial
Catalog=aspnet-Griffin_Drywall-20130216003113;Integrated
Security=SSPI;AttachDBFilename=|DataDirectory|\aspnet-Griffin_Drywall-
20130216003113.mdf" providerName="System.Data.SqlClient" />
    <add name="Griffin_Drywall" connectionString="Data
Source=nikki;Database=Griffin_Drywall;Integrated Security=True"
providerName="System.Data.SqlClient " />
  </connectionStrings>
  <appSettings>
    <add key="MyEmailAddr" value="cgriffindw@gmail.com" />
    <add key="ToEmailAddr" value="cgriffindw@gmail.com" />
    <add key="EmailHost" value="smtp.gmail.com" />
    <add key="FromEmailAddr" value="cgriffindw@gmail.com" />
  </appSettings>
  <system.net>
    <mailSettings>
      <smtp from="cgriffindw@gmail.com">
<!-- Password has been removed for privacy purposes -->
        <network host="smtp.gmail.com" password="*******" port="587"
userName="cgriffindw@gmail.com" />
      </smtp>
    </mailSettings>
  </system.net>
  <system.web>
    <compilation debug="true" targetFramework="4.5" />
    <httpRuntime targetFramework="4.5" />
    <urlMappings enabled="true">
      <add url="~/Account/Login" mappedUrl="~/Account/Login.aspx" />
      <add url="~/Home" mappedUrl="~/Default.aspx" />
      <add url="~/Account/Manage" mappedUrl="~/Account/Manage.aspx" />
      <add url="~/Admin/AdminMenu" mappedUrl="~/Admin/AdminPage.aspx" />
      <add url="~/Subcon/AddWorkEntry" mappedUrl="~/Subcon/AddWorkEntry.aspx" />
      <add url="~/Admin/MgnUsers" mappedUrl="~/Admin/MgnUsers.aspx" />
    </urlMappings>
    <pages>
      <namespaces>
        <add namespace="System.Web.Optimization" />
      </namespaces>
      <controls>
        <add assembly="Microsoft.AspNet.Web.Optimization.WebForms"
namespace="Microsoft.AspNet.Web.Optimization.WebForms" tagPrefix="webopt" />
        <add tagPrefix="dc" src="~/Controls/letterLinks.ascx" tagName="alphalinks"
/>
        <add tagPrefix="ajaxToolkit" assembly="AjaxControlToolkit"
namespace="AjaxControlToolkit" />
      </controls>
    </pages>
    <authentication mode="Forms">
      <forms loginUrl="~/Account/Login.aspx" defaultUrl="~/Home" timeout="2880" />
    </authentication>
    <profile defaultProvider="DefaultProfileProvider">
      <providers>
        <add name="DefaultProfileProvider"
type="System.Web.Providers.DefaultProfileProvider, System.Web.Providers,
Version=1.0.0.0, Culture=neutral, PublicKeyToken=31bf3856ad364e35"
connectionStringName="Griffin_Drywall" applicationName="/" />
      </providers>
    </profile>
    <membership defaultProvider="DefaultMembershipProvider">
      <providers>
        <add name="DefaultMembershipProvider"
type="System.Web.Providers.DefaultMembershipProvider, System.Web.Providers,
Version=1.0.0.0, Culture=neutral, PublicKeyToken=31bf3856ad364e35"
connectionStringName="Griffin_Drywall" enablePasswordRetrieval="false"
enablePasswordReset="true" requiresQuestionAndAnswer="false"
requiresUniqueEmail="false" maxInvalidPasswordAttempts="5"
minRequiredPasswordLength="6" minRequiredNonalphanumericCharacters="0"
passwordAttemptWindow="10" applicationName="/" />
      </providers>
    </membership>
    <roleManager enabled="true" defaultProvider="DefaultRoleProvider">
      <providers>
        <add name="DefaultRoleProvider"
type="System.Web.Providers.DefaultRoleProvider, System.Web.Providers,
Version=1.0.0.0, Culture=neutral, PublicKeyToken=31bf3856ad364e35"
connectionStringName="Griffin_Drywall" applicationName="/" />
      </providers>
    </roleManager>
    <!--
            If you are deploying to a cloud environment that has multiple web
server instances,
            you should change session state mode from "InProc" to "Custom". In
addition,
            change the connection string named "DefaultConnection" to connect to
an instance
            of SQL Server (including SQL Azure and SQL  Compact) instead of to SQL
Server Express.
      -->
    <sessionState mode="InProc" customProvider="DefaultSessionProvider">
      <providers>
        <add name="DefaultSessionProvider"
type="System.Web.Providers.DefaultSessionStateProvider, System.Web.Providers,
Version=1.0.0.0, Culture=neutral, PublicKeyToken=31bf3856ad364e35"
connectionStringName="Griffin_Drywall" />
      </providers>
    </sessionState>
  </system.web>
  <entityFramework>
    <defaultConnectionFactory
type="System.Data.Entity.Infrastructure.LocalDbConnectionFactory,
EntityFramework">
      <parameters>
        <parameter value="v11.0" />
      </parameters>
    </defaultConnectionFactory>
  </entityFramework>
</configuration>
```

## 10.2 Admin Pages

### 10.2.1 AddUser.aspx

```
<%@ Page Title="" Language="C#" MasterPageFile="~/Site.Master"
AutoEventWireup="true" CodeBehind="AddUser.aspx.cs"
Inherits="Griffin_Drywall.Admin.AddUser" %>
<asp:Content ID="Content1" ContentPlaceHolderID="HeadContent" runat="server">
</asp:Content>
<asp:Content ID="Content2" ContentPlaceHolderID="FeaturedContent" runat="server">
</asp:Content>
<asp:Content ID="Content3" ContentPlaceHolderID="MainContent" runat="server">
    <table class="webparts">
        <tr>
            <th>Add User</th>
        </tr>
        <tr>
            <td class="details valign">

                <h3>Roles:</h3>
                <asp:CheckBoxList ID="UserRoles" class="userRoles" runat="server"
/>

                <h3>Main Info:</h3>

                <table>
                    <tr>
                        <td class="detailheader">Active User</td>
                        <td>
                            <asp:CheckBox ID="isapproved" runat="server"
Checked="true" />
                        </td>
                    </tr>
                    <tr>
                        <td class="detailheader">User Name</td>
                        <td>
                            <asp:TextBox ID="username"
runat="server"></asp:TextBox>
                        </td>
                    </tr>
                    <tr>
                        <td class="detailheader">Password</td>
                        <td>
                            <asp:TextBox ID="password"
runat="server"></asp:TextBox>
                        </td>
                    </tr>
                    <tr>
                        <td class="detailheader">Email</td>
                        <td>
                            <asp:TextBox ID="email" runat="server"></asp:TextBox>
                        </td>
                    </tr>
                    <tr>
                        <td class="detailheader">Comment</td>
                        <td>
                            <asp:TextBox ID="comment"
runat="server"></asp:TextBox>
```

```
                        </td>
                    </tr>
                    <tr>
                        <td colspan="2">
                            <br />
                            <input type="submit" value="Add User" />&nbsp;
              <input type="reset" />
                        </td>
                    </tr>
                    <tr>
                        <td colspan="2">
                            <div id="ConfirmationMessage" runat="server"
class="alert"></div>
                        </td>
                    </tr>
                </table>

                <asp:ObjectDataSource ID="MemberData" runat="server"
DataObjectTypeName="System.Web.Security.MembershipUser" SelectMethod="GetUser"
UpdateMethod="UpdateUser" TypeName="System.Web.Security.Membership">
                    <SelectParameters>
                        <asp:QueryStringParameter Name="username"
QueryStringField="username" />
                    </SelectParameters>
                </asp:ObjectDataSource>
            </td>
        </tr>
    </table>
    <a id="A1" runat="server" href="~/Admin/MgnUsers">Back to Menu</a>
</asp:Content>
```

### 10.2.2 AddUser.aspx.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Web.Security;

namespace Griffin_Drywall.Admin
{
    public partial class AddUser : System.Web.UI.Page
    {
        MembershipUser user;

        private void Page_Load()
        {
            if (IsPostBack)
            {
                try
                {
                    AddUserS();

                    Response.Redirect(GetRouteUrl("MgnUsersRoute", null));
                }
                catch (Exception ex)
                {
                    ConfirmationMessage.InnerText = "Insert Failure: " +
ex.Message;
                }
            }
        }

        protected void AddUserS()
        {
            // Add User.
            MembershipUser newUser = Membership.CreateUser(username.Text,
password.Text, email.Text);
            newUser.Comment = comment.Text;
            Membership.UpdateUser(newUser);

            // Add Roles.
            foreach (ListItem rolebox in UserRoles.Items)
            {
                if (rolebox.Selected)
                {
                    Roles.AddUserToRole(username.Text, rolebox.Text);
                }
            }
        }

        private void Page_PreRender()
        {
            UserRoles.DataSource = Roles.GetAllRoles();
            UserRoles.DataBind();
        }
    }
}
```

### 10.2.3 AdminPage.aspx

```
<%@ Page Title="" Language="C#" MasterPageFile="~/Site.Master"
AutoEventWireup="true" CodeBehind="AdminPage.aspx.cs"
Inherits="Griffin_Drywall.Admin.AdminPage" %>
<asp:Content ID="Content1" ContentPlaceHolderID="HeadContent" runat="server">
</asp:Content>
<asp:Content ID="Content2" ContentPlaceHolderID="FeaturedContent" runat="server">
</asp:Content>
<asp:Content ID="Content3" ContentPlaceHolderID="MainContent" runat="server">
    <ul class="adminList">
        <li><a id="addUser" href="<%: GetRouteUrl("AddUserRoute", null) %>">Add
User</a></li>
        <li><a id="manageUsers" href="<%: GetRouteUrl("MgnUsersRoute", null)
%>">Manage Users</a></li>
        <li><a id="manageServiceTypes" href="<%:
GetRouteUrl("EditServiceTypeRoute", null) %>">Manage Service Types</a></li>
        <li><a id="manageServices" href="<%: GetRouteUrl("EditServiceRoute", null)
%>">Manage Services</a></li>
        <li><a id="manageProjects" href="<%: GetRouteUrl("EditProjectRoute", null)
%>">Manage Projects</a></li>
        <li><a id="managePGallery" href="<%: GetRouteUrl("EditPGalleryRoute",
null) %>">Manage Project Galleries</a></li>
        <li><a id="manageJobs" href="<%: GetRouteUrl("EditJobRoute", null)
%>">Manage Jobs</a></li>
        <li><a id="manageSubCons" href="<%: GetRouteUrl("EditSubConRoute", null)
%>">Manage Subcontractors</a></li>
```

```
        <li><a id="manageWork" href="<%: GetRouteUrl("EditWorkRoute", null)
%>">Schedule Work</a></li>
        <li><a id="viewWork" href="<%: GetRouteUrl("ViewWorkRoute", null) %>">View
Work</a></li>
    </ul>
</asp:Content>
```

### 10.2.4 AdminPage.aspx.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;

namespace Griffin_Drywall.Admin
{
    public partial class AdminPage : System.Web.UI.Page
    {
        protected void Page_Load(object sender, EventArgs e)
        {

        }
    }
}
```

### 10.2.5 EditJob.aspx

```
<%@ Page Title="" Language="C#" MasterPageFile="~/Site.Master"
AutoEventWireup="true" CodeBehind="EditJob.aspx.cs"
Inherits="Griffin_Drywall.Admin.EditJob" %>

<asp:Content ID="Content1" ContentPlaceHolderID="HeadContent" runat="server">
</asp:Content>
<asp:Content ID="Content2" ContentPlaceHolderID="FeaturedContent" runat="server">
</asp:Content>
<asp:Content ID="Content3" ContentPlaceHolderID="MainContent" runat="server">
    <article class="projects">
       <h1>Manange Jobs</h1>
        <p>

        </p>
   </article>

    <asp:Label ID="lblMessage" runat="server" Text="" ForeColor="Red"></asp:Label>
    <div style="vertical-align: top; left: auto; overflow: auto; width: auto;">
        <asp:GridView ID="gridJob" runat="server" AutoGenerateColumns="False"
ShowFooter="True"
            CssClass="" OnRowCommand="gridJob_RowCommand"
            DataKeyNames="JobID" CellPadding="4" ForeColor="#333333"
            GridLines="None" OnRowCancelingEdit="gridJob_RowCancelingEdit"
            OnRowEditing="gridJob_RowEditing"
            OnRowUpdating="gridJob_RowUpdating"
            onrowdatabound="gridJob_RowDataBound"
            OnRowDeleting="gridJob_RowDeleting">
            <AlternatingRowStyle BackColor="White" />
            <Columns>
                <asp:TemplateField HeaderText="">
```

```
                    <ItemTemplate>
                        <asp:LinkButton ID="lnkEdit" runat="server" Text=""
CommandName="Edit" ToolTip="Edit"
                            CommandArgument=''><img src="../Images/edit.png"
/></asp:LinkButton>
                        <asp:LinkButton ID="lnkDelete" runat="server"
Text="Delete" CommandName="Delete"
                            ToolTip="Delete" OnClientClick='return confirm("Are
you sure you want to delete this entry?");'
                            CommandArgument=''><img src="../Images/delete.png"
/></asp:LinkButton>
                    </ItemTemplate>
                    <EditItemTemplate>
                        <asp:LinkButton ID="lnkInsert" runat="server" Text=""
ValidationGroup="editGrp" CommandName="Update" ToolTip="Save"
                            CommandArgument=''><img src="../Images/save.png"
/></asp:LinkButton>
                        <asp:LinkButton ID="lnkCancel" runat="server" Text=""
CommandName="Cancel" ToolTip="Cancel"
                            CommandArgument=''><img src="../Images/refresh.png"
/></asp:LinkButton>
                    </EditItemTemplate>
                    <FooterTemplate>
                        <asp:LinkButton ID="lnkInsert" runat="server" Text=""
ValidationGroup="newGrp" CommandName="InsertNew" ToolTip="Add New Entry"
                            CommandArgument=''><img src="../Images/add.png"
/></asp:LinkButton>
                        <asp:LinkButton ID="lnkCancel" runat="server" Text=""
CommandName="CancelNew" ToolTip="Cancel"
                            CommandArgument=''><img src="../Images/refresh.png"
/></asp:LinkButton>
                    </FooterTemplate>
                </asp:TemplateField>
                <asp:TemplateField HeaderText="Job Name">
                    <EditItemTemplate>
                        <asp:TextBox ID="txtJobName" runat="server" Width="150"
Text='<%# Bind("JobName") %>' CssClass="" MaxLength="100"></asp:TextBox>
                        <asp:RequiredFieldValidator ID="valJobName" runat="server"
ControlToValidate="txtJobName"
                            Display="Dynamic" ErrorMessage="Job Name is required."
ForeColor="Red" SetFocusOnError="True"

ValidationGroup="editGrp">*</asp:RequiredFieldValidator>
                    </EditItemTemplate>
                    <ItemTemplate>
                        <asp:Label ID="lblJobName" runat="server" Width="150"
Text='<%# Bind("JobName") %>'></asp:Label>
                    </ItemTemplate>
                    <FooterTemplate>
                        <asp:TextBox ID="txtJobNameNew" runat="server" Width="150"
CssClass="" MaxLength="100"></asp:TextBox>
                        <asp:RequiredFieldValidator ID="valJobNameNew"
runat="server" ControlToValidate="txtJobNameNew"
                            Display="Dynamic" ErrorMessage="Job Name is required."
ForeColor="Red" SetFocusOnError="True"

ValidationGroup="newGrp">*</asp:RequiredFieldValidator>
                    </FooterTemplate>
                </asp:TemplateField>
                <asp:TemplateField HeaderText="Description">
                    <EditItemTemplate>

                        <asp:TextBox ID="txtDescription" runat="server" Text='<%#
Bind("Description") %>' CssClass="" TextMode="multiline"
MaxLength="10000"></asp:TextBox>

                    </EditItemTemplate>
                    <ItemTemplate>
                        <asp:Label ID="lblDescription" runat="server" Style="text-
overflow: ellipsis; overflow: hidden" Width="300px" Height="50px" Text='<%#
Bind("Description") %>'></asp:Label>
                    </ItemTemplate>
                    <FooterTemplate>
                        <asp:TextBox ID="txtDescriptionNew" runat="server"
CssClass="" Style="text-overflow: ellipsis; overflow: hidden; margin-left: 5px"
Width="280px"  Height="22px" TextMode="multiline" MaxLength="10000"></asp:TextBox>

                    </FooterTemplate>
                </asp:TemplateField>
                <asp:TemplateField HeaderText="Begin Date">
                    <EditItemTemplate>
                        <asp:TextBox ID="txtBeginDate" runat="server" Width="150"
Text='<%# Bind("BeginDate", "{0:MM/dd/yyyy}") %>' CssClass=""
MaxLength="50"></asp:TextBox>
                        <asp:RequiredFieldValidator ID="valBeginDate"
runat="server" ControlToValidate="txtBeginDate"
                            Display="Dynamic" ErrorMessage="Begin Date is
required." ForeColor="Red" SetFocusOnError="True"

ValidationGroup="editGrp">*</asp:RequiredFieldValidator>
                        <asp:CompareValidator
                            ID="dateValidator1" runat="server"
                            Type="Date"
                            Operator="DataTypeCheck"
                            ControlToValidate="txtBeginDate"
                            ErrorMessage="Please enter a valid date."
                            ForeColor="Red" SetFocusOnError="True"
                            ValidationGroup="editGrp">*
                        </asp:CompareValidator>
                        <ajaxToolkit:CalendarExtender
                            ID="CalendarExtender1"
                            TargetControlID="txtBeginDate"
                            runat="server" />
                    </EditItemTemplate>
                    <ItemTemplate>
                        <asp:Label ID="lblBeginDate" runat="server" Width="150"
Text='<%# Bind("BeginDate", "{0:MM/dd/yyyy}") %>'></asp:Label>
                    </ItemTemplate>
                    <FooterTemplate>
                        <asp:TextBox ID="txtBeginDateNew" runat="server"
Width="150" CssClass="" MaxLength="50"></asp:TextBox>
                        <asp:RequiredFieldValidator ID="valBeginDateNew"
runat="server" ControlToValidate="txtBeginDateNew"
                            Display="Dynamic" ErrorMessage="Begin Date is
required." ForeColor="Red" SetFocusOnError="True"
```

```
ValidationGroup="newGrp">*</asp:RequiredFieldValidator>
                        <asp:CompareValidator
                            ID="dateValidator2" runat="server"
                            Type="Date"
                            Operator="DataTypeCheck"
                            ControlToValidate="txtBeginDateNew"
                            ErrorMessage="Please enter a valid date."
                            ForeColor="Red" SetFocusOnError="True"
                            ValidationGroup="newGrp">*
                        </asp:CompareValidator>
                        <ajaxToolkit:CalendarExtender
                            ID="CalendarExtender2"
                            TargetControlID="txtBeginDateNew"
                            runat="server" />
                    </FooterTemplate>
                </asp:TemplateField>
                <asp:TemplateField HeaderText="End Date">
                    <EditItemTemplate>
                        <asp:TextBox ID="txtEndDate" runat="server" Width="150"
Text='<%# Bind("EndDate", "{0:MM/dd/yyyy}") %>' CssClass=""
MaxLength="50"></asp:TextBox>
                        <asp:RequiredFieldValidator ID="valEndDate" runat="server"
ControlToValidate="txtEndDate"
                            Display="Dynamic" ErrorMessage="End Date is required."
ForeColor="Red" SetFocusOnError="True"

ValidationGroup="editGrp">*</asp:RequiredFieldValidator>
                        <asp:CompareValidator
                            ID="dateValidator3" runat="server"
                            Type="Date"
                            Operator="DataTypeCheck"
                            ControlToValidate="txtEndDate"
                            ErrorMessage="Please enter a valid date."
                            ForeColor="Red" SetFocusOnError="True"
                            ValidationGroup="editGrp">*
                        </asp:CompareValidator>
                        <ajaxToolkit:CalendarExtender
                            ID="CalendarExtender3"
                            TargetControlID="txtEndDate"
                            runat="server" />
                    </EditItemTemplate>
                    <ItemTemplate>
                        <asp:Label ID="lblEndDate" runat="server" Width="150"
Text='<%# Bind("EndDate", "{0:MM/dd/yyyy}") %>'></asp:Label>
                    </ItemTemplate>
                    <FooterTemplate>
                        <asp:TextBox ID="txtEndDateNew" runat="server" Width="150"
CssClass="" MaxLength="50"></asp:TextBox>
                        <asp:RequiredFieldValidator ID="vaEndDateNew"
runat="server" ControlToValidate="txtEndDateNew"
                            Display="Dynamic" ErrorMessage="End Date is required."
ForeColor="Red" SetFocusOnError="True"

ValidationGroup="newGrp">*</asp:RequiredFieldValidator>
                        <asp:CompareValidator
                            ID="dateValidator4" runat="server"
                            Type="Date"
                            Operator="DataTypeCheck"
                            ControlToValidate="txtEndDateNew"
                            ErrorMessage="Please enter a valid date."
                            ForeColor="Red" SetFocusOnError="True"
                            ValidationGroup="newGrp">*
                        </asp:CompareValidator>
                        <ajaxToolkit:CalendarExtender
                            ID="CalendarExtender4"
                            TargetControlID="txtEndDateNew"
                            runat="server" />
                    </FooterTemplate>
                </asp:TemplateField>

                <asp:TemplateField HeaderText="Project">
                    <EditItemTemplate>
                        <asp:DropDownList ID="ddlProject" runat="server">
                        </asp:DropDownList>
                        <asp:RequiredFieldValidator ID="valProject" runat="server"
ControlToValidate="ddlProject"
                            Display="Dynamic" ErrorMessage="Project is required."
ForeColor="Red" SetFocusOnError="True"

ValidationGroup="editGrp">*</asp:RequiredFieldValidator>
                    </EditItemTemplate>
                    <ItemTemplate>
                        <asp:Label ID="lblProject" runat="server" Text='<%#
Bind("Project.ProjectName") %>'></asp:Label>
                    </ItemTemplate>
                    <FooterTemplate>
                        <asp:DropDownList ID="ddlProjectNew" runat="server">
                        </asp:DropDownList>
                        <asp:RequiredFieldValidator ID="valProjectNew"
runat="server" ControlToValidate="ddlProjectNew"
                            Display="Dynamic" ErrorMessage="Project is required."
ForeColor="Red" SetFocusOnError="True"

ValidationGroup="newGrp">*</asp:RequiredFieldValidator>
                    </FooterTemplate>
                </asp:TemplateField>

            </Columns>
            <EditRowStyle BackColor="#2461BF" />
            <FooterStyle BackColor="#507CD1" Font-Bold="True" ForeColor="White" />
            <HeaderStyle BackColor="#507CD1" Font-Bold="True" ForeColor="White" />
            <PagerStyle BackColor="#2461BF" ForeColor="White"
HorizontalAlign="Center" />
            <RowStyle BackColor="#EFF3FB" />
            <SelectedRowStyle BackColor="#D1DDF1" Font-Bold="True"
ForeColor="#333333" />
            <SortedAscendingCellStyle BackColor="#F5F7FB" />
            <SortedAscendingHeaderStyle BackColor="#6D95E1" />
            <SortedDescendingCellStyle BackColor="#E9EBEF" />
            <SortedDescendingHeaderStyle BackColor="#4870BE" />
        </asp:GridView>
        <asp:Repeater ID="rptPager" runat="server">
            <ItemTemplate>
                <asp:LinkButton ID="lnkPage" runat="server"
                    Text='<%#Eval("Text") %>'
                    CommandArgument='<%#Eval("Value") %>'
                    Enabled='<%#Eval("Enabled") %>'
                    OnClick="Page_Changed"
                    ForeColor="#267CB2"
                    Font-Bold="true" />
            </ItemTemplate>
        </asp:Repeater>
    </div>
    <asp:ValidationSummary ID="ValidationSummary1" ValidationGroup="newGrp"
class="validation-summary-errors" runat="server" />
    <asp:ValidationSummary ID="ValidationSummary2" ValidationGroup="editGrp"
class="validation-summary-errors" runat="server" />
    <br />
    <a id="A1" runat="server" href="~/Admin/AdminMenu">Back to Menu</a>
</asp:Content>
```

### 10.2.6 EditJob.aspx.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
using Griffin_Drywall.Models;
using System.Web.ModelBinding;

namespace Griffin_Drywall.Admin
{
    public partial class EditJob : System.Web.UI.Page
    {
        protected void Page_Load(object sender, EventArgs e)
        {
            if (!IsPostBack)
            {
                BindGrid(0);
            }
            lblMessage.Text = "";
        }

        void BindGrid(int pageIndex)
        {
            using (GriffinContext context = new
Griffin_Drywall.Models.GriffinContext())
            {
                int totalRecords = context.Jobs.Count();
                int pageSize = 10;
                if (pageIndex < 0)
                { pageIndex = 0; }
                int startRow = pageIndex * pageSize;

                if (context.Jobs.Count() > 0)
                {
                    gridJob.DataSource = context.Jobs.OrderBy(x =>
x.JobID).Skip(startRow).Take(pageSize).ToList();
                    gridJob.DataBind();

                    BindPager(totalRecords, pageIndex, pageSize);
                }
                else
                {
                    var obj = new List<Job>();
                    obj.Add(new Job());
                    // Bind the DataTable which contain a blank row to the
GridView
                    gridJob.DataSource = obj.ToList();
                    gridJob.DataBind();
                    int columnsCount = gridJob.Columns.Count;
                    gridJob.Rows[0].Cells.Clear();// clear all the cells in the
row
                    gridJob.Rows[0].Cells.Add(new TableCell()); //add a new blank
cell
                    gridJob.Rows[0].Cells[0].ColumnSpan = columnsCount; //set the
column span to the new added cell

                    //You can set the styles here
                    gridJob.Rows[0].Cells[0].HorizontalAlign =
HorizontalAlign.Center;
                    gridJob.Rows[0].Cells[0].ForeColor = System.Drawing.Color.Red;
                    gridJob.Rows[0].Cells[0].Font.Bold = true;
                    //set No Results found to the new added cell
                    gridJob.Rows[0].Cells[0].Text = "NO RESULT FOUND!";
                }
            }
        }

        private void BindPager(int totalRecordCount, int currentPageIndex, int
pageSize)
        {
            double getPageCount = (double)((decimal)totalRecordCount /
(decimal)pageSize);
            int pageCount = (int)Math.Ceiling(getPageCount);
            List<ListItem> pages = new List<ListItem>();
            if (pageCount > 1)
            {
                pages.Add(new ListItem("FIRST", "1", currentPageIndex > 1));
                for (int i = 1; i <= pageCount; i++)
                {
                    pages.Add(new ListItem(i.ToString(), i.ToString(), i !=
currentPageIndex + 1));
                }
                pages.Add(new ListItem("LAST", pageCount.ToString(),
currentPageIndex < pageCount - 1));
            }

            rptPager.DataSource = pages;
            rptPager.DataBind();
        }

        protected void Page_Changed(object sender, EventArgs e)
        {
            int pageIndex = Convert.ToInt32(((sender as
LinkButton).CommandArgument));
            gridJob.PageIndex = pageIndex;
            BindGrid(pageIndex - 1);
        }
        protected void gridJob_RowDataBound(object sender, GridViewRowEventArgs e)
        {
            DropDownList ddl = null;
            if (e.Row.RowType == DataControlRowType.Footer)
            {
                ddl = e.Row.FindControl("ddlProjectNew") as DropDownList;
            }
            if (e.Row.RowType == DataControlRowType.DataRow)
            {
                ddl = e.Row.FindControl("ddlProject") as DropDownList;
            }
            if (ddl != null)
            {
                using (GriffinContext context = new
Griffin_Drywall.Models.GriffinContext())
                {
                    ddl.DataSource = context.Projects.ToList();
                    ddl.DataTextField = "ProjectName";
                    ddl.DataValueField = "ProjectID";
                    ddl.DataBind();
                    ddl.Items.Insert(0, new ListItem(""));
                }
                if (e.Row.RowType == DataControlRowType.DataRow)
                {
                    ddl.SelectedValue =
((Job)(e.Row.DataItem)).ProjectID.ToString();
                }
            }
        }

        protected void gridJob_RowCommand(object sender, GridViewCommandEventArgs
e)
        {
            if (e.CommandName == "InsertNew")
            {
                GridViewRow row = gridJob.FooterRow;
                TextBox txtJobName = row.FindControl("txtJobNameNew") as TextBox;
                TextBox txtDescription = row.FindControl("txtDescriptionNew") as
TextBox;
                TextBox txtBeginDate = row.FindControl("txtBeginDateNew") as
TextBox;
                DateTime dtBeginDate = DateTime.Parse(txtBeginDate.Text);
                TextBox txtEndDate = row.FindControl("txtEndDateNew") as TextBox;
                DateTime dtEndDate = DateTime.Parse(txtEndDate.Text);
                DropDownList ddlProject = row.FindControl("ddlProjectNew") as
DropDownList;

                if (txtJobName != null && ddlProject != null)
                {
                    using (GriffinContext context = new
Griffin_Drywall.Models.GriffinContext())
                    {
                        Job obj = new Job();
                        obj.JobName = txtJobName.Text;
                        obj.Description = txtDescription.Text;
                        obj.BeginDate = dtBeginDate;
                        obj.EndDate = dtEndDate;
                        obj.ProjectID = Convert.ToInt32(ddlProject.SelectedValue);
                        context.Jobs.Add(obj);
                        context.SaveChanges();
                        lblMessage.Text = "Added successfully.";
                        BindGrid(((GridView)sender).PageIndex - 1);
                    }
                }
                else
                {
                    lblMessage.Text = "Unable to add new job to database.";
                }
            }
        }

        protected void gridJob_RowEditing(object sender, GridViewEditEventArgs e)
        {
            gridJob.EditIndex = e.NewEditIndex;
            BindGrid(((GridView)sender).PageIndex - 1);
        }
        protected void gridJob_RowCancelingEdit(object sender,
GridViewCancelEditEventArgs e)
        {
            gridJob.EditIndex = -1;
            BindGrid(((GridView)sender).PageIndex - 1);
        }

        protected void gridJob_RowUpdating(object sender, GridViewUpdateEventArgs
e)
        {
            GridViewRow row = gridJob.Rows[e.RowIndex];

            TextBox txtJobName = row.FindControl("txtJobName") as TextBox;
            TextBox txtDescription = row.FindControl("txtDescription") as TextBox;
            TextBox txtBeginDate = row.FindControl("txtBeginDate") as TextBox;
            DateTime dtBeginDate = DateTime.Parse(txtBeginDate.Text);
            TextBox txtEndDate = row.FindControl("txtEndDate") as TextBox;
            DateTime dtEndDate = DateTime.Parse(txtEndDate.Text);
            DropDownList ddlProject = row.FindControl("ddlProject") as
DropDownList;

            if (txtJobName != null && ddlProject != null)
            {
                int jobID = Convert.ToInt32(gridJob.DataKeys[e.RowIndex].Value);
                using (GriffinContext context = new
Griffin_Drywall.Models.GriffinContext())
                {
                    Job obj = context.Jobs.First(x => x.JobID == jobID);
                    obj.JobName = txtJobName.Text;
                    obj.Description = txtDescription.Text;
                    obj.BeginDate = dtBeginDate;
                    obj.EndDate = dtEndDate;
                    obj.ProjectID = Convert.ToInt32(ddlProject.SelectedValue);
                    context.SaveChanges();
                    lblMessage.Text = "Saved successfully.";
                    gridJob.EditIndex = -1;
                    BindGrid(((GridView)sender).PageIndex - 1);
                }
            }
            else
            {
                lblMessage.Text = "Unable to edit job in database.";
            }
        }

        protected void gridJob_RowDeleting(object sender, GridViewDeleteEventArgs
e)
        {
            int jobID = Convert.ToInt32(gridJob.DataKeys[e.RowIndex].Value);
            using (GriffinContext context = new
Griffin_Drywall.Models.GriffinContext())
            {
                Job obj = context.Jobs.First(x => x.JobID == jobID);
                context.Jobs.Remove(obj);
                context.SaveChanges();
                BindGrid(0);
                lblMessage.Text = "Deleted successfully.";
            }
        }
    }
}
```

### 10.2.7 EditPGallery.aspx

```
<%@ Page Title="" Language="C#" MasterPageFile="~/Site.Master"
AutoEventWireup="true" CodeBehind="EditPGallery.aspx.cs"
Inherits="Griffin_Drywall.Admin.EditPGallery" %>
<asp:Content ID="Content1" ContentPlaceHolderID="HeadContent" runat="server">
</asp:Content>
<asp:Content ID="Content2" ContentPlaceHolderID="FeaturedContent" runat="server">
</asp:Content>
<asp:Content ID="Content3" ContentPlaceHolderID="MainContent" runat="server">
    <article class="projects">
       <h1>Manange Project Galleries</h1>
        <p>

        </p>
   </article>
    <asp:Label ID="lblMessage" runat="server" Text="" ForeColor="Red"></asp:Label>
    <div style="vertical-align: top; left: auto; overflow: auto; width: auto;">
        <asp:GridView ID="gridPGallery" runat="server" AutoGenerateColumns="False"
ShowFooter="True"
            CssClass="" OnRowCommand="gridPGallery_RowCommand"
            DataKeyNames="GImageID" CellPadding="4" ForeColor="#333333"
            GridLines="None" OnRowCancelingEdit="gridPGallery_RowCancelingEdit"
            OnRowEditing="gridPGallery_RowEditing"
            OnRowUpdating="gridPGallery_RowUpdating"
            onrowdatabound="gridPGallery_RowDataBound"
            OnRowDeleting="gridPGallery_RowDeleting">
            <AlternatingRowStyle BackColor="White" />
            <Columns>
                <asp:TemplateField HeaderText="">
                    <ItemTemplate>
                        <asp:LinkButton ID="lnkEdit" runat="server" Text=""
CommandName="Edit" ToolTip="Edit"
                            CommandArgument=''><img src="../Images/edit.png"
/></asp:LinkButton>
                        <asp:LinkButton ID="lnkDelete" runat="server"
Text="Delete" CommandName="Delete"
```

```
                            ToolTip="Delete" OnClientClick='return confirm("Are
you sure you want to delete this entry?");'
                            CommandArgument=''><img src="../Images/delete.png"
/></asp:LinkButton>
                    </ItemTemplate>
                    <EditItemTemplate>
                        <asp:LinkButton ID="lnkInsert" runat="server" Text=""
ValidationGroup="editGrp" CommandName="Update" ToolTip="Save"
                            CommandArgument=''><img src="../Images/save.png"
/></asp:LinkButton>
                        <asp:LinkButton ID="lnkCancel" runat="server" Text=""
CommandName="Cancel" ToolTip="Cancel"
                            CommandArgument=''><img src="../Images/refresh.png"
/></asp:LinkButton>
                    </EditItemTemplate>
                    <FooterTemplate>
                        <asp:LinkButton ID="lnkInsert" runat="server" Text=""
ValidationGroup="newGrp" CommandName="InsertNew" ToolTip="Add New Entry"
                            CommandArgument=''><img src="../Images/add.png"
/></asp:LinkButton>
                        <asp:LinkButton ID="lnkCancel" runat="server" Text=""
CommandName="CancelNew" ToolTip="Cancel"
                            CommandArgument=''><img src="../Images/refresh.png"
/></asp:LinkButton>
                    </FooterTemplate>
                </asp:TemplateField>
                <asp:TemplateField HeaderText="Project">
                    <EditItemTemplate>
                        <asp:DropDownList ID="ddlProject" runat="server">
                        </asp:DropDownList>
                        <asp:RequiredFieldValidator ID="valProject" runat="server"
ControlToValidate="ddlProject"
                            Display="Dynamic" ErrorMessage="Project is required."
ForeColor="Red" SetFocusOnError="True"

ValidationGroup="editGrp">*</asp:RequiredFieldValidator>
                    </EditItemTemplate>
                    <ItemTemplate>
                        <asp:Label ID="lblProject" runat="server" Text='<%#
Bind("Project.ProjectName") %>'></asp:Label>
                    </ItemTemplate>
                    <FooterTemplate>
                        <asp:DropDownList ID="ddlProjectNew" runat="server">
                        </asp:DropDownList>
                        <asp:RequiredFieldValidator ID="valProjectNew"
runat="server" ControlToValidate="ddlProjectNew"
                            Display="Dynamic" ErrorMessage="Project is required."
ForeColor="Red" SetFocusOnError="True"

ValidationGroup="newGrp">*</asp:RequiredFieldValidator>
                    </FooterTemplate>
                </asp:TemplateField>
                <asp:TemplateField HeaderText="Image Path">
                    <EditItemTemplate>
                        <asp:FileUpload ID="flImagePath" runat="server" />
                    </EditItemTemplate>
                    <ItemTemplate>
```

```
                        <asp:Label ID="lblImagePath" runat="server" Width="150"
Text='<%# Bind("ImagePath") %>'></asp:Label>
                    </ItemTemplate>
                    <FooterTemplate>
                        <asp:FileUpload ID="flImagePathNew" runat="server" />
                        <asp:RequiredFieldValidator ID="valImagePathNew"
runat="server" ErrorMessage="Image path required."  ForeColor="Red"
ControlToValidate="flImagePathNew" SetFocusOnError="true" Display="Dynamic"
ValidationGroup="newGrp">*</asp:RequiredFieldValidator>
                    </FooterTemplate>
                </asp:TemplateField>
            </Columns>
            <EditRowStyle BackColor="#2461BF" />
            <FooterStyle BackColor="#507CD1" Font-Bold="True" ForeColor="White" />
            <HeaderStyle BackColor="#507CD1" Font-Bold="True" ForeColor="White" />
            <PagerStyle BackColor="#2461BF" ForeColor="White"
HorizontalAlign="Center" />
            <RowStyle BackColor="#EFF3FB" />
            <SelectedRowStyle BackColor="#D1DDF1" Font-Bold="True"
ForeColor="#333333" />
            <SortedAscendingCellStyle BackColor="#F5F7FB" />
            <SortedAscendingHeaderStyle BackColor="#6D95E1" />
            <SortedDescendingCellStyle BackColor="#E9EBEF" />
            <SortedDescendingHeaderStyle BackColor="#4870BE" />
        </asp:GridView>
        <asp:Repeater ID="rptPager" runat="server">
            <ItemTemplate>
                <asp:LinkButton ID="lnkPage" runat="server"
                    Text='<%#Eval("Text") %>'
                    CommandArgument='<%#Eval("Value") %>'
                    Enabled='<%#Eval("Enabled") %>'
                    OnClick="Page_Changed"
                    ForeColor="#267CB2"
                    Font-Bold="true" />
            </ItemTemplate>
        </asp:Repeater>
    </div>
    <asp:ValidationSummary ID="ValidationSummary1" ValidationGroup="newGrp"
class="validation-summary-errors" runat="server" />
    <asp:ValidationSummary ID="ValidationSummary2" ValidationGroup="editGrp"
class="validation-summary-errors" runat="server" />
    <br />
    <a id="A1" runat="server" href="~/Admin/AdminMenu">Back to Menu</a>
</asp:Content>
```

### 10.2.8 EditPGallery.aspx.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
using Griffin_Drywall.Models;
using System.Web.ModelBinding;

namespace Griffin_Drywall.Admin
{
    public partial class EditPGallery : System.Web.UI.Page
    {
        protected void Page_Load(object sender, EventArgs e)
        {
            if (!IsPostBack)
            {
                BindGrid(0);
            }
            lblMessage.Text = "";
        }

        void BindGrid(int pageIndex)
        {
            using (GriffinContext context = new
Griffin_Drywall.Models.GriffinContext())
            {
                int totalRecords = context.GImages.Count();
                int pageSize = 10;
                if (pageIndex < 0)
                { pageIndex = 0; }
                int startRow = pageIndex * pageSize;

                if (context.GImages.Count() > 0)
                {
                    gridPGallery.DataSource = context.GImages.OrderBy(x =>
x.GImageID).Skip(startRow).Take(pageSize).ToList();
                    gridPGallery.DataBind();

                    BindPager(totalRecords, pageIndex, pageSize);
                }
                else
                {
                    var obj = new List<GImage>();
                    obj.Add(new GImage());
                    // Bind the DataTable which contain a blank row to the
GridView
                    gridPGallery.DataSource = obj.ToList();
                    gridPGallery.DataBind();
                    int columnsCount = gridPGallery.Columns.Count;
                    gridPGallery.Rows[0].Cells.Clear();// clear all the cells in
the row
                    gridPGallery.Rows[0].Cells.Add(new TableCell()); //add a new
blank cell
                    gridPGallery.Rows[0].Cells[0].ColumnSpan = columnsCount; //set
the column span to the new added cell

                    //You can set the styles here
                    gridPGallery.Rows[0].Cells[0].HorizontalAlign =
HorizontalAlign.Center;
                    gridPGallery.Rows[0].Cells[0].ForeColor =
System.Drawing.Color.Red;
                    gridPGallery.Rows[0].Cells[0].Font.Bold = true;
                    //set No Results found to the new added cell
                    gridPGallery.Rows[0].Cells[0].Text = "NO RESULT FOUND!";
                }
            }
        }
        private void BindPager(int totalRecordCount, int currentPageIndex, int
pageSize)
        {
            double getPageCount = (double)((decimal)totalRecordCount /
(decimal)pageSize);
            int pageCount = (int)Math.Ceiling(getPageCount);
            List<ListItem> pages = new List<ListItem>();
            if (pageCount > 1)
            {
                pages.Add(new ListItem("FIRST", "1", currentPageIndex > 1));
                for (int i = 1; i <= pageCount; i++)
                {
                    pages.Add(new ListItem(i.ToString(), i.ToString(), i !=
currentPageIndex + 1));
                }
                pages.Add(new ListItem("LAST", pageCount.ToString(),
currentPageIndex < pageCount - 1));
            }

            rptPager.DataSource = pages;
            rptPager.DataBind();
        }

        protected void Page_Changed(object sender, EventArgs e)
        {
            int pageIndex = Convert.ToInt32(((sender as
LinkButton).CommandArgument));
            gridPGallery.PageIndex = pageIndex;
            BindGrid(pageIndex - 1);
        }

        protected void gridPGallery_RowDataBound(object sender,
GridViewRowEventArgs e)
        {
            DropDownList ddl = null;
            if (e.Row.RowType == DataControlRowType.Footer)
            {
                ddl = e.Row.FindControl("ddlProjectNew") as DropDownList;
            }
            if (e.Row.RowType == DataControlRowType.DataRow)
            {
                ddl = e.Row.FindControl("ddlProject") as DropDownList;
            }
            if (ddl != null)
            {
                using (GriffinContext context = new
Griffin_Drywall.Models.GriffinContext())
                {
                    ddl.DataSource = context.Projects.ToList();
                    ddl.DataTextField = "ProjectName";
                    ddl.DataValueField = "ProjectID";
                    ddl.DataBind();
                    ddl.Items.Insert(0, new ListItem(""));
                }
                if (e.Row.RowType == DataControlRowType.DataRow)
                {
                    ddl.SelectedValue =
((GImage)(e.Row.DataItem)).ProjectID.ToString();
```

```
                }
            }
        }

        protected void gridPGallery_RowCommand(object sender,
GridViewCommandEventArgs e)
        {
            if (e.CommandName == "InsertNew")
            {
                Boolean fileOK = false;
                String path = Server.MapPath("~/Images/Projects/PGallery/");
                GridViewRow row = gridPGallery.FooterRow;
                FileUpload flImagePath = row.FindControl("flImagePathNew") as
FileUpload;
                if (flImagePath.HasFile)
                {
                    String fileExtension =
System.IO.Path.GetExtension(flImagePath.FileName).ToLower();
                    String[] allowedExtensions = { ".gif", ".png", ".jpeg", ".jpg"
};
                    for (int i = 0; i < allowedExtensions.Length; i++)
                    {
                        if (fileExtension == allowedExtensions[i])
                        {
                            fileOK = true;
                        }
                    }
                }
                if (fileOK)
                {
                    try
                    {
                        // Save to Images folder.
                        flImagePath.PostedFile.SaveAs(path +
flImagePath.FileName);
                    }
                    catch (Exception ex)
                    {
                        lblMessage.Text = ex.Message;
                    }

                    DropDownList ddlProject = row.FindControl("ddlProjectNew") as
DropDownList;
                    if (ddlProject != null && flImagePath.FileName != null)
                    {
                        using (GriffinContext context = new
Griffin_Drywall.Models.GriffinContext())
                        {
                            GImage obj = new GImage();
                            obj.ProjectID =
Convert.ToInt32(ddlProject.SelectedValue);
                            obj.ImagePath = flImagePath.FileName;
                            context.GImages.Add(obj);
                            context.SaveChanges();
                            lblMessage.Text = "Added successfully.";
                            BindGrid(((GridView)sender).PageIndex - 1);
                        }
                    }
                    else
                    {
                        lblMessage.Text = "Unable to add new service to
database.";
                    }
                }
                else
                {
                    lblMessage.Text = "Unable to accept file type.";
                }
            }
        }

        protected void gridPGallery_RowEditing(object sender,
GridViewEditEventArgs e)
        {
            gridPGallery.EditIndex = e.NewEditIndex;
            BindGrid(((GridView)sender).PageIndex - 1);
        }
        protected void gridPGallery_RowCancelingEdit(object sender,
GridViewCancelEditEventArgs e)
        {
            gridPGallery.EditIndex = -1;
            BindGrid(((GridView)sender).PageIndex - 1);
        }

        protected void gridPGallery_RowUpdating(object sender,
GridViewUpdateEventArgs e)
        {
            Boolean fileOK = false;
            String path = Server.MapPath("~/Images/Projects/PGallery/");
            GridViewRow row = gridPGallery.Rows[e.RowIndex];
            FileUpload flImagePath = row.FindControl("flImagePath") as FileUpload;

                if (flImagePath.HasFile)
                {
                    String fileExtension =
System.IO.Path.GetExtension(flImagePath.FileName).ToLower();
                    String[] allowedExtensions = { ".gif", ".png", ".jpeg", ".jpg"
};
                    for (int i = 0; i < allowedExtensions.Length; i++)
                    {
                        if (fileExtension == allowedExtensions[i])
                        {
                            fileOK = true;
                        }
                    }
                }
                if (fileOK)
                {
                    try
                    {
                        // Save to Images folder.
                        flImagePath.PostedFile.SaveAs(path +
flImagePath.FileName);
                    }
                    catch (Exception ex)
                    {
                        lblMessage.Text = ex.Message;
                    }
                    DropDownList ddlProject = row.FindControl("ddlProject") as
DropDownList;
                    if (ddlProject != null && flImagePath.FileName != null)
                    {
                        int imageID =
Convert.ToInt32(gridPGallery.DataKeys[e.RowIndex].Value);
                        using (GriffinContext context = new
Griffin_Drywall.Models.GriffinContext())
                        {
                            GImage obj = context.GImages.First(x => x.GImageID ==
imageID);
                            obj.ProjectID =
Convert.ToInt32(ddlProject.SelectedValue);
                            obj.ImagePath = flImagePath.FileName;
                            context.SaveChanges();
                            lblMessage.Text = "Saved successfully.";
                            gridPGallery.EditIndex = -1;
                            BindGrid(((GridView)sender).PageIndex - 1);
                        }
                    }
                    else
                    {
                        lblMessage.Text = "Unable to add new image to gallery.";
                    }
                }
                else
                {
                    lblMessage.Text = "Unable to accept file type.";
                }
        }

        protected void gridPGallery_RowDeleting(object sender,
GridViewDeleteEventArgs e)
        {
            int imageID =
Convert.ToInt32(gridPGallery.DataKeys[e.RowIndex].Value);
            using (GriffinContext context = new
Griffin_Drywall.Models.GriffinContext())
            {
                GImage obj = context.GImages.First(x => x.GImageID == imageID);
                context.GImages.Remove(obj);
                context.SaveChanges();
                BindGrid(0);
                lblMessage.Text = "Deleted successfully.";
            }
        }
    }
}
```

### 10.2.9 EditProjects.aspx

```
<%@ Page Title="" Language="C#" MasterPageFile="~/Site.Master"
AutoEventWireup="true" CodeBehind="EditProjects.aspx.cs"
Inherits="Griffin_Drywall.Admin.EditProjects" %>
<asp:Content ID="Content1" ContentPlaceHolderID="HeadContent" runat="server">
</asp:Content>
<asp:Content ID="Content2" ContentPlaceHolderID="FeaturedContent" runat="server">
```

```
</asp:Content>
<asp:Content ID="Content3" ContentPlaceHolderID="MainContent" runat="server">
    <article class="projects">
       <h1>Manange Projects</h1>
        <p>

        </p>
   </article>
    <asp:Label ID="lblMessage" runat="server" Text="" ForeColor="Red"></asp:Label>
    <div style="vertical-align: top; left: auto; overflow: auto; width: auto;">
        <asp:GridView ID="gridProject" runat="server" AutoGenerateColumns="False" 
ShowFooter="True"
            CssClass="" OnRowCommand="gridProject_RowCommand"
            DataKeyNames="ProjectID" CellPadding="4" ForeColor="#333333"
            GridLines="None" OnRowCancelingEdit="gridProject_RowCancelingEdit"
            OnRowEditing="gridProject_RowEditing"
            OnRowUpdating="gridProject_RowUpdating"
            onrowdatabound="gridProject_RowDataBound"
            OnRowDeleting="gridProject_RowDeleting">
            <AlternatingRowStyle BackColor="White" />
            <Columns>
                <asp:TemplateField HeaderText="">
                    <ItemTemplate>
                        <asp:LinkButton ID="lnkEdit" runat="server" Text="" 
CommandName="Edit" ToolTip="Edit"
                            CommandArgument=''><img src="../Images/edit.png" 
/></asp:LinkButton>
                        <asp:LinkButton ID="lnkDelete" runat="server" 
Text="Delete" CommandName="Delete"
                            ToolTip="Delete" OnClientClick='return confirm("Are 
you sure you want to delete this entry?");'
                            CommandArgument=''><img src="../Images/delete.png" 
/></asp:LinkButton>
                    </ItemTemplate>
                    <EditItemTemplate>
                        <asp:LinkButton ID="lnkInsert" runat="server" Text="" 
ValidationGroup="editGrp" CommandName="Update" ToolTip="Save"
                            CommandArgument=''><img src="../Images/save.png" 
/></asp:LinkButton>
                        <asp:LinkButton ID="lnkCancel" runat="server" Text="" 
CommandName="Cancel" ToolTip="Cancel"
                            CommandArgument=''><img src="../Images/refresh.png" 
/></asp:LinkButton>
                    </EditItemTemplate>
                    <FooterTemplate>
                        <asp:LinkButton ID="lnkInsert" runat="server" Text="" 
ValidationGroup="newGrp" CommandName="InsertNew" ToolTip="Add New Entry"
                            CommandArgument=''><img src="../Images/add.png" 
/></asp:LinkButton>
                        <asp:LinkButton ID="lnkCancel" runat="server" Text="" 
CommandName="CancelNew" ToolTip="Cancel"
                            CommandArgument=''><img src="../Images/refresh.png" 
/></asp:LinkButton>
                    </FooterTemplate>
                </asp:TemplateField>
                <asp:TemplateField HeaderText="Project Name">
                    <EditItemTemplate>
```

```
                        <asp:TextBox ID="txtProjectName" runat="server"
Width="150" Text='<%# Bind("ProjectName") %>' CssClass=""
MaxLength="100"></asp:TextBox>
                        <asp:RequiredFieldValidator ID="valProjectName"
runat="server" ControlToValidate="txtProjectName"
                            Display="Dynamic" ErrorMessage="Project Name is
required." ForeColor="Red" SetFocusOnError="True"

ValidationGroup="editGrp">*</asp:RequiredFieldValidator>
                    </EditItemTemplate>
                    <ItemTemplate>
                        <asp:Label ID="lblProjectName" runat="server" Width="150"
Text='<%# Bind("ProjectName") %>'></asp:Label>
                    </ItemTemplate>
                    <FooterTemplate>
                        <asp:TextBox ID="txtProjectNameNew" runat="server"
Width="150" CssClass="" MaxLength="100"></asp:TextBox>
                        <asp:RequiredFieldValidator ID="valProjectNameNew"
runat="server" ControlToValidate="txtProjectNameNew"
                            Display="Dynamic" ErrorMessage="Project Name is
required." ForeColor="Red" SetFocusOnError="True"

ValidationGroup="newGrp">*</asp:RequiredFieldValidator>
                    </FooterTemplate>
                </asp:TemplateField>
                <asp:TemplateField HeaderText="Description">
                    <EditItemTemplate>
                        <asp:TextBox ID="txtDescription" runat="server" Text='<%#
Bind("Description") %>' CssClass="" TextMode="multiline"
MaxLength="10000"></asp:TextBox>
                        <asp:RequiredFieldValidator ID="valDescription"
runat="server" ControlToValidate="txtDescription"
                            Display="Dynamic" ErrorMessage="Description is
required." ForeColor="Red" SetFocusOnError="True"

ValidationGroup="editGrp">*</asp:RequiredFieldValidator>
                    </EditItemTemplate>
                    <ItemTemplate>
                        <asp:Label ID="lblDescription" runat="server" Style="text-
overflow: ellipsis; overflow: hidden" Width="300px" Height="50px" Text='<%#
Bind("Description") %>'></asp:Label>
                    </ItemTemplate>
                    <FooterTemplate>
                        <asp:TextBox ID="txtDescriptionNew" runat="server"
CssClass="" Style="text-overflow: ellipsis; overflow: hidden; margin-left: 5px"
Width="280px"  Height="22px" TextMode="multiline" MaxLength="10000"></asp:TextBox>
                        <asp:RequiredFieldValidator ID="valDescriptionNew"
runat="server" ControlToValidate="txtDescriptionNew"
                            Display="Dynamic" ErrorMessage="Description is
required." ForeColor="Red" SetFocusOnError="True"

ValidationGroup="newGrp">*</asp:RequiredFieldValidator>
                    </FooterTemplate>
                </asp:TemplateField>
                <asp:TemplateField HeaderText="Image Path">
                    <EditItemTemplate>
                        <asp:FileUpload ID="flImagePath" runat="server" />
                    </EditItemTemplate>
                    <ItemTemplate>
                        <asp:Label ID="lblImagePath" runat="server" Width="150"
Text='<%# Bind("ImagePath") %>'></asp:Label>
                    </ItemTemplate>
                    <FooterTemplate>
                        <asp:FileUpload ID="flImagePathNew" runat="server" />
                        <asp:RequiredFieldValidator ID="valImagePathNew"
runat="server" ErrorMessage="Image path required."  ForeColor="Red"
ControlToValidate="flImagePathNew" SetFocusOnError="true" Display="Dynamic"
ValidationGroup="newGrp">*</asp:RequiredFieldValidator>
                    </FooterTemplate>
                </asp:TemplateField>
                <asp:TemplateField HeaderText="Company">
                    <EditItemTemplate>
                        <asp:DropDownList ID="ddlCompany" runat="server">
                        </asp:DropDownList>
                        <asp:RequiredFieldValidator ID="valCompany" runat="server"
ControlToValidate="ddlCompany"
                            Display="Dynamic" ErrorMessage="Company is required."
ForeColor="Red" SetFocusOnError="True"

ValidationGroup="editGrp">*</asp:RequiredFieldValidator>
                    </EditItemTemplate>
                    <ItemTemplate>
                        <asp:Label ID="lblCompany" runat="server" Text='<%#
Bind("Company.CompanyName") %>'></asp:Label>
                    </ItemTemplate>
                    <FooterTemplate>
                        <asp:DropDownList ID="ddlCompanyNew" runat="server">
                        </asp:DropDownList>
                        <asp:RequiredFieldValidator ID="valCompanyNew"
runat="server" ControlToValidate="ddlCompanyNew"
                            Display="Dynamic" ErrorMessage="Company is required."
ForeColor="Red" SetFocusOnError="True"

ValidationGroup="newGrp">*</asp:RequiredFieldValidator>
                    </FooterTemplate>
                </asp:TemplateField>
            </Columns>
            <EditRowStyle BackColor="#2461BF" />
            <FooterStyle BackColor="#507CD1" Font-Bold="True" ForeColor="White" />
            <HeaderStyle BackColor="#507CD1" Font-Bold="True" ForeColor="White" />
            <PagerStyle BackColor="#2461BF" ForeColor="White"
HorizontalAlign="Center" />
            <RowStyle BackColor="#EFF3FB" />
            <SelectedRowStyle BackColor="#D1DDF1" Font-Bold="True"
ForeColor="#333333" />
            <SortedAscendingCellStyle BackColor="#F5F7FB" />
            <SortedAscendingHeaderStyle BackColor="#6D95E1" />
            <SortedDescendingCellStyle BackColor="#E9EBEF" />
            <SortedDescendingHeaderStyle BackColor="#4870BE" />
        </asp:GridView>
        <asp:Repeater ID="rptPager" runat="server">
            <ItemTemplate>
                <asp:LinkButton ID="lnkPage" runat="server"
                    Text='<%#Eval("Text") %>'
                    CommandArgument='<%#Eval("Value") %>'
                    Enabled='<%#Eval("Enabled") %>'
                    OnClick="Page_Changed"
                    ForeColor="#267CB2"
                    Font-Bold="true" />
            </ItemTemplate>
        </asp:Repeater>
    </div>
    <asp:ValidationSummary ID="ValidationSummary1" ValidationGroup="newGrp"
class="validation-summary-errors" runat="server" />
    <asp:ValidationSummary ID="ValidationSummary2" ValidationGroup="editGrp"
class="validation-summary-errors" runat="server" />
    <br />
    <a id="A1" runat="server" href="~/Admin/AdminMenu">Back to Menu</a>
</asp:Content>
```

### 10.2.10 EditProjects.aspx.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
using Griffin_Drywall.Models;
using System.Web.ModelBinding;

namespace Griffin_Drywall.Admin
{
    public partial class EditProjects : System.Web.UI.Page
    {
        protected void Page_Load(object sender, EventArgs e)
        {
            if (!IsPostBack)
            {
                BindGrid(0);
            }
            lblMessage.Text = "";
        }

        void BindGrid(int pageIndex)
        {
            using (GriffinContext context = new
Griffin_Drywall.Models.GriffinContext())
            {
                int totalRecords = context.Projects.Count();
                int pageSize = 5;
                if (pageIndex < 0)
                { pageIndex = 0; }
                int startRow = pageIndex * pageSize;

                if (context.Projects.Count() > 0)
                {
                    gridProject.DataSource = context.Projects.OrderBy(x =>
x.ProjectID).Skip(startRow).Take(pageSize).ToList();
                    gridProject.DataBind();

                    BindPager(totalRecords, pageIndex, pageSize);
                }
                else
                {
                    var obj = new List<Project>();
                    obj.Add(new Project());
                    // Bind the DataTable which contain a blank row to the
GridView
                    gridProject.DataSource = obj.ToList();
                    gridProject.DataBind();
                    int columnsCount = gridProject.Columns.Count;
                    gridProject.Rows[0].Cells.Clear();// clear all the cells in
the row
                    gridProject.Rows[0].Cells.Add(new TableCell()); //add a new
blank cell
                    gridProject.Rows[0].Cells[0].ColumnSpan = columnsCount; //set
the column span to the new added cell

                    //You can set the styles here
                    gridProject.Rows[0].Cells[0].HorizontalAlign =
HorizontalAlign.Center;
                    gridProject.Rows[0].Cells[0].ForeColor =
System.Drawing.Color.Red;
                    gridProject.Rows[0].Cells[0].Font.Bold = true;
                    //set No Results found to the new added cell
                    gridProject.Rows[0].Cells[0].Text = "NO RESULT FOUND!";
                }
            }
        }

        private void BindPager(int totalRecordCount, int currentPageIndex, int
pageSize)
        {
            double getPageCount = (double)((decimal)totalRecordCount /
(decimal)pageSize);
            int pageCount = (int)Math.Ceiling(getPageCount);
            List<ListItem> pages = new List<ListItem>();
            if (pageCount > 1)
            {
                pages.Add(new ListItem("FIRST", "1", currentPageIndex > 1));
                for (int i = 1; i <= pageCount; i++)
                {
                    pages.Add(new ListItem(i.ToString(), i.ToString(), i !=
currentPageIndex + 1));
                }
                pages.Add(new ListItem("LAST", pageCount.ToString(),
currentPageIndex < pageCount - 1));
            }

            rptPager.DataSource = pages;
            rptPager.DataBind();
        }

        protected void Page_Changed(object sender, EventArgs e)
        {
            int pageIndex = Convert.ToInt32(((sender as
LinkButton).CommandArgument));
            gridProject.PageIndex = pageIndex;
            BindGrid(pageIndex - 1);
        }
        protected void gridProject_RowDataBound(object sender,
GridViewRowEventArgs e)
        {
            DropDownList ddl = null;
            if (e.Row.RowType == DataControlRowType.Footer)
            {
                ddl = e.Row.FindControl("ddlCompanyNew") as DropDownList;
            }
            if (e.Row.RowType == DataControlRowType.DataRow)
            {
                ddl = e.Row.FindControl("ddlCompany") as DropDownList;
            }
            if (ddl != null)
            {
                using (GriffinContext context = new
Griffin_Drywall.Models.GriffinContext())
                {
                    ddl.DataSource = context.Companies.ToList();
                    ddl.DataTextField = "CompanyName";
                    ddl.DataValueField = "CompanyID";
                    ddl.DataBind();
                    ddl.Items.Insert(0, new ListItem(""));
                }
                if (e.Row.RowType == DataControlRowType.DataRow)
                {
                    ddl.SelectedValue =
((Project)(e.Row.DataItem)).CompanyID.ToString();
                }
            }
        }

        protected void gridProject_RowCommand(object sender,
GridViewCommandEventArgs e)
        {
            if (e.CommandName == "InsertNew")
            {
                Boolean fileOK = false;
                String path = Server.MapPath("~/Images/Projects/Thumbs/");
                GridViewRow row = gridProject.FooterRow;
                FileUpload flImagePath = row.FindControl("flImagePathNew") as
FileUpload;
                if (flImagePath.HasFile)
                {
                    String fileExtension =
System.IO.Path.GetExtension(flImagePath.FileName).ToLower();
                    String[] allowedExtensions = { ".gif", ".png", ".jpeg", ".jpg"
};

                    for (int i = 0; i < allowedExtensions.Length; i++)
                    {
                        if (fileExtension == allowedExtensions[i])
                        {
                            fileOK = true;
                        }
                    }
                }
                if (fileOK)
                {
                    try
                    {
                        // Save to Images folder.
                        flImagePath.PostedFile.SaveAs(path +
flImagePath.FileName);
                    }
                    catch (Exception ex)
                    {
                        lblMessage.Text = ex.Message;
                    }

                    TextBox txtProjectName = row.FindControl("txtProjectNameNew")
as TextBox;
                    TextBox txtDescription = row.FindControl("txtDescriptionNew")
as TextBox;
                    DropDownList ddlCompany = row.FindControl("ddlCompanyNew") as
DropDownList;
                    if (txtProjectName != null && txtDescription != null &&
flImagePath.FileName != null && ddlCompany != null)
                    {
                        using (GriffinContext context = new
Griffin_Drywall.Models.GriffinContext())
                        {
                            Project obj = new Project();
                            obj.ProjectName = txtProjectName.Text;
                            obj.Description = txtDescription.Text;
                            obj.ImagePath = flImagePath.FileName;
                            obj.CompanyID =
Convert.ToInt32(ddlCompany.SelectedValue);
                            context.Projects.Add(obj);
                            context.SaveChanges();
                            lblMessage.Text = "Added successfully.";
                            BindGrid(((GridView)sender).PageIndex - 1);
                        }
                    }
                    else
                    {
                        lblMessage.Text = "Unable to add new project to
database.";
                    }
                }
                else
                {
                    lblMessage.Text = "Unable to accept file type.";
                }
            }
        }

        protected void gridProject_RowEditing(object sender, GridViewEditEventArgs
e)
        {
            gridProject.EditIndex = e.NewEditIndex;
            BindGrid(((GridView)sender).PageIndex - 1);
        }
        protected void gridProject_RowCancelingEdit(object sender,
GridViewCancelEditEventArgs e)
        {
            gridProject.EditIndex = -1;
            BindGrid(((GridView)sender).PageIndex - 1);
        }

        protected void gridProject_RowUpdating(object sender,
GridViewUpdateEventArgs e)
        {
            Boolean fileOK = false;
            String path = Server.MapPath("~/Images/Projects/Thumbs/");
            GridViewRow row = gridProject.Rows[e.RowIndex];
            FileUpload flImagePath = row.FindControl("flImagePath") as FileUpload;
            if (flImagePath.HasFile)
            {
                if (flImagePath.HasFile)
                {
                    String fileExtension =
System.IO.Path.GetExtension(flImagePath.FileName).ToLower();
                    String[] allowedExtensions = { ".gif", ".png", ".jpeg", ".jpg"
};
                    for (int i = 0; i < allowedExtensions.Length; i++)
                    {
                        if (fileExtension == allowedExtensions[i])
                        {
                            fileOK = true;
                        }
                    }
                }
                if (fileOK)
                {
                    try
                    {
                        // Save to Images folder.
                        flImagePath.PostedFile.SaveAs(path +
flImagePath.FileName);
                    }
                    catch (Exception ex)
                    {
                        lblMessage.Text = ex.Message;
                    }
                    TextBox txtProjectName = row.FindControl("txtProjectName") as
TextBox;
                    TextBox txtDescription = row.FindControl("txtDescription") as
TextBox;
                    DropDownList ddlCompany = row.FindControl("ddlCompany") as
DropDownList;
                    if (txtProjectName != null && txtDescription != null &&
flImagePath.FileName != null && ddlCompany != null)
                    {
                        int projectID =
Convert.ToInt32(gridProject.DataKeys[e.RowIndex].Value);
                        using (GriffinContext context = new
Griffin_Drywall.Models.GriffinContext())
                        {
                            Project obj = context.Projects.First(x => x.ProjectID
== projectID);
                            obj.ProjectName = txtProjectName.Text;
                            obj.Description = txtDescription.Text;
                            obj.ImagePath = flImagePath.FileName;
                            obj.CompanyID =
Convert.ToInt32(ddlCompany.SelectedValue);
```

```
                            context.SaveChanges();
                            lblMessage.Text = "Saved successfully.";
                            gridProject.EditIndex = -1;
                            BindGrid(((GridView)sender).PageIndex - 1);
                        }
                    }
                    else
                    {
                        lblMessage.Text = "Unable to edit project in database.";
                    }
                }
                else
                {
                    lblMessage.Text = "Unable to accept file type.";
                }
            }
            else
            {
                TextBox txtProjectName = row.FindControl("txtProjectName") as
TextBox;
                TextBox txtDescription = row.FindControl("txtDescription") as
TextBox;
                DropDownList ddlCompany = row.FindControl("ddlCompany") as
DropDownList;
                if (txtProjectName != null && txtDescription != null && ddlCompany
!= null)
                {
                    int projectID =
Convert.ToInt32(gridProject.DataKeys[e.RowIndex].Value);
                    using (GriffinContext context = new
Griffin_Drywall.Models.GriffinContext())
                    {
                        Project obj = context.Projects.First(x => x.ProjectID ==
projectID);
                        obj.ProjectName = txtProjectName.Text;
                        obj.Description = txtDescription.Text;
                        obj.CompanyID = Convert.ToInt32(ddlCompany.SelectedValue);
                        context.SaveChanges();
                        lblMessage.Text = "Saved successfully.";
                        gridProject.EditIndex = -1;
                        BindGrid(((GridView)sender).PageIndex - 1);
                    }
                }
                else
                {
                    lblMessage.Text = "Unable to edit project in database.";
                }
            }
        }

        protected void gridProject_RowDeleting(object sender,
GridViewDeleteEventArgs e)
        {
            int projectID =
Convert.ToInt32(gridProject.DataKeys[e.RowIndex].Value);
            using (GriffinContext context = new
Griffin_Drywall.Models.GriffinContext())
            {
                Project obj = context.Projects.First(x => x.ProjectID ==
projectID);
                context.Projects.Remove(obj);
                context.SaveChanges();
                BindGrid(0);
                lblMessage.Text = "Deleted successfully.";
            }
        }
    }
}
```

### 10.2.11 EditService.aspx

```
<%@ Page Title="" Language="C#" MasterPageFile="~/Site.Master"
AutoEventWireup="true" CodeBehind="EditService.aspx.cs"
Inherits="Griffin_Drywall.Admin.EditService" %>
<asp:Content ID="Content1" ContentPlaceHolderID="HeadContent" runat="server">
</asp:Content>
<asp:Content ID="Content2" ContentPlaceHolderID="FeaturedContent" runat="server">
</asp:Content>
<asp:Content ID="Content3" ContentPlaceHolderID="MainContent" runat="server">
    <article class="projects">
       <h1>Manange Services</h1>
        <p>

        </p>
   </article>
    <asp:Label ID="lblMessage" runat="server" Text="" ForeColor="Red"></asp:Label>
    <div style="vertical-align: top; left: auto; overflow: auto; width: auto;">
        <asp:GridView ID="gridService" runat="server" AutoGenerateColumns="False"
ShowFooter="True"
            CssClass="" OnRowCommand="gridService_RowCommand"
            DataKeyNames="ServiceID" CellPadding="4" ForeColor="#333333"
            GridLines="None" OnRowCancelingEdit="gridService_RowCancelingEdit"
            OnRowEditing="gridService_RowEditing"
            OnRowUpdating="gridService_RowUpdating"
            OnRowDeleting="gridService_RowDeleting">
            <AlternatingRowStyle BackColor="White" />
            <Columns>
                <asp:TemplateField HeaderText="">
                    <ItemTemplate>
                        <asp:LinkButton ID="lnkEdit" runat="server" Text=""
CommandName="Edit" ToolTip="Edit"
                            CommandArgument=''><img src="../Images/edit.png"
/></asp:LinkButton>
                        <asp:LinkButton ID="lnkDelete" runat="server"
Text="Delete" CommandName="Delete"
                            ToolTip="Delete" OnClientClick='return confirm("Are
you sure you want to delete this entry?");'
                            CommandArgument=''><img src="../Images/delete.png"
/></asp:LinkButton>
                    </ItemTemplate>
                    <EditItemTemplate>
                        <asp:LinkButton ID="lnkInsert" runat="server" Text=""
ValidationGroup="editGrp" CommandName="Update" ToolTip="Save"
                            CommandArgument=''><img src="../Images/save.png"
/></asp:LinkButton>
                        <asp:LinkButton ID="lnkCancel" runat="server" Text=""
CommandName="Cancel" ToolTip="Cancel"
```

```
                            CommandArgument=''><img src="../Images/refresh.png"
/></asp:LinkButton>
                    </EditItemTemplate>
                    <FooterTemplate>
                        <asp:LinkButton ID="lnkInsert" runat="server" Text=""
ValidationGroup="newGrp" CommandName="InsertNew" ToolTip="Add New Entry"
                            CommandArgument=''><img src="../Images/add.png"
/></asp:LinkButton>
                        <asp:LinkButton ID="lnkCancel" runat="server" Text=""
CommandName="CancelNew" ToolTip="Cancel"
                            CommandArgument=''><img src="../Images/refresh.png"
/></asp:LinkButton>
                    </FooterTemplate>
                </asp:TemplateField>
                <asp:TemplateField HeaderText="Service Name">
                    <EditItemTemplate>
                        <asp:TextBox ID="txtServiceName" runat="server"
Width="150" Text='<%# Bind("ServiceName") %>' CssClass=""
MaxLength="100"></asp:TextBox>
                        <asp:RequiredFieldValidator ID="valServiceName"
runat="server" ControlToValidate="txtServiceName"
                            Display="Dynamic" ErrorMessage="Service Name is
required." ForeColor="Red" SetFocusOnError="True"

ValidationGroup="editGrp">*</asp:RequiredFieldValidator>
                    </EditItemTemplate>
                    <ItemTemplate>
                        <asp:Label ID="lblServiceName" runat="server" Width="150"
Text='<%# Bind("ServiceName") %>'></asp:Label>
                    </ItemTemplate>
                    <FooterTemplate>
                        <asp:TextBox ID="txtServiceNameNew" runat="server"
Width="150" CssClass="" MaxLength="100"></asp:TextBox>
                        <asp:RequiredFieldValidator ID="valServiceNameNew"
runat="server" ControlToValidate="txtServiceNameNew"
                            Display="Dynamic" ErrorMessage="Service Name is
required." ForeColor="Red" SetFocusOnError="True"

ValidationGroup="newGrp">*</asp:RequiredFieldValidator>
                    </FooterTemplate>
                </asp:TemplateField>
                <asp:TemplateField HeaderText="Description">
                    <EditItemTemplate>
                        <asp:TextBox ID="txtDescription" runat="server" Text='<%#
Bind("Description") %>' CssClass="" TextMode="multiline"
MaxLength="10000"></asp:TextBox>
                        <asp:RequiredFieldValidator ID="valDescription"
runat="server" ControlToValidate="txtDescription"
                            Display="Dynamic" ErrorMessage="Description is
required." ForeColor="Red" SetFocusOnError="True"

ValidationGroup="editGrp">*</asp:RequiredFieldValidator>
                    </EditItemTemplate>
                    <ItemTemplate>
                        <asp:Label ID="lblDescription" runat="server" Style="text-
overflow: ellipsis; overflow: hidden" Width="300px" Height="50px" Text='<%#
Bind("Description") %>'></asp:Label>
                    </ItemTemplate>
                    <FooterTemplate>
                        <asp:TextBox ID="txtDescriptionNew" runat="server"
CssClass="" Style="text-overflow: ellipsis; overflow: hidden; margin-left: 5px"
Width="280px"  Height="22px" TextMode="multiline" MaxLength="10000"></asp:TextBox>
                        <asp:RequiredFieldValidator ID="valDescriptionNew"
runat="server" ControlToValidate="txtDescriptionNew"
                            Display="Dynamic" ErrorMessage="Description is
required." ForeColor="Red" SetFocusOnError="True"

ValidationGroup="newGrp">*</asp:RequiredFieldValidator>
                    </FooterTemplate>
                </asp:TemplateField>
                <asp:TemplateField HeaderText="Description 2nd">
                    <EditItemTemplate>
                        <asp:TextBox ID="txtDescription2" runat="server" Text='<%#
Bind("Description2") %>' CssClass="" TextMode="multiline"
MaxLength="10000"></asp:TextBox>
                    </EditItemTemplate>
                    <ItemTemplate>
                        <asp:Label ID="lblDescription2" runat="server"
Style="text-overflow: ellipsis; overflow: hidden" Width="300px" Height="50px"
Text='<%# Bind("Description2") %>'></asp:Label>
                    </ItemTemplate>
                    <FooterTemplate>
                        <asp:TextBox ID="txtDescription2New" runat="server"
Style="text-overflow: ellipsis; overflow: hidden" Width="280px" Height="22px"
CssClass="" TextMode="multiline" MaxLength="10000"></asp:TextBox>
                    </FooterTemplate>
                </asp:TemplateField>
                <asp:TemplateField HeaderText="Description 3rd">
                    <EditItemTemplate>
                        <asp:TextBox ID="txtDescription3" runat="server" Text='<%#
Bind("Description3") %>' CssClass="" TextMode="multiline"
MaxLength="10000"></asp:TextBox>
                    </EditItemTemplate>
                    <ItemTemplate>
                        <asp:Label ID="lblDescription3" runat="server"
Style="text-overflow: ellipsis; overflow: hidden" Width="300px" Height="50px"
Text='<%# Bind("Description3") %>'></asp:Label>
                    </ItemTemplate>
                    <FooterTemplate>
                        <asp:TextBox ID="txtDescription3New" runat="server"
Style="text-overflow: ellipsis; overflow: hidden" Width="280px" Height="22px"
CssClass="" TextMode="multiline" MaxLength="10000"></asp:TextBox>
                    </FooterTemplate>
                </asp:TemplateField>
                <asp:TemplateField HeaderText="Image Path">
                    <EditItemTemplate>
                        <asp:FileUpload ID="flImagePath" runat="server" />
                    </EditItemTemplate>
                    <ItemTemplate>
                        <asp:Label ID="lblImagePath" runat="server" Width="150"
Text='<%# Bind("ImagePath") %>'></asp:Label>
                    </ItemTemplate>
                    <FooterTemplate>
                        <asp:FileUpload ID="flImagePathNew" runat="server" />
                        <asp:RequiredFieldValidator ID="valImagePathNew"
runat="server" ErrorMessage="Image path required."  ForeColor="Red"
```

```
ControlToValidate="flImagePathNew" SetFocusOnError="true" Display="Dynamic"
ValidationGroup="newGrp">*</asp:RequiredFieldValidator>
                    </FooterTemplate>
                </asp:TemplateField>
            </Columns>
            <EditRowStyle BackColor="#2461BF" />
            <FooterStyle BackColor="#507CD1" Font-Bold="True" ForeColor="White" />
            <HeaderStyle BackColor="#507CD1" Font-Bold="True" ForeColor="White" />
            <PagerStyle BackColor="#2461BF" ForeColor="White"
HorizontalAlign="Center" />
            <RowStyle BackColor="#EFF3FB" />
            <SelectedRowStyle BackColor="#D1DDF1" Font-Bold="True"
ForeColor="#333333" />
            <SortedAscendingCellStyle BackColor="#F5F7FB" />
            <SortedAscendingHeaderStyle BackColor="#6D95E1" />
            <SortedDescendingCellStyle BackColor="#E9EBEF" />
            <SortedDescendingHeaderStyle BackColor="#4870BE" />
        </asp:GridView>
        <asp:Repeater ID="rptPager" runat="server">
            <ItemTemplate>
                <asp:LinkButton ID="lnkPage" runat="server"
                    Text='<%#Eval("Text") %>'
                    CommandArgument='<%#Eval("Value") %>'
                    Enabled='<%#Eval("Enabled") %>'
                    OnClick="Page_Changed"
                    ForeColor="#267CB2"
                    Font-Bold="true" />
            </ItemTemplate>
        </asp:Repeater>
    </div>
    <asp:ValidationSummary ID="ValidationSummary1" ValidationGroup="newGrp"
class="validation-summary-errors" runat="server" />
    <asp:ValidationSummary ID="ValidationSummary2" ValidationGroup="editGrp"
class="validation-summary-errors" runat="server" />
    <br />
    <a id="A1" runat="server" href="~/Admin/AdminMenu">Back to Menu</a>
</asp:Content>
```

### 10.2.12 EditService.aspx.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
using Griffin_Drywall.Models;
using System.Web.ModelBinding;

namespace Griffin_Drywall.Admin
{
    public partial class EditService : System.Web.UI.Page
    {
        protected void Page_Load(object sender, EventArgs e)
        {
            if (!IsPostBack)
            {
                BindGrid(0);
            }
            lblMessage.Text = "";
        }

        void BindGrid(int pageIndex)
        {
            using (GriffinContext context = new
Griffin_Drywall.Models.GriffinContext())
            {
                int totalRecords = context.Services.Count();
                int pageSize = 5;
                if (pageIndex < 0)
                { pageIndex = 0; }
                int startRow = pageIndex * pageSize;

                if (context.Services.Count() > 0)
                {
                    gridService.DataSource = context.Services.OrderBy(x =>
x.ServiceID).Skip(startRow).Take(pageSize).ToList();
                    gridService.DataBind();

                    BindPager(totalRecords, pageIndex, pageSize);
                }
                else
                {
                    var obj = new List<Service>();
                    obj.Add(new Service());
                    // Bind the DataTable which contain a blank row to the
GridView
                    gridService.DataSource = obj.ToList();
                    gridService.DataBind();
                    int columnsCount = gridService.Columns.Count;
                    gridService.Rows[0].Cells.Clear();// clear all the cells in
the row
                    gridService.Rows[0].Cells.Add(new TableCell()); //add a new
blank cell
                    gridService.Rows[0].Cells[0].ColumnSpan = columnsCount; //set
the column span to the new added cell

                    //You can set the styles here
                    gridService.Rows[0].Cells[0].HorizontalAlign =
HorizontalAlign.Center;
                    gridService.Rows[0].Cells[0].ForeColor =
System.Drawing.Color.Red;
                    gridService.Rows[0].Cells[0].Font.Bold = true;
                    //set No Results found to the new added cell
                    gridService.Rows[0].Cells[0].Text = "NO RESULT FOUND!";
                }
            }
        }

        private void BindPager(int totalRecordCount, int currentPageIndex, int
pageSize)
        {
            double getPageCount = (double)((decimal)totalRecordCount /
(decimal)pageSize);
            int pageCount = (int)Math.Ceiling(getPageCount);
            List<ListItem> pages = new List<ListItem>();
            if (pageCount > 1)
            {
                pages.Add(new ListItem("FIRST", "1", currentPageIndex > 1));
                for (int i = 1; i <= pageCount; i++)
                {
                    pages.Add(new ListItem(i.ToString(), i.ToString(), i !=
currentPageIndex + 1));
                }
                pages.Add(new ListItem("LAST", pageCount.ToString(),
currentPageIndex < pageCount - 1));
            }

            rptPager.DataSource = pages;
            rptPager.DataBind();
        }

        protected void Page_Changed(object sender, EventArgs e)
        {
            int pageIndex = Convert.ToInt32(((sender as
LinkButton).CommandArgument));
            gridService.PageIndex = pageIndex;
            BindGrid(pageIndex - 1);
        }

        protected void gridService_RowCommand(object sender,
GridViewCommandEventArgs e)
        {
            if (e.CommandName == "InsertNew")
            {
                Boolean fileOK = false;
                String path = Server.MapPath("~/Images/Services/");
                GridViewRow row = gridService.FooterRow;
                FileUpload flImagePath = row.FindControl("flImagePathNew") as
FileUpload;
                if (flImagePath.HasFile)
                {
                    String fileExtension =
System.IO.Path.GetExtension(flImagePath.FileName).ToLower();
                    String[] allowedExtensions = { ".gif", ".png", ".jpeg", ".jpg"
};
                    for (int i = 0; i < allowedExtensions.Length; i++)
                    {
                        if (fileExtension == allowedExtensions[i])
                        {
                            fileOK = true;
                        }
                    }
                }
                if (fileOK)
                {
                    try
                    {
                        // Save to Images folder.
                        flImagePath.PostedFile.SaveAs(path +
flImagePath.FileName);
                    }
                    catch (Exception ex)
                    {
                        lblMessage.Text = ex.Message;
                    }

                    TextBox txtServiceName = row.FindControl("txtServiceNameNew")
as TextBox;
                    TextBox txtDescription = row.FindControl("txtDescriptionNew")
as TextBox;
                    TextBox txtDescription2 =
row.FindControl("txtDescription2New") as TextBox;
                    TextBox txtDescription3 =
row.FindControl("txtDescription3New") as TextBox;
                    if (txtServiceName != null && txtDescription != null &&
flImagePath.FileName != null)
                    {
                        using (GriffinContext context = new
Griffin_Drywall.Models.GriffinContext())
                        {
                            Service obj = new Service();
                            obj.ServiceName = txtServiceName.Text;
                            obj.Description = txtDescription.Text;
                            obj.Description2 = txtDescription2.Text;
                            obj.Description3 = txtDescription3.Text;
                            obj.ImagePath = flImagePath.FileName;
                            context.Services.Add(obj);
                            context.SaveChanges();
                            lblMessage.Text = "Added successfully.";
                            BindGrid(((GridView)sender).PageIndex - 1);
                        }
                    }
                    else
                    {
                        lblMessage.Text = "Unable to add new service to
database.";
                    }
                }
                else
                {
                    lblMessage.Text = "Unable to accept file type.";
                }
            }
        }

        protected void gridService_RowEditing(object sender, GridViewEditEventArgs
e)
        {
            gridService.EditIndex = e.NewEditIndex;
            BindGrid(((GridView)sender).PageIndex - 1);
        }
        protected void gridService_RowCancelingEdit(object sender,
GridViewCancelEditEventArgs e)
        {
            gridService.EditIndex = -1;
            BindGrid(((GridView)sender).PageIndex - 1);
        }

        protected void gridService_RowUpdating(object sender,
GridViewUpdateEventArgs e)
        {
            Boolean fileOK = false;
            String path = Server.MapPath("~/Images/Services/");
            GridViewRow row = gridService.Rows[e.RowIndex];
            FileUpload flImagePath = row.FindControl("flImagePath") as FileUpload;
            if (flImagePath.HasFile)
            {
                if (flImagePath.HasFile)
                {
                    String fileExtension =
System.IO.Path.GetExtension(flImagePath.FileName).ToLower();
                    String[] allowedExtensions = { ".gif", ".png", ".jpeg", ".jpg"
};
                    for (int i = 0; i < allowedExtensions.Length; i++)
                    {
                        if (fileExtension == allowedExtensions[i])
                        {
                            fileOK = true;
                        }
                    }
                }
                if (fileOK)
                {
                    try
                    {
                        // Save to Images folder.
                        flImagePath.PostedFile.SaveAs(path +
flImagePath.FileName);
                    }
                    catch (Exception ex)
                    {
                        lblMessage.Text = ex.Message;
                    }
                    TextBox txtServiceName = row.FindControl("txtServiceName") as
TextBox;
                    TextBox txtDescription = row.FindControl("txtDescription") as
TextBox;
                    TextBox txtDescription2 = row.FindControl("txtDescription2")
as TextBox;
                    TextBox txtDescription3 = row.FindControl("txtDescription3")
as TextBox;
                    if (txtServiceName != null && txtDescription != null &&
flImagePath.FileName != null)
                    {
                        int serviceID =
Convert.ToInt32(gridService.DataKeys[e.RowIndex].Value);
                        using (GriffinContext context = new
Griffin_Drywall.Models.GriffinContext())
                        {
                            Service obj = context.Services.First(x => x.ServiceID
== serviceID);

                            obj.ServiceName = txtServiceName.Text;
                            obj.Description = txtDescription.Text;
                            obj.Description2 = txtDescription2.Text;
                            obj.Description3 = txtDescription3.Text;
                            obj.ImagePath = flImagePath.FileName;
                            context.SaveChanges();
                            lblMessage.Text = "Saved successfully.";
                            gridService.EditIndex = -1;
                            BindGrid(((GridView)sender).PageIndex - 1);
                        }
                    }
                    else
                    {
                        lblMessage.Text = "Unable to add new service to 
database.";
                    }
                }
                else
                {
                    lblMessage.Text = "Unable to accept file type.";
                }
            }
            else
            {
                TextBox txtServiceName = row.FindControl("txtServiceName") as 
TextBox;
                TextBox txtDescription = row.FindControl("txtDescription") as 
TextBox;
                TextBox txtDescription2 = row.FindControl("txtDescription2") as 
TextBox;
                TextBox txtDescription3 = row.FindControl("txtDescription3") as 
TextBox;
                if (txtServiceName != null && txtDescription != null)
                {
                    int serviceID = 
Convert.ToInt32(gridService.DataKeys[e.RowIndex].Value);
                    using (GriffinContext context = new 
Griffin_Drywall.Models.GriffinContext())
                    {
                        Service obj = context.Services.First(x => x.ServiceID == 
serviceID);
                        obj.ServiceName = txtServiceName.Text;
                        obj.Description = txtDescription.Text;
                        obj.Description2 = txtDescription2.Text;
                        obj.Description3 = txtDescription3.Text;
                        context.SaveChanges();
                        lblMessage.Text = "Saved successfully.";
                        gridService.EditIndex = -1;
                        BindGrid(((GridView)sender).PageIndex - 1);
                    }
                }
                else
                {
                    lblMessage.Text = "Unable to add new service to database.";
                }
            }
        }

        protected void gridService_RowDeleting(object sender, 
GridViewDeleteEventArgs e)
        {
            int serviceID = 
Convert.ToInt32(gridService.DataKeys[e.RowIndex].Value);
            using (GriffinContext context = new 
Griffin_Drywall.Models.GriffinContext())
            {
                Service obj = context.Services.First(x => x.ServiceID ==
serviceID);
                context.Services.Remove(obj);
                context.SaveChanges();
                BindGrid(0);
                lblMessage.Text = "Deleted successfully.";
            }
        }
    }
}
```

### 10.2.13 EditServiceType.aspx

```
<%@ Page Title="" Language="C#" MasterPageFile="~/Site.Master"
AutoEventWireup="true" CodeBehind="EditServiceType.aspx.cs"
Inherits="Griffin_Drywall.Admin.EditServiceType" %>
<asp:Content ID="Content1" ContentPlaceHolderID="HeadContent" runat="server">
</asp:Content>
<asp:Content ID="Content2" ContentPlaceHolderID="FeaturedContent" runat="server">
</asp:Content>
<asp:Content ID="Content3" ContentPlaceHolderID="MainContent" runat="server">
    <article class="projects">
       <h1>Manange Service Types</h1>
        <p>

        </p>
   </article>
    <asp:Label ID="lblMessage" runat="server" Text="" ForeColor="Red"></asp:Label>
    <div style="vertical-align: top; left: auto; overflow: auto; width: auto;">
        <asp:GridView ID="gridServiceType" runat="server"
AutoGenerateColumns="False" ShowFooter="True"
            CssClass="" OnRowCommand="gridServiceType_RowCommand"
            DataKeyNames="ServiceTypeID" CellPadding="4" ForeColor="#333333"
            GridLines="None" OnRowCancelingEdit="gridServiceType_RowCancelingEdit"
            OnRowEditing="gridServiceType_RowEditing"
            OnRowUpdating="gridServiceType_RowUpdating"
            OnRowDeleting="gridServiceType_RowDeleting">
            <AlternatingRowStyle BackColor="White" />
            <Columns>
                <asp:TemplateField HeaderText="">
                    <ItemTemplate>
                        <asp:LinkButton ID="lnkEdit" runat="server" Text=""
CommandName="Edit" ToolTip="Edit"
                            CommandArgument=''><img src="../Images/edit.png"
/></asp:LinkButton>
                        <asp:LinkButton ID="lnkDelete" runat="server"
Text="Delete" CommandName="Delete"
                            ToolTip="Delete" OnClientClick='return confirm("Are
you sure you want to delete this entry?");'
                            CommandArgument=''><img src="../Images/delete.png"
/></asp:LinkButton>
                    </ItemTemplate>
                    <EditItemTemplate>
                        <asp:LinkButton ID="lnkInsert" runat="server" Text=""
ValidationGroup="editGrp" CommandName="Update" ToolTip="Save"
                            CommandArgument=''><img src="../Images/save.png"
/></asp:LinkButton>
                        <asp:LinkButton ID="lnkCancel" runat="server" Text=""
CommandName="Cancel" ToolTip="Cancel"
```

```
                            CommandArgument=''><img src="../Images/refresh.png"
/></asp:LinkButton>
                    </EditItemTemplate>
                    <FooterTemplate>
                        <asp:LinkButton ID="lnkInsert" runat="server" Text=""
ValidationGroup="newGrp" CommandName="InsertNew" ToolTip="Add New Entry"
                            CommandArgument=''><img src="../Images/add.png"
/></asp:LinkButton>
                        <asp:LinkButton ID="lnkCancel" runat="server" Text=""
CommandName="CancelNew" ToolTip="Cancel"
                            CommandArgument=''><img src="../Images/refresh.png"
/></asp:LinkButton>
                    </FooterTemplate>
                </asp:TemplateField>
                <asp:TemplateField HeaderText="Service Name">
                    <EditItemTemplate>
                        <asp:TextBox ID="txtServiceType" runat="server"
Width="150" Text='<%# Bind("Type") %>' CssClass="" MaxLength="100"></asp:TextBox>
                        <asp:RequiredFieldValidator ID="valServiceType"
runat="server" ControlToValidate="txtServiceType"
                            Display="Dynamic" ErrorMessage="Service Type is
required." ForeColor="Red" SetFocusOnError="True"

ValidationGroup="editGrp">*</asp:RequiredFieldValidator>
                    </EditItemTemplate>
                    <ItemTemplate>
                        <asp:Label ID="lblServiceType" runat="server" Width="150"
Text='<%# Bind("Type") %>'></asp:Label>
                    </ItemTemplate>
                    <FooterTemplate>
                        <asp:TextBox ID="txtServiceTypeNew" runat="server"
Width="150" CssClass="" MaxLength="100"></asp:TextBox>
                        <asp:RequiredFieldValidator ID="valServiceTypeNew"
runat="server" ControlToValidate="txtServiceTypeNew"
                            Display="Dynamic" ErrorMessage="Service Type is
required." ForeColor="Red" SetFocusOnError="True"

ValidationGroup="newGrp">*</asp:RequiredFieldValidator>
                    </FooterTemplate>
                </asp:TemplateField>
            </Columns>
            <EditRowStyle BackColor="#2461BF" />
            <FooterStyle BackColor="#507CD1" Font-Bold="True" ForeColor="White" />
            <HeaderStyle BackColor="#507CD1" Font-Bold="True" ForeColor="White" />
            <PagerStyle BackColor="#2461BF" ForeColor="White"
HorizontalAlign="Center" />
            <RowStyle BackColor="#EFF3FB" />
            <SelectedRowStyle BackColor="#D1DDF1" Font-Bold="True"
ForeColor="#333333" />
            <SortedAscendingCellStyle BackColor="#F5F7FB" />
            <SortedAscendingHeaderStyle BackColor="#6D95E1" />
            <SortedDescendingCellStyle BackColor="#E9EBEF" />
            <SortedDescendingHeaderStyle BackColor="#4870BE" />
        </asp:GridView>
        <asp:Repeater ID="rptPager" runat="server">
            <ItemTemplate>
                <asp:LinkButton ID="lnkPage" runat="server"
                    Text='<%#Eval("Text") %>'
                    CommandArgument='<%#Eval("Value") %>'
                    Enabled='<%#Eval("Enabled") %>'
                    OnClick="Page_Changed"
                    ForeColor="#267CB2"
                    Font-Bold="true" />
            </ItemTemplate>
        </asp:Repeater>
    </div>
    <asp:ValidationSummary ID="ValidationSummary1" ValidationGroup="newGrp"
class="validation-summary-errors" runat="server" />
    <asp:ValidationSummary ID="ValidationSummary2" ValidationGroup="editGrp"
class="validation-summary-errors" runat="server" />
    <br />
    <a id="A1" runat="server" href="~/Admin/AdminMenu">Back to Menu</a>
</asp:Content>
```

### 10.2.14 EditServiceType.aspx.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
using Griffin_Drywall.Models;
using System.Web.ModelBinding;

namespace Griffin_Drywall.Admin
{
    public partial class EditServiceType : System.Web.UI.Page
    {
        protected void Page_Load(object sender, EventArgs e)
        {
            if (!IsPostBack)
            {
                BindGrid(0);
            }
            lblMessage.Text = "";
        }

        void BindGrid(int pageIndex)
        {
            using (GriffinContext context = new
Griffin_Drywall.Models.GriffinContext())
            {
                int totalRecords = context.ServiceTypes.Count();
                int pageSize = 5;
                if (pageIndex < 0)
                    { pageIndex = 0; }
                int startRow = pageIndex * pageSize;

                if (context.ServiceTypes.Count() > 0)
                {
                    gridServiceType.DataSource = context.ServiceTypes.OrderBy(x =>
x.ServiceTypeID).Skip(startRow).Take(pageSize).ToList();
                    gridServiceType.DataBind();

                    BindPager(totalRecords, pageIndex, pageSize);
                }
                else
                {
                    var obj = new List<ServiceType>();
                    obj.Add(new ServiceType());
                    // Bind the DataTable which contain a blank row to the
GridView
                    gridServiceType.DataSource = obj.ToList();
                    gridServiceType.DataBind();
                    int columnsCount = gridServiceType.Columns.Count;
                    gridServiceType.Rows[0].Cells.Clear();// clear all the cells
in the row
                    gridServiceType.Rows[0].Cells.Add(new TableCell()); //add a
new blank cell
                    gridServiceType.Rows[0].Cells[0].ColumnSpan = columnsCount;
//set the column span to the new added cell

                    //You can set the styles here
                    gridServiceType.Rows[0].Cells[0].HorizontalAlign =
HorizontalAlign.Center;
                    gridServiceType.Rows[0].Cells[0].ForeColor =
System.Drawing.Color.Red;
                    gridServiceType.Rows[0].Cells[0].Font.Bold = true;
                    //set No Results found to the new added cell
                    gridServiceType.Rows[0].Cells[0].Text = "NO RESULT FOUND!";
                }
            }
        }

        private void BindPager(int totalRecordCount, int currentPageIndex, int
pageSize)
        {
            double getPageCount = (double)((decimal)totalRecordCount /
(decimal)pageSize);
            int pageCount = (int)Math.Ceiling(getPageCount);
            List<ListItem> pages = new List<ListItem>();
            if (pageCount > 1)
            {
                pages.Add(new ListItem("FIRST", "1", currentPageIndex > 1));
                for (int i = 1; i <= pageCount; i++)
                {
                    pages.Add(new ListItem(i.ToString(), i.ToString(), i !=
currentPageIndex + 1));
                }
                pages.Add(new ListItem("LAST", pageCount.ToString(),
currentPageIndex < pageCount - 1));
            }

            rptPager.DataSource = pages;
            rptPager.DataBind();
        }

        protected void Page_Changed(object sender, EventArgs e)
        {
            int pageIndex = Convert.ToInt32(((sender as
LinkButton).CommandArgument));
            gridServiceType.PageIndex = pageIndex;
            BindGrid(pageIndex - 1);
        }
        protected void gridServiceType_RowCommand(object sender,
GridViewCommandEventArgs e)
        {
            if (e.CommandName == "InsertNew")
            {

                GridViewRow row = gridServiceType.FooterRow;
                TextBox txtServiceType = row.FindControl("txtServiceTypeNew") as
TextBox;

                    if (txtServiceType != null)
                    {
                        using (GriffinContext context = new
Griffin_Drywall.Models.GriffinContext())
                        {
                            ServiceType obj = new ServiceType();
                            obj.Type = txtServiceType.Text;
                            context.ServiceTypes.Add(obj);
                            context.SaveChanges();
                            lblMessage.Text = "Added successfully.";
                            BindGrid(((GridView)sender).PageIndex - 1);
                        }
                    }
                    else
                    {
                        lblMessage.Text = "Unable to add new service type to
database.";
                    }
            }
        }

        protected void gridServiceType_RowEditing(object sender,
GridViewEditEventArgs e)
        {
            gridServiceType.EditIndex = e.NewEditIndex;
            BindGrid(((GridView)sender).PageIndex - 1);
        }
        protected void gridServiceType_RowCancelingEdit(object sender,
GridViewCancelEditEventArgs e)
        {
            gridServiceType.EditIndex = -1;
            BindGrid(((GridView)sender).PageIndex - 1);
        }

        protected void gridServiceType_RowUpdating(object sender,
GridViewUpdateEventArgs e)
        {
            GridViewRow row = gridServiceType.Rows[e.RowIndex];

                    TextBox txtServiceType = row.FindControl("txtServiceType") as
TextBox;

                    if (txtServiceType != null)
                    {
                        int serviceTypeID =
Convert.ToInt32(gridServiceType.DataKeys[e.RowIndex].Value);
```

```
                        using (GriffinContext context = new
Griffin_Drywall.Models.GriffinContext())
                        {
                            ServiceType obj = context.ServiceTypes.First(x =>
x.ServiceTypeID == serviceTypeID);
                            obj.Type = txtServiceType.Text;
                            context.SaveChanges();
                            lblMessage.Text = "Saved successfully.";
                            gridServiceType.EditIndex = -1;
                            BindGrid(((GridView)sender).PageIndex - 1);
                        }
                    }
                    else
                    {
                        lblMessage.Text = "Unable to edit service type in
database.";
                    }
        }

        protected void gridServiceType_RowDeleting(object sender,
GridViewDeleteEventArgs e)
        {
            int serviceTypeID =
Convert.ToInt32(gridServiceType.DataKeys[e.RowIndex].Value);
            using (GriffinContext context = new
Griffin_Drywall.Models.GriffinContext())
            {
                ServiceType obj = context.ServiceTypes.First(x => x.ServiceTypeID
== serviceTypeID);
                context.ServiceTypes.Remove(obj);
                context.SaveChanges();
                BindGrid(0);
                lblMessage.Text = "Deleted successfully.";
            }
        }
    }
}
```

### 10.2.15 EditSubCon.aspx

```
<%@ Page Title="" Language="C#" MasterPageFile="~/Site.Master"
AutoEventWireup="true" CodeBehind="EditSubCon.aspx.cs"
Inherits="Griffin_Drywall.Admin.EditSubCon" %>
<asp:Content ID="Content1" ContentPlaceHolderID="HeadContent" runat="server">
</asp:Content>
<asp:Content ID="Content2" ContentPlaceHolderID="FeaturedContent" runat="server">
</asp:Content>
<asp:Content ID="Content3" ContentPlaceHolderID="MainContent" runat="server">
    <article class="projects">
        <h1>Manange Subcontractors</h1>
        <p>
        </p>
    </article>
    <asp:Label ID="lblMessage" runat="server" Text="" ForeColor="Red"></asp:Label>
    <div style="vertical-align: top; left: auto; overflow: auto; width: auto;">
        <asp:GridView ID="gridSubCon" runat="server" AutoGenerateColumns="False"
ShowFooter="True"
            CssClass="" OnRowCommand="gridSubCon_RowCommand"
            DataKeyNames="SubconID" CellPadding="4" ForeColor="#333333"
            GridLines="None" OnRowCancelingEdit="gridSubCon_RowCancelingEdit"
            OnRowEditing="gridSubCon_RowEditing"
            OnRowUpdating="gridSubCon_RowUpdating"
            OnRowDataBound="gridSubCon_RowDataBound"
            OnRowDeleting="gridSubCon_RowDeleting">
            <AlternatingRowStyle BackColor="White" />
            <Columns>
                <asp:TemplateField HeaderText="">
                    <ItemTemplate>
                        <asp:LinkButton ID="lnkEdit" runat="server" Text=""
CommandName="Edit" ToolTip="Edit"
                            CommandArgument=''><img src="../Images/edit.png"
/></asp:LinkButton>
                        <asp:LinkButton ID="lnkDelete" runat="server"
Text="Delete" CommandName="Delete"
                            ToolTip="Delete" OnClientClick='return confirm("Are
you sure you want to delete this entry?");'
                            CommandArgument=''><img src="../Images/delete.png"
/></asp:LinkButton>
                    </ItemTemplate>
                    <EditItemTemplate>
                        <asp:LinkButton ID="lnkInsert" runat="server" Text=""
ValidationGroup="editGrp" CommandName="Update" ToolTip="Save"
                            CommandArgument=''><img src="../Images/save.png"
/></asp:LinkButton>
                        <asp:LinkButton ID="lnkCancel" runat="server" Text=""
CommandName="Cancel" ToolTip="Cancel"
                            CommandArgument=''><img src="../Images/refresh.png"
/></asp:LinkButton>
                    </EditItemTemplate>
                    <FooterTemplate>
                        <asp:LinkButton ID="lnkInsert" runat="server" Text=""
ValidationGroup="newGrp" CommandName="InsertNew" ToolTip="Add New Entry"
                            CommandArgument=''><img src="../Images/add.png"
/></asp:LinkButton>
                        <asp:LinkButton ID="lnkCancel" runat="server" Text=""
CommandName="CancelNew" ToolTip="Cancel"
                            CommandArgument=''><img src="../Images/refresh.png"
/></asp:LinkButton>
                    </FooterTemplate>
                </asp:TemplateField>
                <asp:TemplateField HeaderText="SubCon Name">
                    <EditItemTemplate>
                        <asp:TextBox ID="txtSubConName" runat="server" Width="150"
Text='<%# Bind("SubconName") %>' CssClass="" MaxLength="100"></asp:TextBox>
                        <asp:RequiredFieldValidator ID="valSubConName"
runat="server" ControlToValidate="txtSubConName"
                            Display="Dynamic" ErrorMessage="Subcontractor Name is
required." ForeColor="Red" SetFocusOnError="True"

ValidationGroup="editGrp">*</asp:RequiredFieldValidator>
                    </EditItemTemplate>
                    <ItemTemplate>
                        <asp:Label ID="lblSubConName" runat="server" Width="150"
Text='<%# Bind("SubconName") %>'></asp:Label>
                    </ItemTemplate>
                    <FooterTemplate>
```

```
                        <asp:TextBox ID="txtSubConNameNew" runat="server"
Width="150" CssClass="" MaxLength="100"></asp:TextBox>
                        <asp:RequiredFieldValidator ID="valSubConNameNew"
runat="server" ControlToValidate="txtSubConNameNew"
                            Display="Dynamic" ErrorMessage="Subcontractor is
required." ForeColor="Red" SetFocusOnError="True"

ValidationGroup="newGrp">*</asp:RequiredFieldValidator>
                    </FooterTemplate>
                </asp:TemplateField>
                <asp:TemplateField HeaderText="Description">
                    <EditItemTemplate>
                        <asp:TextBox ID="txtDescription" runat="server" Text='<%#
Bind("Description") %>' CssClass="" TextMode="multiline"
MaxLength="10000"></asp:TextBox>
                    </EditItemTemplate>
                    <ItemTemplate>
                        <asp:Label ID="lblDescription" runat="server" Style="text-
overflow: ellipsis; overflow: hidden" Width="300px" Height="50px" Text='<%#
Bind("Description") %>'></asp:Label>
                    </ItemTemplate>
                    <FooterTemplate>
                        <asp:TextBox ID="txtDescriptionNew" runat="server"
CssClass="" Style="text-overflow: ellipsis; overflow: hidden; margin-left: 5px"
Width="280px" Height="22px" TextMode="multiline" MaxLength="10000"></asp:TextBox>
                    </FooterTemplate>
                </asp:TemplateField>
                <asp:TemplateField HeaderText="User Name">
                    <EditItemTemplate>
                        <asp:DropDownList ID="ddlUser" runat="server">
                        </asp:DropDownList>
                        <asp:RequiredFieldValidator ID="valUser" runat="server"
ControlToValidate="ddlUser"
                            Display="Dynamic" ErrorMessage="User is required."
ForeColor="Red" SetFocusOnError="True"

ValidationGroup="editGrp">*</asp:RequiredFieldValidator>
                    </EditItemTemplate>
                    <ItemTemplate>
                        <asp:Label ID="lblUser" runat="server" Text='<%#
Bind("UserID") %>'></asp:Label>
                    </ItemTemplate>
                    <FooterTemplate>
                        <asp:DropDownList ID="ddlUserNew" runat="server">
                        </asp:DropDownList>
                        <asp:RequiredFieldValidator ID="valUserNew" runat="server"
ControlToValidate="ddlUserNew"
                            Display="Dynamic" ErrorMessage="User is required."
ForeColor="Red" SetFocusOnError="True"

ValidationGroup="newGrp">*</asp:RequiredFieldValidator>
                    </FooterTemplate>
                </asp:TemplateField>
            </Columns>
            <EditRowStyle BackColor="#2461BF" />
            <FooterStyle BackColor="#507CD1" Font-Bold="True" ForeColor="White" />
            <HeaderStyle BackColor="#507CD1" Font-Bold="True" ForeColor="White" />
            <PagerStyle BackColor="#2461BF" ForeColor="White"
HorizontalAlign="Center" />
            <RowStyle BackColor="#EFF3FB" />
            <SelectedRowStyle BackColor="#D1DDF1" Font-Bold="True"
ForeColor="#333333" />
            <SortedAscendingCellStyle BackColor="#F5F7FB" />
            <SortedAscendingHeaderStyle BackColor="#6D95E1" />
            <SortedDescendingCellStyle BackColor="#E9EBEF" />
            <SortedDescendingHeaderStyle BackColor="#4870BE" />
        </asp:GridView>
        <asp:Repeater ID="rptPager" runat="server">
            <ItemTemplate>
                <asp:LinkButton ID="lnkPage" runat="server"
                    Text='<%#Eval("Text") %>'
                    CommandArgument='<%#Eval("Value") %>'
                    Enabled='<%#Eval("Enabled") %>'
                    OnClick="Page_Changed"
                    ForeColor="#267CB2"
                    Font-Bold="true" />
            </ItemTemplate>
        </asp:Repeater>
    </div>
    <asp:ValidationSummary ID="ValidationSummary1" ValidationGroup="newGrp"
class="validation-summary-errors" runat="server" />
    <asp:ValidationSummary ID="ValidationSummary2" ValidationGroup="editGrp"
class="validation-summary-errors" runat="server" />
    <br />
    <a id="A1" runat="server" href="~/Admin/AdminMenu">Back to Menu</a>
</asp:Content>
```

### 10.2.16 EditSubCon.aspx.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
using Griffin_Drywall.Models;
using System.Web.ModelBinding;
using System.Web.Security;

namespace Griffin_Drywall.Admin
{
    public partial class EditSubCon : System.Web.UI.Page
    {
        protected void Page_Load(object sender, EventArgs e)
        {
            if (!IsPostBack)
            {
                BindGrid(0);
            }
            lblMessage.Text = "";
        }

        void BindGrid(int pageIndex)
        {
            using (GriffinContext context = new
Griffin_Drywall.Models.GriffinContext())
```

```
            {
                int totalRecords = context.Subcontractors.Count();
                int pageSize = 10;
                if (pageIndex < 0)
                { pageIndex = 0; }
                int startRow = pageIndex * pageSize;

                if (context.Subcontractors.Count() > 0)
                {
                    gridSubCon.DataSource = context.Subcontractors.OrderBy(x =>
x.SubconID).Skip(startRow).Take(pageSize).ToList();
                    gridSubCon.DataBind();

                    BindPager(totalRecords, pageIndex, pageSize);
                }
                else
                {
                    var obj = new List<Subcontractor>();
                    obj.Add(new Subcontractor());
                    // Bind the DataTable which contain a blank row to the
GridView
                    gridSubCon.DataSource = obj.ToList();
                    gridSubCon.DataBind();
                    int columnsCount = gridSubCon.Columns.Count;
                    gridSubCon.Rows[0].Cells.Clear();// clear all the cells in the
row
                    gridSubCon.Rows[0].Cells.Add(new TableCell()); //add a new
blank cell
                    gridSubCon.Rows[0].Cells[0].ColumnSpan = columnsCount; //set
the column span to the new added cell

                    //You can set the styles here
                    gridSubCon.Rows[0].Cells[0].HorizontalAlign =
HorizontalAlign.Center;
                    gridSubCon.Rows[0].Cells[0].ForeColor =
System.Drawing.Color.Red;
                    gridSubCon.Rows[0].Cells[0].Font.Bold = true;
                    //set No Results found to the new added cell
                    gridSubCon.Rows[0].Cells[0].Text = "NO RESULT FOUND!";
                }
            }
        }

        private void BindPager(int totalRecordCount, int currentPageIndex, int
pageSize)
        {
            double getPageCount = (double)((decimal)totalRecordCount /
(decimal)pageSize);
            int pageCount = (int)Math.Ceiling(getPageCount);
            List<ListItem> pages = new List<ListItem>();
            if (pageCount > 1)
            {
                pages.Add(new ListItem("FIRST", "1", currentPageIndex > 1));
                for (int i = 1; i <= pageCount; i++)
                {
                    pages.Add(new ListItem(i.ToString(), i.ToString(), i !=
currentPageIndex + 1));
                }
                pages.Add(new ListItem("LAST", pageCount.ToString(),
currentPageIndex < pageCount - 1));
            }

            rptPager.DataSource = pages;
            rptPager.DataBind();
        }

        protected void Page_Changed(object sender, EventArgs e)
        {
            int pageIndex = Convert.ToInt32(((sender as
LinkButton).CommandArgument));
            gridSubCon.PageIndex = pageIndex;
            BindGrid(pageIndex - 1);
        }

        protected void gridSubCon_RowDataBound(object sender, GridViewRowEventArgs
e)
        {
            DropDownList ddl = null;
            if (e.Row.RowType == DataControlRowType.Footer)
            {
                ddl = e.Row.FindControl("ddlUserNew") as DropDownList;
            }
            if (e.Row.RowType == DataControlRowType.DataRow)
            {
                ddl = e.Row.FindControl("ddlUser") as DropDownList;
            }
            if (ddl != null)
            {
                using (GriffinContext context = new
Griffin_Drywall.Models.GriffinContext())
                {
                    ddl.DataSource = Membership.GetAllUsers();
                    ddl.DataTextField = "UserName";
                    ddl.DataValueField = "ProviderUserKey";
                    ddl.DataBind();
                    ddl.Items.Insert(0, new ListItem(""));
                }
                if (e.Row.RowType == DataControlRowType.DataRow)
                {
                    ddl.SelectedValue =
((Subcontractor)(e.Row.DataItem)).UserID.ToString();
                }
            }
            Guid text;
            Label lbl = null;
            if (e.Row.RowType == DataControlRowType.DataRow)
            {
                lbl = e.Row.FindControl("lblUser") as Label;
            }
            if (lbl != null && lbl.Text != "")
            {
                //lblMessage.Text = "User:";
               text = new Guid(lbl.Text);
               try
               {
                   lbl.Text = Membership.GetUser(text).UserName;
                }
                catch
                {
                    lbl.Text = "";
                }
            }

        }

        protected void gridSubCon_RowCommand(object sender,
GridViewCommandEventArgs e)
        {
            if (e.CommandName == "InsertNew")
            {
                GridViewRow row = gridSubCon.FooterRow;
                TextBox txtSubConName = row.FindControl("txtSubConNameNew") as
TextBox;
                TextBox txtDescription = row.FindControl("txtDescriptionNew") as
TextBox;
                DropDownList ddlUser = row.FindControl("ddlUserNew") as
DropDownList;
                Guid g = new Guid(ddlUser.SelectedValue);

                if (txtSubConName != null && ddlUser != null)
                {
                    using (GriffinContext context = new
Griffin_Drywall.Models.GriffinContext())
                    {
                        Subcontractor obj = new Subcontractor();
                        obj.SubconName = txtSubConName.Text;
                        obj.Description = txtDescription.Text;
                        obj.UserID = g;
                        context.Subcontractors.Add(obj);
                        context.SaveChanges();
                        lblMessage.Text = "Added successfully.";
                        BindGrid(((GridView)sender).PageIndex - 1);
                    }
                }
                else
                {
                    lblMessage.Text = "Unable to add new subcontractor to
database.";
                }
            }
        }

        protected void gridSubCon_RowEditing(object sender, GridViewEditEventArgs
e)
        {
            gridSubCon.EditIndex = e.NewEditIndex;
            BindGrid(((GridView)sender).PageIndex - 1);
        }
        protected void gridSubCon_RowCancelingEdit(object sender,
GridViewCancelEditEventArgs e)
        {
            gridSubCon.EditIndex = -1;
            BindGrid(((GridView)sender).PageIndex - 1);
        }
        protected void gridSubCon_RowUpdating(object sender,
GridViewUpdateEventArgs e)
        {
            GridViewRow row = gridSubCon.Rows[e.RowIndex];

            TextBox txtSubConName = row.FindControl("txtSubConName") as TextBox;
            TextBox txtDescription = row.FindControl("txtDescription") as TextBox;
            DropDownList ddlUser = row.FindControl("ddlUser") as DropDownList;
            Guid g = new Guid(ddlUser.SelectedValue);

            if (txtSubConName != null && ddlUser != null)
            {
                int subConID =
Convert.ToInt32(gridSubCon.DataKeys[e.RowIndex].Value);
                using (GriffinContext context = new
Griffin_Drywall.Models.GriffinContext())
                {
                    Subcontractor obj = context.Subcontractors.First(x =>
x.SubconID == subConID);
                    obj.SubconName = txtSubConName.Text;
                    obj.Description = txtDescription.Text;
                    obj.UserID = g;
                    context.SaveChanges();
                    lblMessage.Text = "Saved successfully.";
                    gridSubCon.EditIndex = -1;
                    BindGrid(((GridView)sender).PageIndex - 1);
                }
            }
            else
            {
                lblMessage.Text = "Unable to edit subcontractor in database.";
            }
        }

        protected void gridSubCon_RowDeleting(object sender,
GridViewDeleteEventArgs e)
        {
            int subConID = Convert.ToInt32(gridSubCon.DataKeys[e.RowIndex].Value);
            using (GriffinContext context = new
Griffin_Drywall.Models.GriffinContext())
            {
                Subcontractor obj = context.Subcontractors.First(x => x.SubconID
== subConID);
                context.Subcontractors.Remove(obj);
                context.SaveChanges();
                BindGrid(0);
                lblMessage.Text = "Deleted successfully.";
            }
        }
    }
}
```

### 10.2.17 EditUser.aspx

```
<%@ Page Title="" Language="C#" MasterPageFile="~/Site.Master"
AutoEventWireup="true" CodeBehind="EditUser.aspx.cs"
Inherits="Griffin_Drywall.Admin.EditUser" %>
<asp:Content ID="Content1" ContentPlaceHolderID="HeadContent" runat="server">
```

```
</asp:Content>
<asp:Content ID="Content2" ContentPlaceHolderID="FeaturedContent" runat="server">
</asp:Content>
<asp:Content ID="Content3" ContentPlaceHolderID="MainContent" runat="server">
    <table class="webparts">
        <tr>
            <th>User Information</th>
        </tr>
        <tr>
            <td class="details valign">

                <h3>Roles:</h3>
                <asp:CheckBoxList ID="UserRoles" class="userRoles" runat="server"
/>

                <h3>Main Info:</h3>
                <asp:DetailsView AutoGenerateRows="False"
DataSourceID="MemberData"
                    ID="UserInfo" runat="server"
OnItemUpdating="UserInfo_ItemUpdating">
                    <Fields>
                        <asp:BoundField DataField="UserName" HeaderText="User
Name" ReadOnly="True" HeaderStyle-CssClass="detailheader" ItemStyle-
CssClass="detailitem"></asp:BoundField>
                        <asp:BoundField DataField="Email" HeaderText="Email"
HeaderStyle-CssClass="detailheader" ItemStyle-
CssClass="detailitem"></asp:BoundField>
                        <asp:BoundField DataField="Comment" HeaderText="Comment"
HeaderStyle-CssClass="detailheader" ItemStyle-
CssClass="detailitem"></asp:BoundField>
                        <asp:CheckBoxField DataField="IsApproved"
HeaderText="Active User" HeaderStyle-CssClass="detailheader" ItemStyle-
CssClass="detailitem" />
                        <asp:CheckBoxField DataField="IsLockedOut" HeaderText="Is
Locked Out" ReadOnly="true" HeaderStyle-CssClass="detailheader" ItemStyle-
CssClass="detailitem" />

                        <asp:CheckBoxField DataField="IsOnline" HeaderText="Is
Online" ReadOnly="True" HeaderStyle-CssClass="detailheader" ItemStyle-
CssClass="detailitem" />
                        <asp:BoundField DataField="CreationDate"
HeaderText="CreationDate" ReadOnly="True"
                            HeaderStyle-CssClass="detailheader" ItemStyle-
CssClass="detailitem"></asp:BoundField>
                        <asp:BoundField DataField="LastActivityDate"
HeaderText="LastActivityDate" ReadOnly="True" HeaderStyle-CssClass="detailheader"
ItemStyle-CssClass="detailitem"></asp:BoundField>
                        <asp:BoundField DataField="LastLoginDate"
HeaderText="LastLoginDate" ReadOnly="True" HeaderStyle-CssClass="detailheader"
ItemStyle-CssClass="detailitem"></asp:BoundField>
                        <asp:BoundField DataField="LastLockoutDate"
HeaderText="LastLockoutDate" ReadOnly="True" HeaderStyle-CssClass="detailheader"
ItemStyle-CssClass="detailitem"></asp:BoundField>
                        <asp:BoundField DataField="LastPasswordChangedDate"
HeaderText="LastPasswordChangedDate"
                            ReadOnly="True" HeaderStyle-CssClass="detailheader"
ItemStyle-CssClass="detailitem"></asp:BoundField>
```

```
                        <asp:CommandField ButtonType="button"
ShowEditButton="true" EditText="Edit User Info" />
                    </Fields>
                </asp:DetailsView>

                <div class="alert" style="padding: 5px;">
                    <asp:Literal ID="UserUpdateMessage"
runat="server">&nbsp;</asp:Literal>
                </div>

                <div style="text-align: right; width: 100%; margin: 20px 0px;">
                    <asp:Button ID="Button1" runat="server" Text="Unlock User"
OnClick="UnlockUser" OnClientClick="return confirm('Click OK to unlock this
user.')"/>
                    &nbsp;&nbsp;&nbsp;
                    <asp:Button ID="Button2" runat="server" Text="Delete User"
OnClick="DeleteUser" OnClientClick="return confirm('Are you sure?')"/>
                    </div>
                  </td>
            <asp:ObjectDataSource ID="MemberData" runat="server"
DataObjectTypeName="System.Web.Security.MembershipUser" SelectMethod="GetUser"
UpdateMethod="UpdateUser" TypeName="System.Web.Security.Membership">
                <SelectParameters>
                        <asp:QueryStringParameter Name="username"
QueryStringField="username" />
                    </SelectParameters>
            </asp:ObjectDataSource>
        </tr>
    </table>
    <a id="A1" runat="server" href="~/Admin/MgnUsers">Back to Menu</a>
</asp:Content>
```

### 10.2.18 EditUser.aspx.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Web.Security;

namespace Griffin_Drywall.Admin
{
    public partial class EditUser : System.Web.UI.Page
    {
        string username;

        MembershipUser user;

        protected void Page_Load(object sender, EventArgs e)
        {
            username = Request.QueryString["username"];
            if (username == null || username == "")
            {
                Response.Redirect(GetRouteUrl("MgnUsersRoute", null));
            }
            user = Membership.GetUser(username);
            UserUpdateMessage.Text = "";
        }

        protected void UserInfo_ItemUpdating(object sender,
DetailsViewUpdateEventArgs e)
        {
            //Need to handle the update manually because MembershipUser does not
have a
            //parameterless constructor

            user.Email = (string)e.NewValues[0];
            user.Comment = (string)e.NewValues[1];
            user.IsApproved = (bool)e.NewValues[2];

            try
            {
                // Update user info:
                Membership.UpdateUser(user);

                // Update user roles:
                UpdateUserRoles();

                UserUpdateMessage.Text = "Update Successful.";

                e.Cancel = true;
                UserInfo.ChangeMode(DetailsViewMode.ReadOnly);
            }
            catch (Exception ex)
            {
                UserUpdateMessage.Text = "Update Failed: " + ex.Message;

                e.Cancel = true;
                UserInfo.ChangeMode(DetailsViewMode.ReadOnly);
            }
        }

        private void Page_PreRender()
        {
            // Load the User Roles into checkboxes.
            UserRoles.DataSource = Roles.GetAllRoles();
            UserRoles.DataBind();

            // Disable checkboxes if appropriate:
            if (UserInfo.CurrentMode != DetailsViewMode.Edit)
            {
                foreach (ListItem checkbox in UserRoles.Items)
                {
                    checkbox.Enabled = false;
                }
            }

            // Bind these checkboxes to the User's own set of roles.
            string[] userRoles = Roles.GetRolesForUser(username);
            foreach (string role in userRoles)
            {
                ListItem checkbox = UserRoles.Items.FindByValue(role);
                checkbox.Selected = true;
            }
        }
        private void UpdateUserRoles()
        {
            foreach (ListItem rolebox in UserRoles.Items)
            {
                if (rolebox.Selected)
                {
                    if (!Roles.IsUserInRole(username, rolebox.Text))
                    {
                        Roles.AddUserToRole(username, rolebox.Text);
                    }
                }
                else
                {
                    if (Roles.IsUserInRole(username, rolebox.Text))
                    {
                        Roles.RemoveUserFromRole(username, rolebox.Text);
                    }
                }
            }
        }

        protected void UnlockUser(object sender, EventArgs e)
        {
            user.UnlockUser();

            UserInfo.DataBind();
        }

        protected void DeleteUser(object sender, EventArgs e)
        {
            Membership.DeleteUser(username, true);
            Response.Redirect("~/Admin/MgnUsers");
        }
    }
}
```

### 10.2.19 EditWork.aspx

```
<%@ Page Title="" Language="C#" MasterPageFile="~/Site.Master"
AutoEventWireup="true" CodeBehind="EditWork.aspx.cs"
Inherits="Griffin_Drywall.Admin.EditWork" %>
<asp:Content ID="Content1" ContentPlaceHolderID="HeadContent" runat="server">
</asp:Content>
<asp:Content ID="Content2" ContentPlaceHolderID="FeaturedContent" runat="server">
</asp:Content>
<asp:Content ID="Content3" ContentPlaceHolderID="MainContent" runat="server">
    <article class="projects">
       <h1>Schedule Work</h1>
        <p>

        </p>
   </article>
    <article>
        <div>
            <h1>Assign Work</h1>
        </div>
    </article>
     <div style="clear: both"></div>
    <asp:Label ID="lblMessage2" runat="server" Text=""
ForeColor="Red"></asp:Label>
    <div style="vertical-align: top; left: auto; overflow: auto; width: auto;">
        <asp:GridView ID="gridWork" runat="server" AutoGenerateColumns="False"
ShowFooter="True"
            CssClass="" OnRowCommand="gridWork_RowCommand"
            DataKeyNames="JobID,SubconID" CellPadding="4" ForeColor="#333333"
            GridLines="None" OnRowCancelingEdit="gridWork_RowCancelingEdit"
            OnRowEditing="gridWork_RowEditing"
            onrowdatabound="gridWork_RowDataBound"
            OnRowDeleting="gridWork_RowDeleting">
            <AlternatingRowStyle BackColor="White" />
            <Columns>
                <asp:TemplateField HeaderText="">
                    <ItemTemplate>
                        <asp:LinkButton ID="lnkDelete" runat="server"
Text="Delete" CommandName="Delete"
                            ToolTip="Delete" OnClientClick='return confirm("Are
you sure you want to delete this entry? Deleting this assignment will delete any
work details associated with it.");'
                            CommandArgument=''><img src="../Images/delete.png"
/></asp:LinkButton>
                    </ItemTemplate>
                    <FooterTemplate>
                        <asp:LinkButton ID="lnkInsert" runat="server" Text=""
CommandName="InsertNew" ToolTip="Add New Entry"
                            CommandArgument=''><img src="../Images/add.png"
/></asp:LinkButton>
                        <asp:LinkButton ID="lnkCancel" runat="server" Text=""
CommandName="CancelNew" ToolTip="Cancel"
                            CommandArgument=''><img src="../Images/refresh.png"
/></asp:LinkButton>
                    </FooterTemplate>
                </asp:TemplateField>
                <asp:TemplateField HeaderText="Job">
                    <ItemTemplate>
                        <asp:Label ID="lblJob" runat="server" Text='<%#
Bind("Job.JobName") %>'></asp:Label>
                    </ItemTemplate>
                    <FooterTemplate>
                        <asp:DropDownList ID="ddlJobNew" runat="server">
                        </asp:DropDownList>
                    </FooterTemplate>
                </asp:TemplateField>
                 <asp:TemplateField HeaderText="SubContractor">
                    <ItemTemplate>
                        <asp:Label ID="lblSubCon" runat="server" Text='<%#
Bind("Subcontractor.SubconName") %>'></asp:Label>
                    </ItemTemplate>
                    <FooterTemplate>
                        <asp:DropDownList ID="ddlSubConNew" runat="server">
                        </asp:DropDownList>
                    </FooterTemplate>
                </asp:TemplateField>
            </Columns>
            <FooterStyle BackColor="#507CD1" Font-Bold="True" ForeColor="White" />
            <HeaderStyle BackColor="#507CD1" Font-Bold="True" ForeColor="White" />
            <PagerStyle BackColor="#2461BF" ForeColor="White"
HorizontalAlign="Center" />
            <RowStyle BackColor="#EFF3FB" />
            <SelectedRowStyle BackColor="#D1DDF1" Font-Bold="True"
ForeColor="#333333" />
            <SortedAscendingCellStyle BackColor="#F5F7FB" />
            <SortedAscendingHeaderStyle BackColor="#6D95E1" />
            <SortedDescendingCellStyle BackColor="#E9EBEF" />
            <SortedDescendingHeaderStyle BackColor="#4870BE" />
        </asp:GridView>
        <asp:Repeater ID="rptPager1" runat="server">
            <ItemTemplate>
                <asp:LinkButton ID="lnkPage" runat="server"
                    Text='<%#Eval("Text") %>'
                    CommandArgument='<%#Eval("Value") %>'
                    Enabled='<%#Eval("Enabled") %>'
                    OnClick="Page_Changed1"
                    ForeColor="#267CB2"
                    Font-Bold="true" />
            </ItemTemplate>
        </asp:Repeater>
    </div>
    <br />
    <article>
        <div>
            <h1>Manage Work</h1>
        </div>
    </article>
      <div style="clear: both"></div>
    <asp:Label ID="lblMessage" runat="server" Text="" ForeColor="Red"></asp:Label>
    <div style="vertical-align: top; left: auto; overflow: auto; width: auto;">
        <asp:GridView ID="gridWorkDet" runat="server" AutoGenerateColumns="False"
ShowFooter="True"
            CssClass="" OnRowCommand="gridWorkDet_RowCommand"
            DataKeyNames="WorkDetailID" CellPadding="4" ForeColor="#333333"
            GridLines="None" OnRowCancelingEdit="gridWorkDet_RowCancelingEdit"
            OnRowEditing="gridWorkDet_RowEditing"
            OnRowUpdating="gridWorkDet_RowUpdating"
            onrowdatabound="gridWorkDet_RowDataBound"
            OnRowDeleting="gridWorkDet_RowDeleting">
            <AlternatingRowStyle BackColor="White" />
            <Columns>
                <asp:TemplateField HeaderText="">
                    <ItemTemplate>
                        <asp:LinkButton ID="lnkEdit" runat="server" Text=""
CommandName="Edit" ToolTip="Edit"
                            CommandArgument=''><img src="../Images/edit.png"
/></asp:LinkButton>
                        <asp:LinkButton ID="lnkDelete" runat="server"
Text="Delete" CommandName="Delete"
                            ToolTip="Delete" OnClientClick='return confirm("Are
you sure you want to delete this entry?");'
                            CommandArgument=''><img src="../Images/delete.png"
/></asp:LinkButton>
                    </ItemTemplate>
                    <EditItemTemplate>
                        <asp:LinkButton ID="lnkInsert" runat="server" Text=""
ValidationGroup="editGrp" CommandName="Update" ToolTip="Save"
```

```
                            CommandArgument=''><img src="../Images/save.png"
/></asp:LinkButton>
                        <asp:LinkButton ID="lnkCancel" runat="server" Text=""
CommandName="Cancel" ToolTip="Cancel"
                            CommandArgument=''><img src="../Images/refresh.png"
/></asp:LinkButton>
                    </EditItemTemplate>
                    <FooterTemplate>
                        <asp:LinkButton ID="lnkInsert" runat="server" Text=""
ValidationGroup="newGrp" CommandName="InsertNew" ToolTip="Add New Entry"
                            CommandArgument=''><img src="../Images/add.png"
/></asp:LinkButton>
                        <asp:LinkButton ID="lnkCancel" runat="server" Text=""
CommandName="CancelNew" ToolTip="Cancel"
                            CommandArgument=''><img src="../Images/refresh.png"
/></asp:LinkButton>
                    </FooterTemplate>
                </asp:TemplateField>
                <asp:TemplateField HeaderText="Job">
                    <EditItemTemplate>
                        <asp:DropDownList ID="ddlJob" runat="server">
                        </asp:DropDownList>
                    </EditItemTemplate>
                    <ItemTemplate>
                        <asp:Label ID="lblJob" runat="server" Text='<%#
Bind("Work.Job.JobName") %>'></asp:Label>
                    </ItemTemplate>
                    <FooterTemplate>
                        <asp:DropDownList ID="ddlJobNew" runat="server">
                        </asp:DropDownList>
                    </FooterTemplate>
                </asp:TemplateField>
                 <asp:TemplateField HeaderText="SubContractor">
                    <EditItemTemplate>
                        <asp:DropDownList ID="ddlSubCon" runat="server">
                        </asp:DropDownList>
                    </EditItemTemplate>
                    <ItemTemplate>
                        <asp:Label ID="lblSubCon" runat="server" Text='<%#
Bind("Work.Subcontractor.SubconName") %>'></asp:Label>
                    </ItemTemplate>
                    <FooterTemplate>
                        <asp:DropDownList ID="ddlSubConNew" runat="server">
                        </asp:DropDownList>
                    </FooterTemplate>
                </asp:TemplateField>

               <asp:TemplateField HeaderText="Number of Frames">
                    <EditItemTemplate>
                        <asp:TextBox ID="txtFrames" runat="server" Text='<%#
Bind("NumberofFrames") %>' Width="50" CssClass="" MaxLength="10"></asp:TextBox>
                        <asp:RequiredFieldValidator ID="valFrames" runat="server"
ControlToValidate="txtFrames"
                            Display="Dynamic" ErrorMessage="Number of Frames is
required." ForeColor="Red" SetFocusOnError="True"

ValidationGroup="editGrp">*</asp:RequiredFieldValidator>
```

```
                        <asp:CompareValidator ID="CompareValidator1"
runat="server" Operator="DataTypeCheck" Type="Integer"
                            ControlToValidate="txtFrames" ErrorMessage="Frames
must be a whole number"  ValidationGroup="editGrp">*</asp:CompareValidator>
                    </EditItemTemplate>
                    <ItemTemplate>
                        <asp:Label ID="lblFrames" runat="server" Width="50"
Text='<%# Bind("NumberofFrames") %>'></asp:Label>
                    </ItemTemplate>
                    <FooterTemplate>
                        <asp:TextBox ID="txtFramesNew" runat="server" Width="50"
CssClass="" MaxLength="10"></asp:TextBox>
                        <asp:RequiredFieldValidator ID="valFramesNew"
runat="server" ControlToValidate="txtFramesNew"
                            Display="Dynamic" ErrorMessage="Number of Frames is
required." ForeColor="Red" SetFocusOnError="True"

ValidationGroup="newGrp">*</asp:RequiredFieldValidator>
                        <asp:CompareValidator ID="CompareValidator2"
runat="server" Operator="DataTypeCheck" Type="Integer"
                            ControlToValidate="txtFramesNew" ErrorMessage="Frames
must be a whole number"  ValidationGroup="newGrp">*</asp:CompareValidator>
                    </FooterTemplate>
                </asp:TemplateField>
                <asp:TemplateField HeaderText="Work Date">
                    <EditItemTemplate>
                        <asp:TextBox ID="txtWorkDate" runat="server" Width="150"
Text='<%# Bind("WorkDate", "{0:MM/dd/yyyy}") %>' CssClass=""
MaxLength="50"></asp:TextBox>
                        <asp:RequiredFieldValidator ID="valWorkDate"
runat="server" ControlToValidate="txtWorkDate"
                            Display="Dynamic" ErrorMessage="Work Date is
required." ForeColor="Red" SetFocusOnError="True"

ValidationGroup="editGrp">*</asp:RequiredFieldValidator>
                        <asp:CompareValidator
                            ID="dateValidator1" runat="server"
                            Type="Date"
                            Operator="DataTypeCheck"
                            ControlToValidate="txtWorkDate"
                            ErrorMessage="Please enter a valid date."
                            ForeColor="Red" SetFocusOnError="True"
                            ValidationGroup="editGrp">*
                        </asp:CompareValidator>
                        <ajaxToolkit:CalendarExtender
                            ID="CalendarExtender1"
                            TargetControlID="txtWorkDate"
                            runat="server" />
                    </EditItemTemplate>
                    <ItemTemplate>
                        <asp:Label ID="lblWorkDate" runat="server" Width="150"
Text='<%# Bind("WorkDate", "{0:MM/dd/yyyy}") %>'></asp:Label>
                    </ItemTemplate>
                    <FooterTemplate>
                        <asp:TextBox ID="txtWorkDateNew" runat="server"
Width="150" CssClass="" MaxLength="50"></asp:TextBox>
                        <asp:RequiredFieldValidator ID="valWorkDateNew"
runat="server" ControlToValidate="txtWorkDateNew"
```

```
                            Display="Dynamic" ErrorMessage="Work Date is
required." ForeColor="Red" SetFocusOnError="True"

ValidationGroup="newGrp">*</asp:RequiredFieldValidator>
                        <asp:CompareValidator
                            ID="dateValidator2" runat="server"
                            Type="Date"
                            Operator="DataTypeCheck"
                            ControlToValidate="txtWorkDateNew"
                            ErrorMessage="Please enter a valid date."
                            ForeColor="Red" SetFocusOnError="True"
                            ValidationGroup="newGrp">*
                        </asp:CompareValidator>
                        <ajaxToolkit:CalendarExtender
                            ID="CalendarExtender2"
                            TargetControlID="txtWorkDateNew"
                            runat="server" />
                    </FooterTemplate>
                </asp:TemplateField>

            </Columns>
            <EditRowStyle BackColor="#2461BF" />
            <FooterStyle BackColor="#507CD1" Font-Bold="True" ForeColor="White" />
            <HeaderStyle BackColor="#507CD1" Font-Bold="True" ForeColor="White" />
            <PagerStyle BackColor="#2461BF" ForeColor="White"
HorizontalAlign="Center" />
            <RowStyle BackColor="#EFF3FB" />
            <SelectedRowStyle BackColor="#D1DDF1" Font-Bold="True"
ForeColor="#333333" />
            <SortedAscendingCellStyle BackColor="#F5F7FB" />
            <SortedAscendingHeaderStyle BackColor="#6D95E1" />
            <SortedDescendingCellStyle BackColor="#E9EBEF" />
            <SortedDescendingHeaderStyle BackColor="#4870BE" />
        </asp:GridView>
        <asp:Repeater ID="rptPager" runat="server">
            <ItemTemplate>
                <asp:LinkButton ID="lnkPage" runat="server"
                    Text='<%#Eval("Text") %>'
                    CommandArgument='<%#Eval("Value") %>'
                    Enabled='<%#Eval("Enabled") %>'
                    OnClick="Page_Changed"
                    ForeColor="#267CB2"
                    Font-Bold="true" />
            </ItemTemplate>
        </asp:Repeater>
    </div>
    <asp:ValidationSummary ID="ValidationSummary1" ValidationGroup="newGrp"
class="validation-summary-errors" runat="server" />
    <asp:ValidationSummary ID="ValidationSummary2" ValidationGroup="editGrp"
class="validation-summary-errors" runat="server" />
    <br />
    <a id="A1" runat="server" href="~/Admin/AdminMenu">Back to Menu</a>
</asp:Content>
```

### 10.2.20 EditWork.aspx.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
using Griffin_Drywall.Models;
using System.Web.ModelBinding;

namespace Griffin_Drywall.Admin
{
    public partial class EditWork : System.Web.UI.Page
    {
        protected void Page_Load(object sender, EventArgs e)
        {
            if (!IsPostBack)
            {
                BindGrid1(0);
                BindGrid(0);
            }
            lblMessage.Text = "";
            lblMessage2.Text = "";
        }
        void BindGrid1(int pageIndex)
        {
            using (GriffinContext context = new
Griffin_Drywall.Models.GriffinContext())
            {
                int totalRecords = context.Works.Count();
                int pageSize = 10;
                if (pageIndex < 0)
                { pageIndex = 0; }
                int startRow = pageIndex * pageSize;

                if (context.Works.Count() > 0)
                {

                    gridWork.DataSource = context.Works.OrderBy(x =>
x.SubconID).ThenBy(x => x.JobID).Skip(startRow).Take(pageSize).ToList();
                    gridWork.DataBind();

                    BindPager1(totalRecords, pageIndex, pageSize);
                }
                else
                {
                    var obj = new List<Work>();
                    obj.Add(new Work());
                    // Bind the DataTable which contain a blank row to the
GridView
                    gridWork.DataSource = obj.ToList();
                    gridWork.DataBind();
                    int columnsCount = gridWorkDet.Columns.Count;
                    gridWork.Rows[0].Cells.Clear();// clear all the cells in the
row
                    gridWork.Rows[0].Cells.Add(new TableCell()); //add a new blank
cell
                    gridWork.Rows[0].Cells[0].ColumnSpan = columnsCount; //set the
column span to the new added cell

                    //You can set the styles here
                    gridWork.Rows[0].Cells[0].HorizontalAlign =
HorizontalAlign.Center;
                    gridWork.Rows[0].Cells[0].ForeColor =
System.Drawing.Color.Red;
                    gridWork.Rows[0].Cells[0].Font.Bold = true;
                    //set No Results found to the new added cell
                    gridWork.Rows[0].Cells[0].Text = "NO RESULT FOUND!";
                }
            }
        }
        void BindGrid(int pageIndex)
        {
            using (GriffinContext context = new
Griffin_Drywall.Models.GriffinContext())
            {
                int totalRecords = context.WorkDetails.Count();
                int pageSize = 10;
                if (pageIndex < 0)
                { pageIndex = 0; }
                int startRow = pageIndex * pageSize;

                if (context.WorkDetails.Count() > 0)
                {

                    gridWorkDet.DataSource = context.WorkDetails.OrderBy(x =>
x.SubconID).ThenBy(x => x.JobID).Skip(startRow).Take(pageSize).ToList();
                    gridWorkDet.DataBind();

                    BindPager(totalRecords, pageIndex, pageSize);
                }
                else
                {
                    var obj = new List<WorkDetail>();
                    obj.Add(new WorkDetail());
                    // Bind the DataTable which contain a blank row to the
GridView
                    gridWorkDet.DataSource = obj.ToList();
                    gridWorkDet.DataBind();
                    int columnsCount = gridWorkDet.Columns.Count;
                    gridWorkDet.Rows[0].Cells.Clear();// clear all the cells in
the row
                    gridWorkDet.Rows[0].Cells.Add(new TableCell()); //add a new
blank cell
                    gridWorkDet.Rows[0].Cells[0].ColumnSpan = columnsCount; //set
the column span to the new added cell

                    //You can set the styles here
                    gridWorkDet.Rows[0].Cells[0].HorizontalAlign =
HorizontalAlign.Center;
                    gridWorkDet.Rows[0].Cells[0].ForeColor =
System.Drawing.Color.Red;
                    gridWorkDet.Rows[0].Cells[0].Font.Bold = true;
                    //set No Results found to the new added cell
                    gridWorkDet.Rows[0].Cells[0].Text = "NO RESULT FOUND!";
                }
            }
        }
        private void BindPager(int totalRecordCount, int currentPageIndex, int
pageSize)
        {
            double getPageCount = (double)((decimal)totalRecordCount /
(decimal)pageSize);
            int pageCount = (int)Math.Ceiling(getPageCount);
            List<ListItem> pages = new List<ListItem>();
            if (pageCount > 1)
            {
                pages.Add(new ListItem("FIRST", "1", currentPageIndex > 1));
                for (int i = 1; i <= pageCount; i++)
                {
                    pages.Add(new ListItem(i.ToString(), i.ToString(), i !=
currentPageIndex + 1));
                }
                pages.Add(new ListItem("LAST", pageCount.ToString(),
currentPageIndex < pageCount - 1));
            }

            rptPager.DataSource = pages;
            rptPager.DataBind();
        }

        private void BindPager1(int totalRecordCount, int currentPageIndex, int
pageSize)
        {
            double getPageCount = (double)((decimal)totalRecordCount /
(decimal)pageSize);
            int pageCount = (int)Math.Ceiling(getPageCount);
            List<ListItem> pages = new List<ListItem>();
            if (pageCount > 1)
            {
                pages.Add(new ListItem("FIRST", "1", currentPageIndex > 1));
                for (int i = 1; i <= pageCount; i++)
                {
                    pages.Add(new ListItem(i.ToString(), i.ToString(), i !=
currentPageIndex + 1));
                }
                pages.Add(new ListItem("LAST", pageCount.ToString(),
currentPageIndex < pageCount - 1));
            }

            rptPager1.DataSource = pages;
            rptPager1.DataBind();
        }

        protected void Page_Changed(object sender, EventArgs e)
        {
            int pageIndex = Convert.ToInt32(((sender as
LinkButton).CommandArgument));
            gridWorkDet.PageIndex = pageIndex;
            BindGrid(pageIndex - 1);
            BindGrid1(0);
        }

        protected void Page_Changed1(object sender, EventArgs e)
        {
            int pageIndex = Convert.ToInt32(((sender as
LinkButton).CommandArgument));
            gridWork.PageIndex = pageIndex;
            BindGrid1(pageIndex - 1);
            BindGrid(0);
        }

        protected void gridWorkDet_RowDataBound(object sender,
GridViewRowEventArgs e)
        {
            DropDownList ddl = null;
            DropDownList ddl2 = null;

            if (e.Row.RowType == DataControlRowType.Footer)
            {
                ddl = e.Row.FindControl("ddlJobNew") as DropDownList;
                ddl2 = e.Row.FindControl("ddlSubConNew") as DropDownList;
            }
            if (e.Row.RowType == DataControlRowType.DataRow)
            {
                ddl = e.Row.FindControl("ddlJob") as DropDownList;
                ddl2 = e.Row.FindControl("ddlSubCon") as DropDownList;
            }
            if (ddl != null)
            {
                using (GriffinContext context = new
Griffin_Drywall.Models.GriffinContext())
                {
                    ddl.DataSource = context.Jobs.ToList();
                    ddl.DataTextField = "JobName";
                    ddl.DataValueField = "JobID";
                    ddl.DataBind();
                }
                if (e.Row.RowType == DataControlRowType.DataRow)
                {
                    ddl.SelectedValue =
((WorkDetail)(e.Row.DataItem)).JobID.ToString();
                }
            }
            if (ddl2 != null)
            {
                using (GriffinContext context = new
Griffin_Drywall.Models.GriffinContext())
                {
                    ddl2.DataSource = context.Subcontractors.ToList();
                    ddl2.DataTextField = "SubconName";
                    ddl2.DataValueField = "SubconID";
                    ddl2.DataBind();
                }
                if (e.Row.RowType == DataControlRowType.DataRow)
                {
                    ddl2.SelectedValue =
((WorkDetail)(e.Row.DataItem)).SubconID.ToString();
                }
            }

        }
        protected void gridWork_RowDataBound(object sender, GridViewRowEventArgs
e)
        {
            DropDownList ddl = null;
            DropDownList ddl2 = null;

            if (e.Row.RowType == DataControlRowType.Footer)
            {
                ddl = e.Row.FindControl("ddlJobNew") as DropDownList;
                ddl2 = e.Row.FindControl("ddlSubConNew") as DropDownList;
            }
            if (e.Row.RowType == DataControlRowType.DataRow)
            {
                ddl = e.Row.FindControl("ddlJob") as DropDownList;
                ddl2 = e.Row.FindControl("ddlSubCon") as DropDownList;
            }
            if (ddl != null)
            {
                using (GriffinContext context = new
Griffin_Drywall.Models.GriffinContext())
                {
                    ddl.DataSource = context.Jobs.ToList();
                    ddl.DataTextField = "JobName";
                    ddl.DataValueField = "JobID";
                    ddl.DataBind();
                }
                if (e.Row.RowType == DataControlRowType.DataRow)
                {
                    ddl.SelectedValue = ((Work)(e.Row.DataItem)).JobID.ToString();
                }
            }
            if (ddl2 != null)
            {
                using (GriffinContext context = new
Griffin_Drywall.Models.GriffinContext())
                {
                    ddl2.DataSource = context.Subcontractors.ToList();
                    ddl2.DataTextField = "SubconName";
                    ddl2.DataValueField = "SubconID";
                    ddl2.DataBind();
                }
                if (e.Row.RowType == DataControlRowType.DataRow)
                {
                    ddl2.SelectedValue =
((Work)(e.Row.DataItem)).SubconID.ToString();
                }
            }
        }

        protected void gridWorkDet_RowCommand(object sender,
GridViewCommandEventArgs e)
        {
            if (e.CommandName == "InsertNew")
            {
                GridViewRow row = gridWorkDet.FooterRow;
                DropDownList ddlJob = row.FindControl("ddlJobNew") as
DropDownList;
```

```
                DropDownList ddlSubCon = row.FindControl("ddlSubConNew") as
DropDownList;
                TextBox txtNumFrames = row.FindControl("txtFramesNew") as TextBox;
                TextBox txtWorkDate = row.FindControl("txtWorkDateNew") as
TextBox;
                DateTime dtWorkDate = DateTime.Parse(txtWorkDate.Text);
                DateTime DateCreated = DateTime.Now;

                if (ddlJob != null && ddlSubCon != null && txtNumFrames != null &&
dtWorkDate != null)
                {
                    int jobID = Convert.ToInt32(ddlJob.SelectedValue);
                    int subconID = Convert.ToInt32(ddlSubCon.SelectedValue);
                    try
                    {
                        using (GriffinContext context = new
Griffin_Drywall.Models.GriffinContext())
                        {
                            WorkDetail obj = new WorkDetail();
                            obj.JobID = jobID;
                            obj.SubconID = subconID;
                            obj.WorkDate = dtWorkDate;
                            obj.NumberofFrames =
Convert.ToInt32(txtNumFrames.Text);
                            obj.DateCreated = DateTime.Now;
                            context.WorkDetails.Add(obj);
                            context.SaveChanges();
                            lblMessage.Text = "Added successfully.";
                            BindGrid(((GridView)sender).PageIndex - 1);
                            BindGrid1(0);
                        }
                    }
                    catch { lblMessage.Text = "You can't add work until you assign
the job to the subcontractor."; }
                }
                else
                {
                    lblMessage.Text = "Unable to add new work detail to
database.";
                }
            }
        }

        protected void gridWork_RowCommand(object sender, GridViewCommandEventArgs
e)
        {
            if (e.CommandName == "InsertNew")
            {
                GridViewRow row = gridWork.FooterRow;
                DropDownList ddlJob = row.FindControl("ddlJobNew") as
DropDownList;
                DropDownList ddlSubCon = row.FindControl("ddlSubConNew") as
DropDownList;

                if (ddlJob != null && ddlSubCon != null)
                {
                    int jobID = Convert.ToInt32(ddlJob.SelectedValue);
                    int subconID = Convert.ToInt32(ddlSubCon.SelectedValue);
                    try
                    {
                        using (GriffinContext context = new
Griffin_Drywall.Models.GriffinContext())
                        {
                            Work obj = new Work();
                            obj.JobID = jobID;
                            obj.SubconID = subconID;
                            obj.Subcontractor = context.Subcontractors.First(x =>
x.SubconID == subconID);
                            obj.Job = context.Jobs.First(x => x.JobID == jobID);
                            context.Works.Add(obj);
                            context.SaveChanges();
                            lblMessage2.Text = "Added successfully.";
                            BindGrid1(((GridView)sender).PageIndex - 1);
                            BindGrid(0);
                        }
                    }
                    catch { lblMessage2.Text = "The work assignment already exists
in the database. If this is not true, contact Administrator."; }
                }
                else
                {
                    lblMessage2.Text = "Unable to assign work.";
                }
            }
        }

        protected void gridWorkDet_RowEditing(object sender, GridViewEditEventArgs
e)
        {
            gridWorkDet.EditIndex = e.NewEditIndex;
            BindGrid(((GridView)sender).PageIndex - 1);
            BindGrid1(0);
        }
        protected void gridWorkDet_RowCancelingEdit(object sender,
GridViewCancelEditEventArgs e)
        {
            gridWorkDet.EditIndex = -1;
            BindGrid(((GridView)sender).PageIndex - 1);
            BindGrid1(0);
        }

        protected void gridWork_RowEditing(object sender, GridViewEditEventArgs e)
        {
            gridWork.EditIndex = e.NewEditIndex;
            BindGrid1(((GridView)sender).PageIndex - 1);
            BindGrid(0);
        }
        protected void gridWork_RowCancelingEdit(object sender,
GridViewCancelEditEventArgs e)
        {
            gridWork.EditIndex = -1;
            BindGrid1(((GridView)sender).PageIndex - 1);
            BindGrid(0);
        }
        protected void gridWorkDet_RowUpdating(object sender,
GridViewUpdateEventArgs e)
        {
            GridViewRow row = gridWorkDet.Rows[e.RowIndex];

            DropDownList ddlJob = row.FindControl("ddlJob") as DropDownList;
            DropDownList ddlSubCon = row.FindControl("ddlSubCon") as DropDownList;
            TextBox txtNumFrames = row.FindControl("txtFrames") as TextBox;
            TextBox txtWorkDate = row.FindControl("txtWorkDate") as TextBox;
            DateTime dtWorkDate = DateTime.Parse(txtWorkDate.Text);


            if (ddlJob != null && ddlSubCon != null && txtNumFrames != null &&
dtWorkDate != null)
            {
                int workDetailID =
Convert.ToInt32(gridWorkDet.DataKeys[e.RowIndex].Value);

                try
                {
                using (GriffinContext context = new
Griffin_Drywall.Models.GriffinContext())
                {
                    WorkDetail obj = context.WorkDetails.First(x => x.WorkDetailID
== workDetailID);
                    obj.JobID = Convert.ToInt32(ddlJob.SelectedValue);
                    obj.SubconID = Convert.ToInt32(ddlSubCon.SelectedValue);
                    obj.WorkDate = dtWorkDate;
                    obj.NumberofFrames = Convert.ToInt32(txtNumFrames.Text);
                    context.SaveChanges();
                    lblMessage.Text = "Saved successfully.";
                    gridWorkDet.EditIndex = -1;
                    BindGrid(((GridView)sender).PageIndex - 1);
                    BindGrid1(0);
                }
                }
                catch { lblMessage.Text = "Edit cannot be processed. You are
either editing this entry to an entry which already exists in the database or the
work date is too ambiguous (Please check date)."; }
            }

            else
            {
                lblMessage.Text = "Unable to edit work detail in database.";
            }
        }

        protected void gridWorkDet_RowDeleting(object sender,
GridViewDeleteEventArgs e)
        {
            int workDetailID =
Convert.ToInt32(gridWorkDet.DataKeys[e.RowIndex].Value);

            using (GriffinContext context = new
Griffin_Drywall.Models.GriffinContext())
            {
                WorkDetail obj = context.WorkDetails.First(x => x.WorkDetailID ==
workDetailID);
```

```
                context.WorkDetails.Remove(obj);
                context.SaveChanges();
                BindGrid(0);
                BindGrid1(0);
                lblMessage.Text = "Deleted successfully.";
            }
        }
        protected void gridWork_RowDeleting(object sender, GridViewDeleteEventArgs
e)
        {
            int jobID =
Convert.ToInt32(gridWork.DataKeys[e.RowIndex].Values[0].ToString());
            int subconID =
Convert.ToInt32(gridWork.DataKeys[e.RowIndex].Values[1].ToString());

            using (GriffinContext context = new
Griffin_Drywall.Models.GriffinContext())
            {
                Work obj = context.Works.First(x => x.JobID == jobID && x.SubconID
== subconID);
                context.Works.Remove(obj);
                context.SaveChanges();
                BindGrid1(0);
                BindGrid(0);
                lblMessage2.Text = "Deleted successfully.";
            }
        }
    }
}
```

### 10.2.21 MgnUsers.aspx

```
<%@ Page Title="" Language="C#" MasterPageFile="~/Site.Master"
AutoEventWireup="true" CodeBehind="MgnUsers.aspx.cs"
Inherits="Griffin_Drywall.Admin.MgnUsers" %>

<asp:Content ID="Content1" ContentPlaceHolderID="HeadContent" runat="server">
</asp:Content>
<asp:Content ID="Content2" ContentPlaceHolderID="FeaturedContent" runat="server">
</asp:Content>
<asp:Content ID="Content3" ContentPlaceHolderID="MainContent" runat="server">

    <table class="webparts">
<tr>
       <th>Users by Name</th>
</tr>
<tr>
<td class="details valign">
User Name filter:&nbsp;&nbsp;&nbsp;
<asp:repeater id="rptLetters" runat="server" OnItemCommand="letters_ItemCommand">
  <itemtemplate>
    <asp:linkbutton id="lnkLetter" runat="server"
            commandname="Filter"
            commandargument='<%# DataBinder.Eval(Container, "DataItem.Letter")%>'>
      <%# DataBinder.Eval(Container, "DataItem.Letter")%>
    </asp:linkbutton>
  </itemtemplate>
</asp:repeater>
<br />
```

```
<asp:GridView runat="server" ID="Users" AutoGenerateColumns="false"
       CssClass="list" AlternatingRowStyle-CssClass="odd" GridLines="none"
       >
<Columns>
       <asp:TemplateField>
              <HeaderTemplate>User Name</HeaderTemplate>
              <ItemTemplate>
              <a href="Admin/EditUser.aspx?username=<%# Eval("UserName") %>"><%#
Eval("UserName") %></a>
              </ItemTemplate>
       </asp:TemplateField>
       <asp:BoundField DataField="email" HeaderText="Email" />
       <asp:BoundField DataField="comment" HeaderText="Comments" />
       <asp:BoundField DataField="creationdate" HeaderText="Creation Date" />
       <asp:BoundField DataField="lastlogindate" HeaderText="Last Login Date" />
       <asp:BoundField DataField="lastactivitydate" HeaderText="Last Activity
Date" />
       <asp:BoundField DataField="isapproved" HeaderText="Is Active" />
       <asp:BoundField DataField="isonline" HeaderText="Is Online" />
       <asp:BoundField DataField="islockedout" HeaderText="Is Locked Out" />
</Columns>
</asp:GridView>
    <br />
    <a runat="server" href="~/Admin/AdminMenu">Back to Menu</a>
</td>

</tr></table>
</asp:Content>
```

### 10.2.22 MgnUsers.aspx.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Web.Security;
using System.Data;

namespace Griffin_Drywall.Admin
{
        public partial class MgnUsers : System.Web.UI.Page
        {

            #region Web Form Protected/Private Fields

            protected string _letterFilter;

            #endregion

            private void Page_Load(object sender, System.EventArgs e)
            {
                object oFilter = ViewState[this.ToString() + "_letterFilter"];
                if (oFilter != null) _letterFilter = (string)oFilter;
                else _letterFilter = "All";

                if (!IsPostBack) initControls();
            }

            private void initControls()
            {
                Users_Bind(true);
            }

            private void Users_Bind(bool requery)
            {

                if (_letterFilter == "All")
                    Users.DataSource = Membership.GetAllUsers();
                else
                    Users.DataSource = Membership.FindUsersByName(_letterFilter +
"%");

                Users.DataBind();

                letters_Bind();
            }

            private void letters_Bind()
            {
                DataTable dt;

                if (Session[this.ToString() + "_LettersData"] == null)
                {
                    string[] letters = { "A", "B", "C", "D", "E", "F", "G", "H",
"I", "J",
                             "K", "L", "M", "N", "O", "P", "Q", "R", "S", "T",
                             "U", "V", "W", "X", "Y", "Z", "All"};

                    dt = new DataTable();

                    dt.Columns.Add(new DataColumn("Letter",
                      typeof(string)));

                    for (int i = 0; i < letters.Length; i++)
                    {
                        DataRow dr = dt.NewRow();
                        dr[0] = letters[i];
                        dt.Rows.Add(dr);
                    }

                    Session[this.ToString() + "_LettersData"] = dt;
                }
                else
                    dt = (DataTable)Session[this.ToString() + "_LettersData"];

                rptLetters.DataSource = dt.DefaultView;
                rptLetters.DataBind();
            }

            private void letters_ItemDataBound(object sender,
System.Web.UI.WebControls.RepeaterItemEventArgs e)
            {
                DataRowView data = (DataRowView)e.Item.DataItem;
                if ((string)data[0] == _letterFilter)
                {
                    LinkButton lnkletter =
(LinkButton)e.Item.FindControl("lnkletter");
                    lnkletter.Enabled = false;
                }
            }

            protected void letters_ItemCommand(object source,
RepeaterCommandEventArgs e)
            {
                if (e.CommandName == "Filter")
                {
                    _letterFilter = (string)e.CommandArgument;
                    ViewState[this.ToString() + "_LetterFilter"] = _letterFilter;
                    Users_Bind(false);
                }
            }
        }
}
```

### 10.2.23 ViewWork.aspx

```
<%@ Page Title="" Language="C#" MasterPageFile="~/Site.Master"
AutoEventWireup="true" CodeBehind="ViewWork.aspx.cs"
Inherits="Griffin_Drywall.Admin.ViewWork" %>
<asp:Content ID="Content1" ContentPlaceHolderID="HeadContent" runat="server">
</asp:Content>
<asp:Content ID="Content2" ContentPlaceHolderID="FeaturedContent" runat="server">
</asp:Content>
<asp:Content ID="Content3" ContentPlaceHolderID="MainContent" runat="server">
    <article class="projects">
       <h1>View Work</h1>
        <p>

        </p>
   </article>Filters: <br />
    Subcontractor:
    <asp:DropDownList ID="ddlSubCon" runat="server" AutoPostBack="True"
OnSelectedIndexChanged="ddlSubCon_SelectedIndexChanged">
                        </asp:DropDownList>
    Job:
    <asp:DropDownList ID="ddlJob" runat="server" AutoPostBack="True"
OnSelectedIndexChanged="ddlJob_SelectedIndexChanged">
                        </asp:DropDownList>
    <asp:GridView ID="GridView1" runat="server" AllowPaging="True" CellPadding="4"
DataSourceID="GriffinContextEntityDataSource" ForeColor="#333333" GridLines="None"
AutoGenerateColumns="False">
        <AlternatingRowStyle BackColor="White" />
        <EditRowStyle BackColor="#2461BF" />
        <FooterStyle BackColor="#507CD1" Font-Bold="True" ForeColor="White" />
        <HeaderStyle BackColor="#507CD1" Font-Bold="True" ForeColor="White" />
        <PagerStyle BackColor="#2461BF" ForeColor="White" HorizontalAlign="Center"
/>
        <RowStyle BackColor="#EFF3FB" />
        <SelectedRowStyle BackColor="#D1DDF1" Font-Bold="True" ForeColor="#333333"
/>
        <SortedAscendingCellStyle BackColor="#F5F7FB" />
        <SortedAscendingHeaderStyle BackColor="#6D95E1" />
        <SortedDescendingCellStyle BackColor="#E9EBEF" />
        <SortedDescendingHeaderStyle BackColor="#4870BE" />
        <Columns>
            <asp:TemplateField>
                    <HeaderTemplate>
                        <asp:Label ID="Label1" runat="server" Width="150"
Text="Subcontractor"></asp:Label>
                    </HeaderTemplate>
                    <ItemTemplate>
                        <asp:Label ID="Label2" runat="server" Text='<%#
Bind("Work.Subcontractor.SubconName") %>'></asp:Label>
                    </ItemTemplate>
                </asp:TemplateField>
                <asp:TemplateField>
                    <HeaderTemplate>
                        <asp:Label ID="Label3" runat="server" Text="Job
Name"></asp:Label>
                    </HeaderTemplate>
                    <ItemTemplate>
                        <asp:Label ID="Label4" runat="server" Text='<%#
Bind("Work.Job.JobName") %>'></asp:Label>
                    </ItemTemplate>
                </asp:TemplateField>
                <asp:TemplateField>
                    <HeaderTemplate>
                        <asp:Label ID="Label5" runat="server" Text="Number of
Frames"></asp:Label>
                    </HeaderTemplate>
                    <ItemTemplate>
                        <div style="text-align: center;">
                        <asp:Label ID="Label6" runat="server" Text='<%#
Eval("NumberofFrames") %>'></asp:Label>
                            </div>
                    </ItemTemplate>
                </asp:TemplateField>
                <asp:TemplateField>
                    <HeaderTemplate>
                        <asp:Label ID="Label7" runat="server" Width="100"
Text="Work Date"></asp:Label>
                    </HeaderTemplate>
                    <ItemTemplate>
                        <asp:Label ID="Label8" runat="server"  Text='<%#
Eval("WorkDate", "{0:MM/dd/yyyy}") %>'></asp:Label>
                    </ItemTemplate>
                </asp:TemplateField>
                <asp:TemplateField>
                    <HeaderTemplate>
                        <asp:Label ID="Label7" runat="server" Width="200"
Text="Date Created"></asp:Label>
                    </HeaderTemplate>
                    <ItemTemplate>
                        <asp:Label ID="Label8" runat="server" Text='<%#
Eval("DateCreated") %>'></asp:Label>
                    </ItemTemplate>
                </asp:TemplateField>
        </Columns>
        <EmptyDataTemplate>
                <div>No Data Available.</div>
        </EmptyDataTemplate>
    </asp:GridView>
    <asp:EntityDataSource ID="GriffinContextEntityDataSource" runat="server"
        OnContextCreating="GriffinContextEntityDataSource_ContextCreating"
        EntitySetName="WorkDetails" OrderBy="it.SubconID,it.JobID,it.WorkDate"
        AutoGenerateWhereClause="true" Where=""
Include="Work,Work.Job,Work.Subcontractor">
        <WhereParameters>
            <asp:ControlParameter ControlID="ddlJob" Type="Int32"
                Name="JobID" PropertyName="SelectedValue" />
            <asp:ControlParameter ControlID="ddlSubCon" Type="Int32"
                Name="SubconID" PropertyName="SelectedValue" />
        </WhereParameters>
    </asp:EntityDataSource>
    <br />
    <a id="A1" runat="server" href="~/Admin/AdminMenu">Back to Menu</a>
    </asp:Content>
```

### 10.2.24 ViewWork.aspx.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
using Griffin_Drywall.Models;
using System.Web.ModelBinding;
using System.Data.Entity.Infrastructure;

namespace Griffin_Drywall.Admin
{
    public partial class ViewWork : System.Web.UI.Page
    {
        private GriffinContext db = new GriffinContext();

        protected void Page_Load(object sender, EventArgs e)
        {
            if (!IsPostBack)
            {
                BindJobs();
                BindSubcontractors();
            }
        }

        private void Page_PreRender()
        {

        }

        protected void BindJobs()
        {
            var jobs = from j in db.Jobs
                       select new { j.JobID, j.JobName };
            ddlJob.DataSource = jobs.ToList();
            ddlJob.DataTextField = "JobName";
            ddlJob.DataValueField = "JobID";
            ddlJob.DataBind();
            ddlJob.Items.Insert(0, new ListItem(""));
        }

        protected void BindSubcontractors()
        {
            var subcontractors = from s in db.Subcontractors
                                 select new { s.SubconID, s.SubconName };
            ddlSubCon.DataSource = subcontractors.ToList();
            ddlSubCon.DataTextField = "SubconName";
            ddlSubCon.DataValueField = "SubconID";
            ddlSubCon.DataBind();
            ddlSubCon.Items.Insert(0, new ListItem(""));
        }

        protected void GriffinContextEntityDataSource_ContextCreating(object
sender, EntityDataSourceContextCreatingEventArgs e)
        {
            var db = new GriffinContext();
            e.Context = (db as IObjectContextAdapter).ObjectContext;
        }

        protected void ddlSubCon_SelectedIndexChanged(object sender, EventArgs e)
        {
            GridView1.PageIndex = 0;
        }

        protected void ddlJob_SelectedIndexChanged(object sender, EventArgs e)
        {
            GridView1.PageIndex = 0;
        }


    }
}
```

### 10.2.25 Web.config

```
<?xml version="1.0"?>
<configuration>
  <system.web>
    <authorization>
      <allow roles="Administrator"/>
      <deny users="*"/>
    </authorization>
  </system.web>
</configuration>
```

## 10.3 Subcon Page

### 10.3.1 AddWorkEntry.aspx

```
<%@ Page Title="" Language="C#" MasterPageFile="~/Site.Master"
AutoEventWireup="true" CodeBehind="AddWorkEntry.aspx.cs"
Inherits="Griffin_Drywall.Subcon.AddWorkEntry" %>
<asp:Content ID="Content1" ContentPlaceHolderID="HeadContent" runat="server">
</asp:Content>
<asp:Content ID="Content2" ContentPlaceHolderID="FeaturedContent" runat="server">
</asp:Content>
<asp:Content ID="Content3" ContentPlaceHolderID="MainContent" runat="server">
    <article class="projects">
       <h1>Add Work Entry</h1>
        <p>

        </p>
   </article>
    <article>
        <div>
            <h1>Previous Entries</h1>
        </div>
    </article>
     <div style="clear: both"></div>
    <div style="vertical-align: top; left: auto; overflow: auto; width: auto;">
        <asp:GridView ID="GridView1" runat="server" AutoGenerateColumns="false"
            CssClass="mGrid" AlternatingRowStyle-CssClass="alt"
            PagerStyle-CssClass="pgr">
            <Columns>
                <asp:TemplateField>
                    <HeaderTemplate>
                        <asp:Label ID="Label1" runat="server" Text="Job
Name"></asp:Label>
                    </HeaderTemplate>
                    <ItemTemplate>
                        <asp:Label ID="Label2" runat="server" Text='<%#
Eval("JobName") %>'></asp:Label>
                    </ItemTemplate>
                </asp:TemplateField>
                <asp:TemplateField>
                    <HeaderTemplate>
                        <asp:Label ID="Label3" runat="server" Text="Number of
Frames"></asp:Label>
                    </HeaderTemplate>
                    <ItemTemplate>
                        <asp:Label ID="Label4" runat="server" Text='<%#
Eval("NumberofFrames") %>'></asp:Label>
                    </ItemTemplate>
                </asp:TemplateField>
                <asp:TemplateField>
                    <HeaderTemplate>
                        <asp:Label ID="Label5" runat="server" Text="Work
Date"></asp:Label>
                    </HeaderTemplate>
                    <ItemTemplate>
                        <asp:Label ID="Label6" runat="server" Text='<%#
Eval("WorkDate", "{0:MM/dd/yyyy}") %>'></asp:Label>
                    </ItemTemplate>
                </asp:TemplateField>
            </Columns>
            <EmptyDataTemplate>
                <div>There are no previous entries.  Please contact the
adminstrator if this is incorrect. Thank you.</div>
            </EmptyDataTemplate>
        </asp:GridView>
        <asp:Repeater ID="rptPager" runat="server">
            <ItemTemplate>
                <asp:LinkButton ID="lnkPage" runat="server"
                    Text='<%#Eval("Text") %>'
                    CommandArgument='<%#Eval("Value") %>'
                    Enabled='<%#Eval("Enabled") %>'
                    OnClick="Page_Changed"
                    ForeColor="#267CB2"
                    Font-Bold="true" />
            </ItemTemplate>
        </asp:Repeater>
        </div>
    <article>
        <div>
            <h1>Add New Entry</h1>
            <p>
                Please select the correct job, the number of frames used and the
date which the work was done then hit submit.
            </p>
            <p>
                If there are no jobs listed then you have not been assign a job.
Please contact the adminstrator to assign you a job.
            </p>
        </div>
    </article>
    <div style="clear: both"></div>
    <asp:Label ID="lblMessage" runat="server" Text="" ForeColor="Red"></asp:Label>
    <table>
        <tr>
            <td>Job Name</td>
            <td>Number of Frames</td>
            <td>Work Date</td>
        </tr>
        <tr>
            <td>
                <asp:DropDownList ID="ddlJobs" runat="server"></asp:DropDownList>
                <asp:RequiredFieldValidator ID="valJobs"
ControlToValidate="ddlJobs" ValidationGroup="newGrp"
                    ErrorMessage="Job Name is a required field." CssClass="field-
validation-error" runat="server">*</asp:RequiredFieldValidator>
            </td>

            <td class="requestPad">
                <asp:TextBox class="txtFrames" ID="txtFrames" Width="150"
runat="server" />
                <asp:RequiredFieldValidator ID="valFrames"
ControlToValidate="txtFrames" ValidationGroup="newGrp"
                    ErrorMessage="Enter the Number of Frames" CssClass="field-
validation-error" runat="server">*</asp:RequiredFieldValidator>
                <asp:CompareValidator ID="CompareValidator1" runat="server"
Operator="DataTypeCheck" Type="Integer" ValidationGroup="newGrp"
```

```
                            ControlToValidate="txtFrames" ErrorMessage="Frames
must be a whole number" >*</asp:CompareValidator>
            </td>
            <td class="requestPad">
                <asp:TextBox class="txtWorkDate" ID="txtWorkDate" Width="150"
runat="server" />
                <asp:RequiredFieldValidator ID="valWorkDate"
ControlToValidate="txtWorkDate" ValidationGroup="newGrp"
                    ErrorMessage="Select a Work Date" CssClass="field-validation-
error" runat="server">*</asp:RequiredFieldValidator>
                <asp:CompareValidator
                    ID="dateValidator1" runat="server"
                    Type="Date"
                    Operator="DataTypeCheck"
                    ControlToValidate="txtWorkDate"
                    ValidationGroup="newGrp"
                    ErrorMessage="Please enter a valid date."
                    ForeColor="Red" SetFocusOnError="True">*
                </asp:CompareValidator>
                <ajaxToolkit:CalendarExtender
                    ID="CalendarExtender1"
                    TargetControlID="txtWorkDate"
                    runat="server" />
            </td>
        </tr>
        <tr>
            <td>
                <asp:Button ID="submitBtn" Text="Submit" Width="80" runat="server"
OnClick="SubmitBtn_Click" />
                <asp:Button ID="resetBtn" Text="Reset" Width="60"
CausesValidation="false"
                    OnClick="ResetBtn_Click" runat="server" />
            </td>
        </tr>
    </table>
    <asp:ValidationSummary CssClass="field-validation-error"
ValidationGroup="newGrp" ID="valSum" runat="server" />
</asp:Content>
```

### 10.3.2 AddWorkEntry.aspx.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Diagnostics;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Data;
using System.Data.Entity.Validation;
using System.Configuration;
using System.Collections;
using System.Net.Mail;
using System.Web.Configuration;
using System.Web.Security;
using System.Web.UI.WebControls.WebParts;
using System.Web.UI.HtmlControls;
using Griffin_Drywall.Models;
using System.Web.ModelBinding;
```

```
namespace Griffin_Drywall.Subcon
{
    public partial class AddWorkEntry : System.Web.UI.Page
    {
        public string userName;
        public const string UserSessionKey = "UserId";
        public Guid userId;
        private GriffinContext db = new GriffinContext();

        protected void Page_Load(object sender, EventArgs e)
        {
            if (!IsPostBack)
            {
                BindGrid(0);
            }
            lblMessage.Text = "";
        }

        void BindGrid(int pageIndex)
        {
            userName = GetUserId();
            Guid g = new
Guid(Membership.GetUser(userName).ProviderUserKey.ToString());
            userId = g;

            using (var db = new Griffin_Drywall.Models.GriffinContext())
            {
                int totalRecords = db.WorkDetails.Count();
                int pageSize = 10;
                if (pageIndex < 0)
                { pageIndex = 0; }
                int startRow = pageIndex * pageSize;

                if (db.WorkDetails.Count() > 0)
                {
                    var query = (from s in
db.Subcontractors.Include("Jobs").Include("Works").Include("WorkDetails")
                                 from w in s.Works
                                 from wd in w.WorkDetails
                                 where s.UserID == userId
                                 orderby w.JobID, wd.WorkDate
                                 select new
                                 {
                                     w.Job.JobName,
                                     wd.NumberofFrames,
                                     wd.WorkDate
                                 }).Skip(startRow).Take(pageSize);

                    GridView1.DataSource = query.ToList();
                    GridView1.DataBind();
                    BindPager(totalRecords, pageIndex, pageSize);
                }
            }
        }

        protected void Page_Changed(object sender, EventArgs e)
        {
            int pageIndex = Convert.ToInt32(((sender as
LinkButton).CommandArgument));
            GridView1.PageIndex = pageIndex;
            BindGrid(pageIndex - 1);
        }

        public string GetUserId()
        {
            if (HttpContext.Current.Session[UserSessionKey] == null)
            {
                if
(!string.IsNullOrWhiteSpace(HttpContext.Current.User.Identity.Name))
                {
                    HttpContext.Current.Session[UserSessionKey] =
HttpContext.Current.User.Identity.Name;
                }
                else
                {
                    // Generate a new random GUID using System.Guid class.
                    Guid tempUserId = Guid.NewGuid();
                    HttpContext.Current.Session[UserSessionKey] =
tempUserId.ToString();
                }
            }
            return HttpContext.Current.Session[UserSessionKey].ToString();
        }



        private void Page_PreRender()
        {
            BindJobs();
        }

        protected override void OnPreRender(EventArgs e)
        {
            base.OnPreRender(e);

            if (GridView1.Rows.Count == 0)
                GridView1.CssClass = "empty";
        }

        protected void BindJobs()
        {
            userName = GetUserId();
            Guid g = new
Guid(Membership.GetUser(userName).ProviderUserKey.ToString());
            userId = g;

            var jobs = from s in
db.Subcontractors.Include("Jobs").Include("Works")
                       from w in s.Works
                       where s.UserID == userId
                       select new { w.Job.JobID, w.Job.JobName };
            ddlJobs.DataSource = jobs.ToList();
            ddlJobs.DataTextField = "JobName";
            ddlJobs.DataValueField = "JobID";
            ddlJobs.DataBind();
            ddlJobs.Items.Insert(0, new ListItem(""));
        }

        private void BindPager(int totalRecordCount, int currentPageIndex, int
pageSize)
        {
            double getPageCount = (double)((decimal)totalRecordCount /
(decimal)pageSize);
            int pageCount = (int)Math.Ceiling(getPageCount);
            List<ListItem> pages = new List<ListItem>();
            if (pageCount > 1)
            {
                pages.Add(new ListItem("FIRST", "1", currentPageIndex > 1));
                for (int i = 1; i <= pageCount; i++)
                {
                    pages.Add(new ListItem(i.ToString(), i.ToString(), i !=
currentPageIndex + 1));
                }
                pages.Add(new ListItem("LAST", pageCount.ToString(),
currentPageIndex < pageCount - 1));
            }

            rptPager.DataSource = pages;
            rptPager.DataBind();
        }

        protected void SubmitBtn_Click(object sender, EventArgs e)
        {
            if (Page.IsValid)
            {
                DateTime dtWorkDate = DateTime.Parse(txtWorkDate.Text);
                userName = GetUserId();
                Guid g = new
Guid(Membership.GetUser(userName).ProviderUserKey.ToString());
                userId = g;

                try
                {
                    var id = Guid.NewGuid().ToString();
                    var workDetail = new WorkDetail
                    {
                            JobID = Convert.ToInt32(ddlJobs.SelectedValue),
                            SubconID = Convert.ToInt32((from s in
db.Subcontractors where s.UserID == userId select s.SubconID).FirstOrDefault()),
                            WorkDate = dtWorkDate,
                            NumberofFrames = Convert.ToInt32(txtFrames.Text),
                            DateCreated = DateTime.Now
                    };

                    db.WorkDetails.Add(workDetail);
                    db.SaveChanges();
                    lblMessage.Text = "Add successful";
                    BindGrid(0);

                }
                catch (DbEntityValidationException dbEx)
                {
                    foreach (var validationErrors in dbEx.EntityValidationErrors)
                    {
                        foreach (var validationError in
validationErrors.ValidationErrors)
                        {
                            System.Diagnostics.Trace.TraceInformation("Property:
{0} Error: {1}", validationError.PropertyName, validationError.ErrorMessage);
                        }
                    }
                }

            }
        }
        protected void ResetBtn_Click(object sender, EventArgs e)
        {
            txtFrames.Text = "";
            txtWorkDate.Text = "";
            ddlJobs.ClearSelection();
        }
    }
}
```

### 10.3.3 Web.config

```
<?xml version="1.0"?>
<configuration>
  <system.web>
    <authorization>
      <allow roles="SubContractor"/>
      <deny users="*"/>
    </authorization>
  </system.web>
</configuration>
```

## 10.4 Content

### 10.4.1 Site.css

```
html {
    background-color: #FFFFFF;
}

html, body, form {
    height:100%;
    margin: 0;
    padding: 0;
}

body {
    /*background-color: #ffff;
    border-top: solid 10px #000;
    color: #333;
    font-size: .85em;
    font-family: "Segoe UI", Verdana, Helvetica, Sans-Serif;
    background: url("../Images/header-slice.jpg");
    background-color: #fff;
    background-repeat: repeat-x;*/
    font-family: "Segoe UI", Verdana, Helvetica, Sans-Serif;
    min-width: 980px;
}

a {
    color: #333;
    outline: none;
    padding-left: 3px;
    padding-right: 3px;

}

    a:link, a:visited,
    a:active, a:hover {
        color: #333;
    }
    a:link {
        text-decoration: none;
    }

    a:visited {
        text-decoration: none;
    }

    a:active {
        text-decoration: none;
    }

    a:hover {
        color: #000;
        text-decoration: underline;
    }

    a.navbtn {
        color: none;
        outline: none;
        padding-left: 0px;
        padding-right: 0px;
        text-decoration: none;
    }

header, footer, hgroup,
nav, section {
    display: block;
}

mark {
    background-color: #a6dbed;
    padding-left: 5px;
    padding-right: 5px;
}

.float-left {
    float: left;
}

.float-right {
    float: right;
}

.clear-fix:after {
    content: ".";
    clear: both;
    display: block;
    height: 0;
    visibility: hidden;
}

h1, h2, h3,
h4, h5, h6 {
    color: #000;
    margin-bottom: 0;
    padding-bottom: 0;
}

h1 {
    font-size: 2em;
}

h2 {
    font-size: 1.75em;
}

h3 {
    font-size: 1.2em;
}

h4 {
    font-size: 1.1em;
}

h5, h6 {
    font-size: 1em;
```

```
}

    h5 a:link, h5 a:visited, h5 a:active {
        padding: 0;
        text-decoration: none;
    }

/* main layout
----------------------------------------------------------*/
.content-wrapper {
    margin: 0 auto;
    max-width: 980px;
}

.content-wrapper2 {
    margin: 0 auto;
    max-width: 865px;
    position:relative;
    right:10px;
}

#adhocMenu {
    float: left;
    width: 10%;
}

    #adhocMenu a.adminBtn {
        background-position: center;
        background-image: url("../Images/admin_btn.png");
        border-radius: 25px 25px 0px 0px;
        display: block;
        height: 21px;
        width: 86px;
        border-top:solid;
        position:relative;
        bottom:3px;
        left:5px;
    }

    #adhocMenu a.adminBtn:hover {
        background-image:url("../Images/Rollover/admin_btn.png");
        border-color:#333333;
    }

#adhocMenu1 {
    float: left;
    width: 10%;
}

#adhocMenu1 a.adminBtn {
        background-position: center;
        background-image: url("../Images/subcon_btn.png");
        border-radius: 25px 25px 0px 0px;
        display: block;
        height: 21px;
        width: 175px;
        border-top:solid;
        position:relative;
        bottom:3px;
        left:20px;
    }

    #adhocMenu1 a.adminBtn:hover {
        background-image:url("../Images/Rollover/subcon_btn.png");
        border-color:#333333;
    }


#wrapper{
       /*background-position: center;
        width: 1200px;
        margin: 0 auto -67px;
        height: 100%;
    height:auto !important;
        */
       width: 100%;
       min-height: 100%;
       text-decoration: none;
    background-color: #badff3;
    background-image: linear-gradient(bottom, rgb(186,223,243) 16%,
rgb(255,255,255) 50%, rgb(186,223,243) 99%);
    background-image: -o-linear-gradient(bottom, rgb(186,223,243) 16%,
rgb(255,255,255) 50%, rgb(186,223,243) 99%);
    background-image: -moz-linear-gradient(bottom, rgb(186,223,243) 16%,
rgb(255,255,255) 50%, rgb(186,223,243) 99%);
    background-image: -webkit-linear-gradient(bottom, rgb(186,223,243) 16%,
rgb(255,255,255) 50%, rgb(186,223,243) 99%);
    background-image: -ms-linear-gradient(bottom, rgb(186,223,243) 16%,
rgb(255,255,255) 50%, rgb(186,223,243) 99%);
    background-image: -webkit-gradient( linear, left bottom, left top, color-
stop(0.16, rgb(186,223,243)), color-stop(0.5, rgb(255,255,255)), color-stop(0.99,
rgb(186,223,243)) );
    min-width: 980px;

}

#top {
    background-position: center;
    background-image:url("../Images/top.png");
    position: relative;
       top:0px;
       width:955px;
       height:16px;
}

#logo {
    background-position: center;
    background-image:url("../Images/logo.png");
       position:relative;
       top:0px;
       width:955px;
       height:154px;
}

#Nav {
       background-position: center;
    background-image:url("../Images/Nav.png");
       position:relative;
       width:44px;
       height:47px;
}

#home-btn {
       background-position: center;
    background-image:url("../Images/home_btn.png");
       position:relative;
       width:158px;
       height:47px;
}

#services-btn {
       background-position: center;
    background-image:url("../Images/services_btn.png");
       position:relative;
       width:158px;
       height:47px;
}

#Nav008 {
       background-position: center;
    background-image:url("../Images/Nav-08.png");
       position:relative;
       width:238px;
       height:47px;
}

#projects-btn {
       background-position: center;
    background-image:url("../Images/projects_btn.png");
       position:relative;
       width:158px;
       height:47px;
}

#contactus-btn {
       background-position: center;
    background-image:url("../Images/contactus_btn.png");
       position:relative;
       width:158px;
       height:47px;
}

#Nav011 {
       background-position: center;
    background-image:url("../Images/Nav-11.png");
       position:relative;
       width:41px;
       height:47px;
}

#Nav-Bottom {
       background-position: center;
    background-image:url("../Images/Nav-Bottom.png");
       position:relative;
       top:0px;
       width:955px;
       height:9px;
}

#main {
    /*background-color: #efeeef;
    clear: both;
    padding-bottom: 35px;*/
    overflow:auto;
    padding-top: 20px;
       padding-bottom: 85px;

}
/*
#push {
    height: 67px;
}
*/
.main-content {
    /*background: url("../Images/body.png") no-repeat;*/
    background-color: #dcd9d9;
    background-image: linear-gradient(bottom, rgb(200,200,200) 32%,
rgb(220,217,217) 66%);
    background-image: -o-linear-gradient(bottom, rgb(200,200,200) 32%,
rgb(220,217,217) 66%);
    background-image: -moz-linear-gradient(bottom, rgb(200,200,200) 32%,
rgb(220,217,217) 66%);
    background-image: -webkit-linear-gradient(bottom, rgb(200,200,200) 32%,
rgb(220,217,217) 66%);
    background-image: -ms-linear-gradient(bottom, rgb(200,200,200) 32%,
rgb(220,217,217) 66%);
    background-image: -webkit-gradient( linear, left bottom, left top, color-
stop(0.32, rgb(200,200,200)), color-stop(0.66, rgb(220,217,217)) );
    border: 2px solid;
    border-radius: 25px;
    -moz-border-radius: 25px;
    padding: 10px;
}

    .featured + .main-content {
         /*background: url("../Images/heroAccent.png") no-repeat;*/
    }

header .content-wrapper {

}

footer {
   /* clear: both;    REMEMBER TO TURN THIS BACK ON*/
    position:relative;
    width: 980px;
       height:67px;
    margin-top:-67px;
    margin-left:auto;
    margin-right:auto;
    /*margin:-67px auto; */
    font-family: Georgia,Times New Roman,serif;
```

```
}

/*Opera Fix*/
body:before {
       content:"";
       height:100%;
       float:left;
       width:0;
       margin-top:-32767px;
}
/*
.footer-note {
    text-align: center;
    width: 50%px;
    color:white;
    position: relative;
    top: 25px;
*/
#footer-bottom {
    width: 955px;
    height:67px;
    background: url("../Images/footer.png") no-repeat;
    position:relative;
}

#foot {
    padding: 20px 0 0;
    width:90%;
    margin-left:auto;
    margin-right:auto;
}

    #foot .footer-note {
        text-align: center;
        width: 80%;
        color: white;
        position: relative;
        top: 10px;
        left:100px;
    }

#foot .left {
    float: left;
    margin-right: 20px;
}

#foot .right {
    float: right;
}


#foot p{
    margin:0;
}

foot .facebookicon {
    margin-left: 30px;
    width: 25%;
```

```
}



/* home page
-----------------------------------------------------------*/
.round-img {

    left: 20px;
    position: relative;
    top: 40px;
    border-width: 1px;
    margin-bottom: 15px;
}
.round-img2 {
    border: solid;
    border-radius: 25px 25px 25px 25px;
    border-width: 1px;
    margin-bottom: 15px;
}

#request-btn {
    position:relative;
    left: 125px;
}

article p {
        text-indent:20px;
       margin:10px;
}

/* services page
-----------------------------------------------------------*/
.serviceMenu {
    width:80%;
    margin-right: auto;
    margin-left: auto;
}
.serviceDetails {
    width:100%;
    margin: 0;
}

.serviceDetails h1 {
    border-bottom: 1px solid #858585;
    color: #444444;
    font-size: 1.3em;
    font-weight: 300;
}

.serviceDetails h2 {
    color: #444444;
    font-size: 1.4em;
    font-weight: 300;
}

.serviceDetails table {
    width:100%;
    margin:0;
}
    .serviceDetails #serviceDesciption {
        width:90%;
    }

    .serviceDetails #serviceDesciption p {
       text-indent:20px;
       margin:10px;
    }

    .serviceDetails img {
        float: right;
        border-radius: 25px;
        -moz-border-radius: 25px;
        padding: 5px;
        position: relative;
        right: 20px;
        margin-top: 20px;
        margin-bottom: 10px;
    }

/* projects page
-----------------------------------------------------------*/
.tbProjects{
    margin:0;
    width: 291px;
}

.tbProjects td{
    text-align:center;
}

.tbPad {
    padding-bottom: 15px;
}

.tbProjects img{
    border: solid;
    border-radius: 25px 25px 25px 25px;
    border-width: 1px;
}

/* projects details page
-----------------------------------------------------------*/
#galleria{
    width: 500px;
    height: 400px;
    margin-top:10px;
    float:left;
}

/* contact us page
-----------------------------------------------------------*/
.contactUs {
    width: 50%;

}
```

```
.Ctitle{
        width: 90%;
    }

.contactUs2 {
    border-right: solid 2px #c8c8c8;
    width: 100%;
}

#contactUsForm {
    width: 90%;
}

#contactUsEmail {
    float:right;
    width: 45%;
}

.contactUsEmail table {
    width: 97%;
}

.contactPad {
    padding-left:5px;
}
 .contactTxtBox {
        width: 260px;
    }

.contactUsEmail fieldset {
    border: thin solid;
    padding-left: 1em;
}

.contactUsEmail fieldset legend {
   display: block;
}

#contactMSG {
    width: 90%;
    margin-left: 45px;
}
.contactMsg {
    position: relative;
    top: 52px;
    padding-left: 25px;
}

.contactUs2 header{
    font-weight:bold;
}

.contactUs2 p{
    margin:0;
}
/* request bid page
----------------------------------------------------------*/
```

```
.requestBid {
    width: 50%;

}

#requestBidEmail {
    width: 100%;
}

.requestBidEmail table {
    width: 97%;
}

#contactDetails {
    float:left;
    width:45%;
}

#descWork {
    float:right;
    width: 53%;
}

#descWork table{
    margin:0;
}

.requestPad {
    padding-left:5px;
}
 .requestTxtBox {
        width: 260px;
    }

.requestBidEmail fieldset {
    border: thin solid;
    padding-left: 1em;
}

.requestBidEmail fieldset legend {
   display: block;
}

#requestMSG {
     margin: auto;
    width: 100%;
    text-align: center;
    color: blue;
}

.requestBid2 header{
    font-weight:bold;
}

.requestBid2 p{
    margin:0;
}
```

```
.serviceChkBox label {
        display: inline;
        padding: 5px;
        font-weight: normal;
    }

.serviceType {
    width:80%;
}

.empty, .empty td {
    border: 0;
    background: none;
    margin: 0;
    padding: 0;
    text-align: center;
    color: blue;
    width: 80%;
    margin: auto;
    background: none;
}

/* admin page
----------------------------------------------------------*/
.adminList {
    list-style-type: none;
    margin: 0;
    padding-left: 20px;
}

    .adminList a {

    }

/* site title
----------------------------------------------------------*/
.site-title {

    font-family: Rockwell, Consolas, "Courier New", Courier, monospace;
    font-size: 2.3em;
    margin: 0;
}

.site-title a, .site-title a:hover, .site-title a:active {
    background: none;
    color: #c8c8c8;
    outline: none;
    text-decoration: none;
}


/* login
----------------------------------------------------------*/
#login {
    display: block;
    font-size: 1em;
    font-weight:600;
    margin: 0 0 10px;
    text-align: right;
}

    #login a {
        margin-left: 10px;
        margin-right: 3px;
        padding: 2px 3px;
        text-decoration: none;
    }

        #login a:hover {
            text-decoration:underline;
        }

    #login a.username {
        background: none;
        margin-left: 0px;
        text-decoration: underline;
    }

    #login ul {
        margin: 0;
    }

    #login li {
        display: inline;
        list-style: none;
    }

    #login li a:hover{
        text-decoration:underline;
    }


/* menu
----------------------------------------------------------*/
ul#menu {
    font-size: 1.3em;
    font-weight: 600;
    margin: 0 0 5px;
    padding: 0;
    text-align: right;
}

    ul#menu li {
        display: inline;
        list-style: none;
        padding-left: 15px;
    }

        ul#menu li a {
            background: none;
            color: #999;
            text-decoration: none;
        }

        ul#menu li a:hover {
            color: #333;
            text-decoration: none;
        }


/* page elements
----------------------------------------------------------*/
/* featured */
.featured {

}

    .featured .content-wrapper {

    }

        .featured hgroup.title h1, .featured hgroup.title h2 {
            color: #fff;
        }

        .featured p {
            font-size: 1.1em;
        }

/* page titles */
hgroup.title {
    margin-bottom: 10px;
}

hgroup.title h1, hgroup.title h2 {
    display: inline;
}

hgroup.title h2 {
    font-weight: normal;
    margin-left: 3px;
}

/* features*/
section.feature {
    width: 300px;
    float: left;
    padding: 10px;
}

/* ordered list */
ol.round {
    list-style-type: none;
    padding-left: 0;
}

    ol.round li {
        margin: 25px 0;
        padding-left: 45px;
    }

        ol.round li.zero {
            background: url("../Images/orderedList0.png") no-repeat;
        }
        ol.round li.one {
            background: url("../Images/orderedList1.png") no-repeat;
        }

        ol.round li.two {
            background: url("../Images/orderedList2.png") no-repeat;
        }

        ol.round li.three {
            background: url("../Images/orderedList3.png") no-repeat;
        }

        ol.round li.four {
            background: url("../Images/orderedList4.png") no-repeat;
        }

        ol.round li.five {
            background: url("../Images/orderedList5.png") no-repeat;
        }

        ol.round li.six {
            background: url("../Images/orderedList6.png") no-repeat;
        }

        ol.round li.seven {
            background: url("../Images/orderedList7.png") no-repeat;
        }

        ol.round li.eight {
            background: url("../Images/orderedList8.png") no-repeat;
        }

        ol.round li.nine {
            background: url("../Images/orderedList9.png") no-repeat;
        }

/* content */
article {
    float: left;
    width: 60%;
}

article.projects {
     width: 80%;
    text-align: center;
    margin-left: auto;
    margin-right: auto;
    float: none;
}

article.reqBid {
     width: 80%;
    text-align: center;
    margin-left: auto;
    margin-right: auto;
    float: none;
}
```

```
article h1 {
    border-bottom: 1px solid #858585;
    color: #444444;
    font-size: 1.3em;
    font-weight: 300;
}

article h2 {
    color: #444444;
    font-size: 1.4em;
    font-weight: 300;
}

aside {
    float: right;
    width: 25%;
}

    aside ul {
        list-style: none;
        padding: 0;
    }

        aside ul li {
            background: url("../Images/bullet.png") no-repeat 0 50%;
            padding: 2px 0 2px 20px;
        }

.label {
    font-weight: 700;
}

/* login page */
#loginForm {
    border-right: solid 2px #c8c8c8;
    float: left;
    width: 55%;
}

    #loginForm .validation-error {
        display: block;
        margin-left: 15px;
    }

#socialLoginForm {
    margin-left: 40px;
    float: left;
    width: 40%;
}

    #socialLoginForm h2 {
        margin-bottom:  5px;
    }

fieldset.open-auth-providers {
    margin-top: 15px;
```

```
}

    fieldset.open-auth-providers button {
        margin-bottom: 12px;
    }

/* contact */
.contact h3 {
    font-size: 1.2em;
}

.contact p {
    margin: 5px 0 0 10px;
}

.contact iframe {
    border: 1px solid #333;
    margin: 5px 0 0 10px;
}

/* forms */
fieldset {
    border: none;
    margin: 0;
    padding: 0;
}

    fieldset legend {
        display: none;
    }

    fieldset ol {
        padding: 0;
        list-style: none;
    }

        fieldset ol li {
            padding-bottom: 5px;
        }

.RegForm ol, ol li{
    margin:0;
    padding:0;
}

    label {
        display: block;
        font-size: 1.2em;
        font-weight: 600;
    }

    .userRoles label {
        display: inline;
        padding: 5px;
    }

    label.checkbox {
        display: inline;
    }

    input, textarea {
        border: 1px solid #e2e2e2;
        background: #fff;
        color: #333;
        font-size: 1.2em;
        margin: 5px 0 6px 0;
        padding: 5px;
        width: 300px;
    }

    textarea {
        font-family: inherit;
        width: 500px;
    }

        input:focus, textarea:focus {
            border: 1px solid #7ac0da;
        }

        input[type="checkbox"] {
            background: transparent;
            border: inherit;
            width: auto;
        }

    input[type="submit"],
    input[type="button"],
    button {
        background-color: #d3dce0;
        border: 1px solid #ffffff;
        border-radius: 25px;
        -moz-border-radius: 25px;
        cursor: pointer;
        font-size: 1.2em;
        font-weight: 600;
        padding: 3px;
        margin-right: 8px;
        width: auto;
    }
input[type="submit"]:hover,
input[type="button"]:hover,
button:hover {
    background-color:#adadad;
    background: url('../Images/btn-rollover.png') no-repeat;
}

    td input[type="submit"],
    td input[type="button"],
    td button {
        font-size: 1em;
        padding: 4px;
        margin-right: 4px;
    }

/* info and errors */
.message-info {
    border: 1px solid;
    clear: both;
    padding: 10px 20px;
}

.message-error {
    clear: both;
    color: #fe0000;
    font-size: 1.1em;
    font-weight: bold;
    margin: 20px 0 10px 0;
}

.message-success {
    color: #7ac0da;
    font-size: 1.3em;
    font-weight: bold;
    margin: 20px 0 10px 0;
}

.error {
    color: #fe0000;
}

/* styles for validation helpers */
.field-validation-error {
    color: #fe0000;
    font-weight: bold;
}

.field-validation-valid {
    display: none;
}

input.input-validation-error {
    border: 1px solid #fe0000;
}

input[type="checkbox"].input-validation-error {
    border: 0 none;
}

.validation-summary-errors {
    color: #fe0000;
    font-weight: bold;
    font-size: 1.1em;
}

.validation-summary-valid {
    display: none;
}

/* tables
----------------------------------------------------------*/
table {
    border-collapse: collapse;
    border-spacing: 0;
    margin-top: 0.75em;
    border: 0 none;
}

th {
       font-size: 1.2em;
    text-align: left;
    border: none 0px;
    padding-left: 0;
}

    th a {
        display: block;
        position: relative;

    }

       th a:link, th a:visited, th a:active, th a:hover {
              color: #333;
              font-weight: 600;
              text-decoration: none;
        padding: 0;
       }

       th a:hover {
              color: #000;
       }

    th.asc a, th.desc a {
        margin-right: .75em;
    }

    th.asc a:after, th.desc a:after {
              display: block;
        position: absolute;
        right: 0em;
        top: 0;
        font-size: 0.75em;
       }

       th.asc a:after {
              content: '▲';
       }

       th.desc a:after {
              content: '▼';
       }

td {
    padding: 0.25em 2em 0.25em 0em;
    border: 0 none;
}

tr.pager td {
    padding: 0 0.25em 0 0;
}

table#navagtion-bar {
    border-collapse: collapse;
    margin-top: 0em;
    position: relative;
}

table#navagtion-bar th, td {
    padding: 0;
}

table.list, td.details table.list {
       width: 100%;
       border: 1px solid #000;
}
table.list tr th, td.details table.list tr th {
       text-align: left;
       background-color: #666;
       color: #FFF;
       border-style: none;
       font-size: 80%;
    padding: 2px 10px 2px 2px;
}
    table.list td.valign {
        text-align: center;
        vertical-align: text-top;
    }

table.list tr td, td.details table.list tr td {
       border-bottom: 1px solid #999;
       padding: 1px 1px 1px 1px;
}
table.list tr {
       background-color: #FFC;
}
table.list tr.odd {
       background-color: #CCC;
}

table.webparts tr th {
       background-color: #009;
       padding: 5px;
       color: #FFF;
       font-size: 130%;
       border: 1px solid #009;
}

td.details div table tr td, td.details div table tr td.detailheader
{
       border-style: none;
}
td.detailheader
{
       text-align: right;
       font-weight: bold;
       white-space: nowrap;
       border-style: none;
       padding: 3px 10px 3px 0px;
}
td.details div table
{
	border-style: none;
}
td.details div table tr td.detailitem
{
	width: 100%;
	padding: 3px 10px 3px 0px;
}
td.details
{
	border: 2px solid #009;
	padding: 10px;
	background-color:#EEE;
}
.alert {
	color: #C00;
	font-weight: bold;
}

/* grid
----------------------------------------------------------*/
.mGrid {
    width: 100%;
    background-color: #fff;
    margin: 5px 0 10px 0;
    border: solid 1px #525252;
    border-collapse: collapse;
}

    .mGrid td {
        padding: 2px;
        border: solid 1px #c1c1c1;
        color: #717171;
    }

    .mGrid th {
        padding: 4px 2px;
        color: #fff;
        background: #424242 url("../Images/grd_head.png") repeat-x top;
        border-left: solid 1px #525252;
        font-size: 0.9em;
    }

    .mGrid .alt {
        background: #fcfcfc url("../Images/grd_alt.png") repeat-x top;
    }

    .mGrid .pgr {
        background: #424242 url("../Images/grd_pgr.png") repeat-x top;
    }

        .mGrid .pgr table {
            margin: 5px 0;
        }

        .mGrid .pgr td {
            border-width: 0;
            padding: 0 6px;
            border-left: solid 1px #666;
            font-weight: bold;
            color: #fff;
            line-height: 12px;
        }

        .mGrid .pgr a {
            color: #666;
            text-decoration: none;
        }

            .mGrid .pgr a:hover {
                color: #000;
                text-decoration: none;
            }
```

# 11. Glossary

| Term | Description |
|---|---|
| .NET Framework | The software framework developed by Microsoft. |
| ADO.NET | The set of classes used in .NET applications to provide data access services. |
| API | Application Programming Interface |
| ASP.NET Membership | Built-in service for validating and storing user credentials. |
| CLR | Common Language Runtime |
| Code First | A workflow in the Entity Framework which emphasizes working in code rather than with a model. |
| Code First Migrations | Implements database changes through the usage of code. |
| DAL | Data Access Layer |
| Data Annotations | Adds the ability to configure Code First conventions within the POCO classes. |
| DbContext API | Introduced with Code First, this API is used as an alternative to ObjectContext which is seen in the Model First approach. |
| DBMS | Database Management System |
| Eager Loading | An entity will initializes all related entities the first time the entity is accessed. |
| EDM | Entity Data Model |
| EF | Entity Framework |
| Entity Framework | The object-relational mapper used by .NET developers to work with relational data using domain-specific objects. |
| Fluent API | Adds the ability to configure Code First conventions within the database context. |
| GUID | Globally Unique Identifier |
| IDE | Integrated Development Environment |
| Impedance Mismatch | Refers to the differences between the relational database and object model in object relational mapping. |
| IntelliSense | Stands for “intelligent sense” and is Visual Studio feature to accelerate coding by reducing commonly made mistakes as well as providing helpful development hints. |
| Lazy Loading | Defers initializing an object until the object is needed by the application. In EF, this means an entity or collection of entities are only loaded when their navigational properties are accessed. |
| LINQ | Language-Integrated Query |
| Model First | A workflow in the Entity Framework which emphasizes working in visual model rather than coded classes. |
| ORM | Object Relational Mapper |
| Persistence Ignorance | An object that does not contain any logic that is related to data storage. |
| POCO | Plan Old CLR Objects |
| POCO Classes | The C# equivalent to Plan Old Java Object classes. |

| Term | Description |
|---|---|
| SQL | Structured Query Language |
| Visual Basic | A programming language used in the .NET Framework. |
| Visual C# | A programming language used in the .NET Framework. |
| Visual Studio | Microsoft's integrated development environment for developing .NET applications. |
| VS | Visual Studios |
| XML | Extensible Markup Language |

## Bibliography

*.NET Framework*. (n.d.). Retrieved February 21, 2013, from Wikipedia: http://en.wikipedia.org/wiki/.NET_Framework

*.NET Framework 4.5*. (n.d.). Retrieved February 21, 2013, from MSDN Microsoft: http://msdn.microsoft.com/en-us/library/vstudio/w0x726c2.aspx

*C# 3.0 Features That Support LINQ*. (2008, September). Retrieved March 17, 2013, from MSDN Microsoft: http://msdn.microsoft.com/en-us/library/bb397909(v=vs.90).aspx

*Entity Framework Overview*. (2012, August 2). Retrieved April 4, 2013, from MSDN Microsoft: http://msdn.microsoft.com/en-us/library/bb399567.aspx

*Entity Framework FAQ: Introduction*. (2013). Retrieved January 14, 2013, from Microsoft TechNet: http://social.technet.microsoft.com/wiki/contents/articles/3923.entity-framework-faq-introduction-en-us.aspx

*What's New*. (2013). Retrieved Febraury 21, 2013, from Microsoft: http://www.microsoft.com/visualstudio/eng/whats-new#story-whats-new

*ADO.NET Architecture*. (n.d.). Retrieved February 21, 2013, from MSDN Microsoft: http://msdn.microsoft.com/en-us/library/27y4ybxw.aspx

*ADO.NET Data Provider*. (n.d.). Retrieved Febraury 21, 2013, from Wikipedia: http://en.wikipedia.org/wiki/ADO.NET_data_provider

*ADO.NET Entity Framework At-a-Glance*. (n.d.). Retrieved February 21, 2013, from MSDN Microsoft: http://msdn.microsoft.com/en-us/data/aa937709.aspx

*ADO.NET Overview*. (n.d.). Retrieved Febraury 21, 2013, from MSDN Microsoft: http://msdn.microsoft.com/en-us/library/h43ks021.aspx

*Adobe Dreamweaver CS6*. (n.d.). Retrieved February 21, 2013, from Adobe: http://www.adobe.com/products/dreamweaver.html

Campbell, M. K. (2012, May 3). *Entity Framework and ORMs: Understand the Trade-Offs.* Retrieved Febraury 21, 2013, from DevPro: http://www.devproconnections.com/article/entity-framework/entity-framework-orms-142979

Corporation, M. (2006, June). *The ADO.NET Entity Framework Overview*. Retrieved February 21, 2013, from MSDN: http://msdn.microsoft.com/en-us/library/aa697427%28v=vs.80%29.aspx

Dammani, B. M. (2012, February 15). *Insert, Update, Delete in ASP.NET Gridview With Entity Framework.* Retrieved February 21, 2013, from TechBrij: http://techbrij.com/insert-update-delete-gridview-entity-framework

dpblogs. (2008, March 27). *ADO.NET Entity Framework Performance Comparison*. Retrieved January 14, 2013, from ADO.NET Blogs: http://blogs.msdn.com/b/adonet/archive/2008/03/27/ado-net-entity-framework-performance-comparison.aspx

dpblogs. (2012, Febraury 14). *Sneak Preview: Entity Framework 5.0 Performance Improvements.* Retrieved Febraury 21, 2013, from ADO.NET Blog: http://blogs.msdn.com/b/adonet/archive/2012/02/14/sneak-preview-entity-framework-5-0-performance-improvements.aspx

*Entity Framework.* (n.d.). Retrieved January 2013, 2013, from The Official Microsoft ASP.NET Site: http://www.asp.net/entity-framework

Fowler, M. (2012, May 9). *Martin Fowler on ORM Hate.* Retrieved Febraury 21, 2013, from DZone: http://java.dzone.com/articles/martin-fowler-orm-hate

Griffin, C. (2011). *CIS 623 Database Services Development using Microsoft tools.* Final Project Submission, Computer Information Science.

*Introduction to Membership.* (n.d.). Retrieved February 21, 2013, from MSDN Microsoft: http://msdn.microsoft.com/en-us/library/yh26yfzy%28v=vs.98%29.aspx

Lokuge, S. (2012, December 10). *How to Improve Performance of Entity Framework Query?* . Retrieved February 21, 2013, from Sampath Lokuge Tech Blog: http://sampathloku.blogspot.com/2012/12/how-to-improve-performance-of-entity.html

Microsoft. (2013). *ADO.NET Entity Framework.* Retrieved Jan 14, 2013, from MSDN Microsoft: http://msdn.microsoft.com/en-us/library/bb399572.aspx

Miller, R. (n.d.). *Entity Framework Code First to a New Database.* Retrieved February 21, 2013, from MSDN Microsoft: http://msdn.microsoft.com/en-us/data/jj572366

Miller, R. (n.d.). *Entity Framework Development Workflows.* Retrieved Febraury 21, 2013, from MSDN Microsoft: http://msdn.microsoft.com/en-us/data/jj590134

Mitchell, S. (2006, June). *Creating a Data Access Layer*. Retrieved February 21, 2013, from http://msdn.microsoft.com/en-us/library/aa581778.aspx

Neward, T. (2006, June 26). *The Vietnam of Computer Science.* Retrieved February 21, 2013, from The Blog Ride: http://blogs.tedneward.com/2006/06/26/The+Vietnam+Of+Computer+Science.aspx

*Object-relational mapping*. (n.d.). Retrieved February 21, 2013, from Wikipedia: http://en.wikipedia.org/wiki/Object-relational_mapping

Singh, R. R. (2012, April 9). *An Introduction to Entity Framework for Absolute Beginners.* Retrieved November 28, 2012, from Code Project: http://www.codeproject.com/Articles/363040/An-Introduction-to-Entity-Framework-for-Absolute-B

*Visual Studio.* (n.d.). Retrieved February 21, 2013, from Microsoft: http://www.microsoft.com/visualstudio/eng/office-dev-tools-for-visual-studio