Fall 2015

Secured Client Portal

Krishnakar Mogili
Governors State University

Rajitha Thippireddy
Governors State University

Shobhan Tula
Governors State University

Table of Contents

1 Project Description

1.1 Project Abstract

This project is aimed at developing an online search Portal for the Placement Department of the college. The system is an online application that can be accessed throughout the organization and outside as well with proper login provided. This system can be used as an Online Job Portal for the Placement Department of the college to manage the student information with regards to placement. Students logging should be able to upload their information in the form of a CV. Visitors/Company representatives logging in may also access/search any information put up by Students.

This Approach Rests on

- A strategy where we architect, integrate and manage technology services and solutions - we call it AIM for success.
- A robust offshore development methodology and reduced demand on customer resources.
- A focus on the use of reusable frameworks to provide cost and times benefits.

They combine the best people, processes and technology to achieve excellent results - consistency. We offer customers the advantages of:

Speed

They understand the importance of timing, of getting there before the competition. A rich portfolio of reusable, modular frameworks helps jump-start projects. Tried and tested methodology ensures that we follow a predictable, low - risk path to achieve results. Our track record is testimony to complex projects delivered within and evens before schedule.

Expertise

Our teams combine cutting edge technology skills with rich domain expertise. What's equally important - they share a strong customer orientation that means they actually start by listening to the customer. They're focused on coming up with solutions that serve customer requirements today and anticipate future needs.

Problems in Existing System

- Cannot Upload and Download the latest updates.
- No use of Web Services and Remoting.
- Risk of mismanagement and of data when the project is under development.
- Less Security.
- No proper coordination between different Applications and Users.
- Fewer Users - Friendly.

Development of New System

The development of the new system contains the following activities, which try to automate the entire process keeping in view of the database integration approach.

- User friendliness is provided in the application with various controls.
- The system makes the overall project management much easier and flexible.
- Readily upload the latest updates, allows user to download the alerts by clicking the URL.
- There is no risk of data mismanagement at any level while the project development is under process.
- It provides high level of security with different level of authentication.
- This Project can also run By using Intranet access.

1.2 Competitive Information

- ❖ Organization Profile
- ❖ Software Solutions

Career Path Software Solutions is an IT solution provider for a dynamic environment where business and technology strategies converge. Their approach focuses on new ways of business combining IT innovation and adoption while also leveraging an organization's current IT assets. Their work with large global corporations and new products or services and to implement prudent business and technology strategies in today's environment.

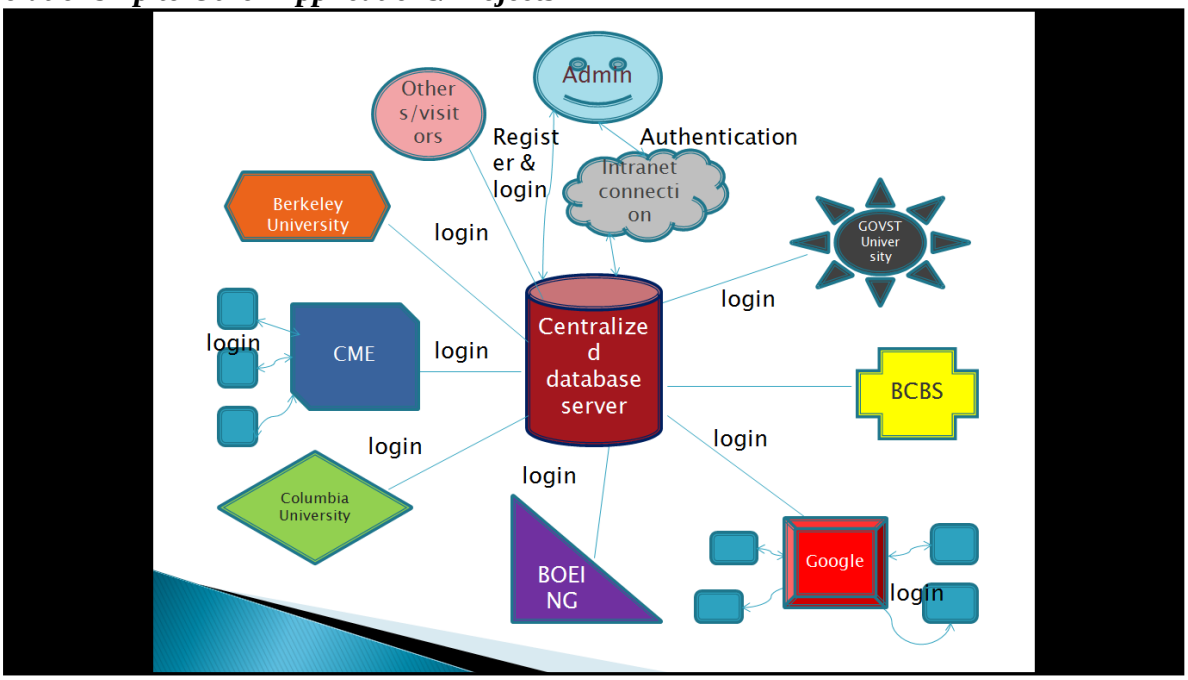
Career Path Range Of Expertise Includes:

- Software Development Services
- Engineering Services
- Systems Integration
- Customer Relationship Management
- Product Development
- Electronic Commerce
- Consulting
- IT Outsourcing

We apply technology with innovation and responsibility to achieve two broad objectives:

- Effectively address the business issues our customers face today.
- Generate new opportunities that will help them stay ahead in the future.

1.3 Relationship to Other Applications/Projects



This Project will use a Centralized database which is maintained by admin can helps to communicate with other applications and projects to update/delete/modify/ share data.

1.4 Assumptions and Dependencies

This Project can run without internet connection, we are using some servlets to run this project by using intranet. All clients will provide their information and while lot of people accessing our website at same time speed and website will still work.

The project has been planned to be having the view of distributed architecture, with centralized storage of the database. The application for the storage of the data has been planned. Using the constructs of MS-SQL Server and all the user interfaces have been designed using the ASP.Net technologies. The database connectivity is planned using the "SQL Connection" methodology. The standards of security and data protective mechanism have been given a big choice for proper usage. The application takes care of different modules and their associated reports, which are produced as per the applicable strategies and standards that are put forwarded by the administrative staff.

The entire project has been developed keeping in view of the distributed client server computing technology, in mind. The specification has been normalized up to 3NF to eliminate all the anomalies that may arise due to the database transaction that are executed by the general users and the organizational administration. The user interfaces are browser specific to give distributed accessibility for the overall system. The internal database has been selected as MS-SQL server 200.The basic constructs of table spaces, clusters and indexes have been exploited to provide higher consistency and reliability for the data storage. The MS-SQL server 200 was a choice as it provides the constructs of high-level reliability and security. The total front end was dominated using the ASP.Net technologies. At all proper levels high care was taken to check that the system manages the data consistency with proper business rules or validations. The database connectivity was planned using the latest "SQL Connection" technology provided by Microsoft Corporation. The authentication and authorization was crosschecked at all the relevant stages. The user level accessibility has been restricted into two zones namely.

1.5 Future Enhancements

- This System being web-based and an undertaking of Cyber Security Division, needs to be thoroughly tested to find out any security gaps.
- A console for the data centre may be made available to allow the personnel to monitor on the sites which were cleared for hosting during a particular period.
- Moreover, it is just a beginning; further the system may be utilized in various other types of auditing operation viz. Network auditing or similar process/workflow based applications

1.6 Definitions and Acronyms

Feasibility Report : It is a document that assesses potential solutions to the business problem or opportunity, and determines which of these are viable for further analysis.

GUI's: graphical user interface or GUI, pronounced /ˈguːi/ ("gooey") is a type of interface that allows users to interact with electronic devices through graphical icons and visual indicators such as secondary notation, as opposed to text-based interfaces, typed command labels or text navigation.

Class Diagram: A class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among objects.

2 Technical Description

In the flexibility of the uses the interface has been developed a graphics concept in mind, associated through a browser interface. The GUI'S at the top level have been categorized as

- Administrative user interface
- The operational or generic user interface

The administrative user interface concentrates on the consistent information that is practically, part of the organizational activities and which needs proper authentication for the data collection. The interfaces help the administrations with all the transactional states like Data insertion, Data deletion and Date updation along with the extensive data search capabilities.

The operational or generic user interface helps the users upon the system in transactions through the existing data and required services. The operational user interface also helps the ordinary users in managing their own information helps the ordinary users in managing their own information in a customized manner as per the assisted flexibilities.

The modules involved are:

This project consists of 6 modules:

- Admin
- Home
- Existing User
- About Us
- Feedback
- Contact Us

Project Instructions

- Based on the given requirements, conceptualize the Solution Architecture. Choose the domain of your interest otherwise develop the application for ultimatedotnet.com. Depict the various architectural components, show interactions and connectedness and show internal and external elements. Design the web services, web methods and database infrastructure needed both and client and server.
- Provide an environment for upgradation of application for newer versions that are available in the same domain as web service target.

Hardware & Software Specifications

S/w

- Microsoft .NET framework 4.0
- Language: C#.Net, Asp.net
- Microsoft Visual Studio 2008IDE or higher
- OS: Microsoft Windows 7 or higher

H/w

- P IV Processor or higher
- 512 MB RAM minimum
- Secondary memory of 20 – 50 GB minimum

Purpose: The main purpose for preparing this document is to give a general insight into the analysis and requirements of the existing system or situation and for determining the operating characteristics of the system.

Scope: This Document plays a vital role in the development life cycle (SDLC) and it describes the complete requirement of the system. It is meant for use by the developers and will be the basic during testing phase. Any changes made to the requirements in the future will have to go through formal change approval process.

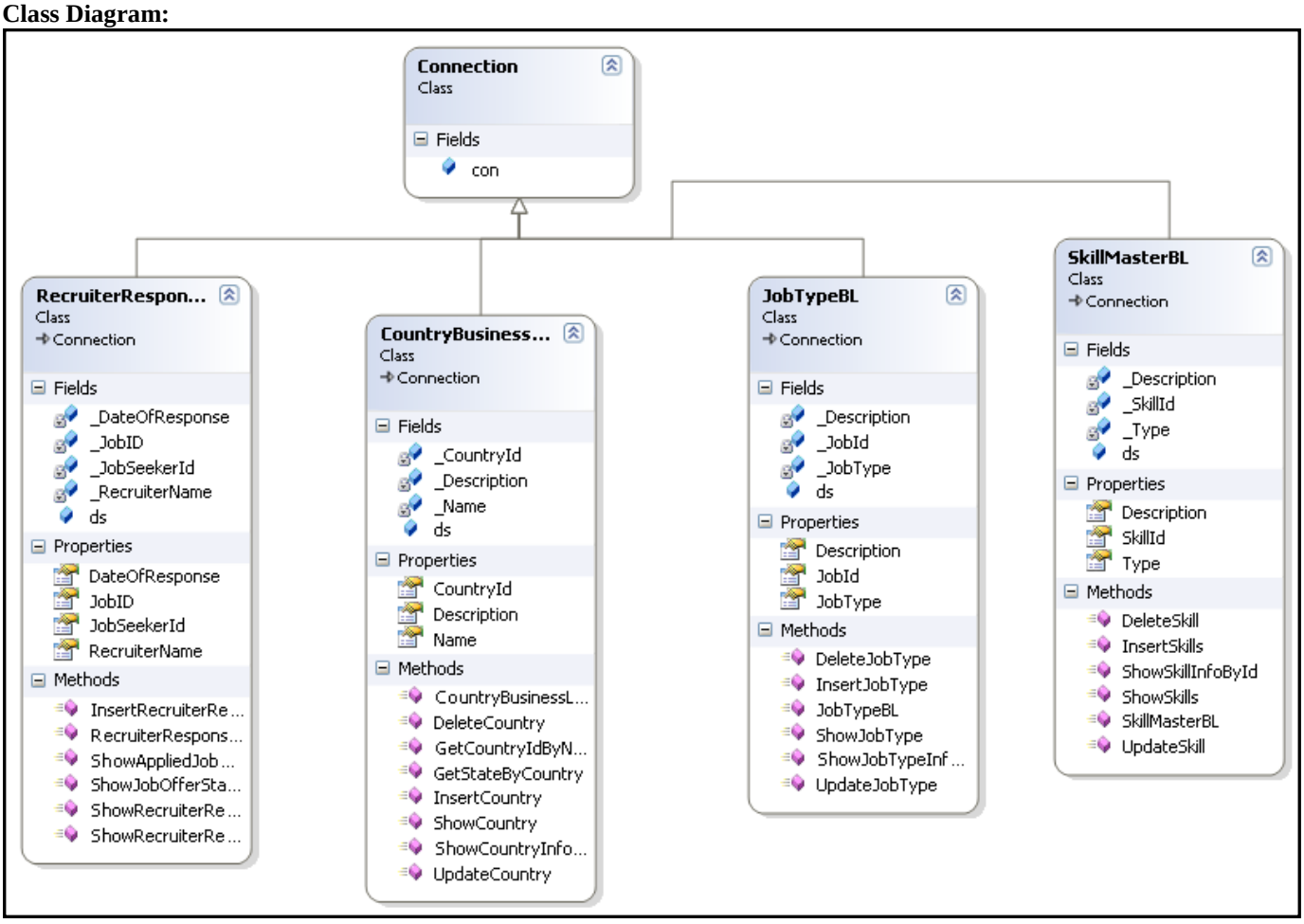
Developers Responsibilities Overview

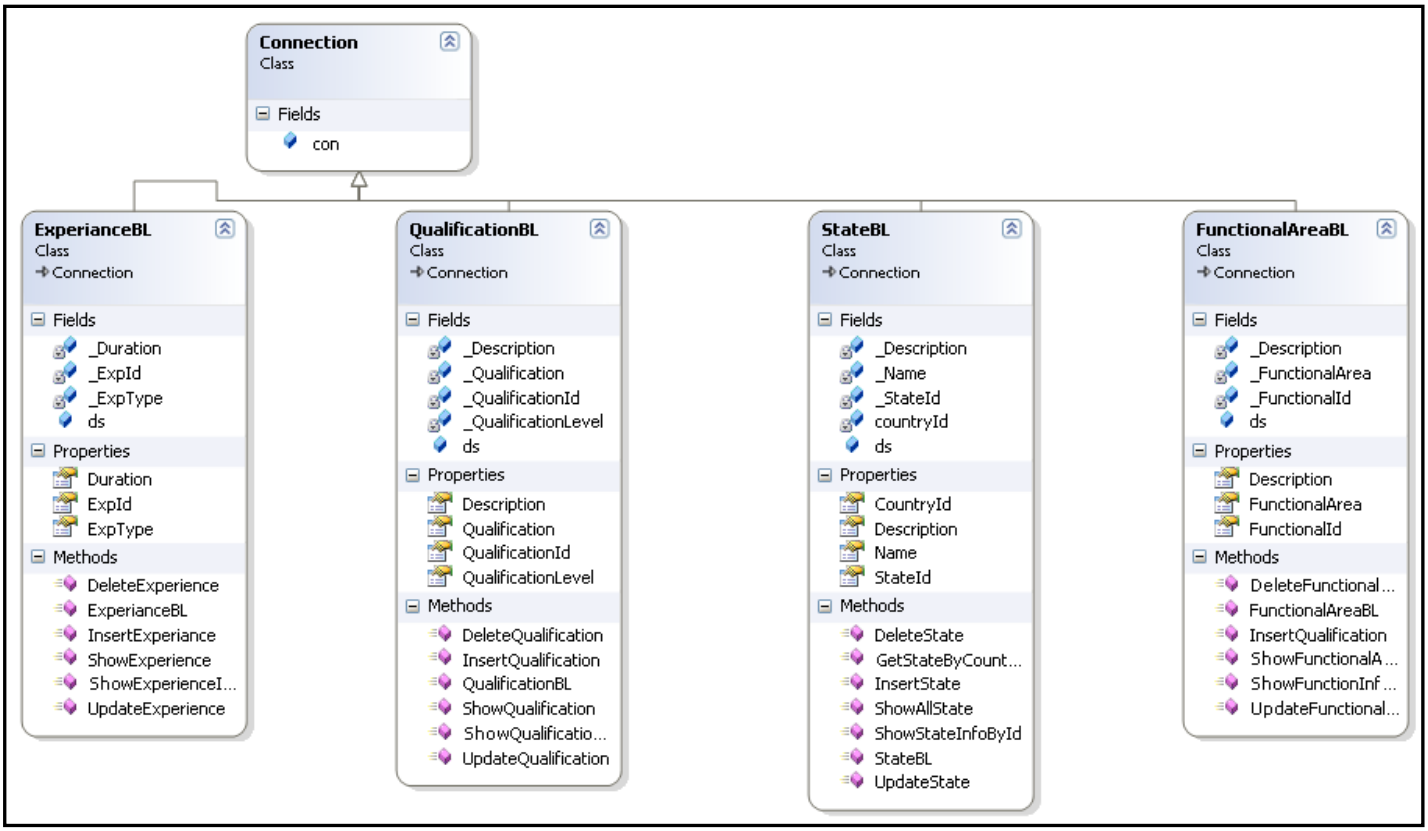
The developer is responsible for:

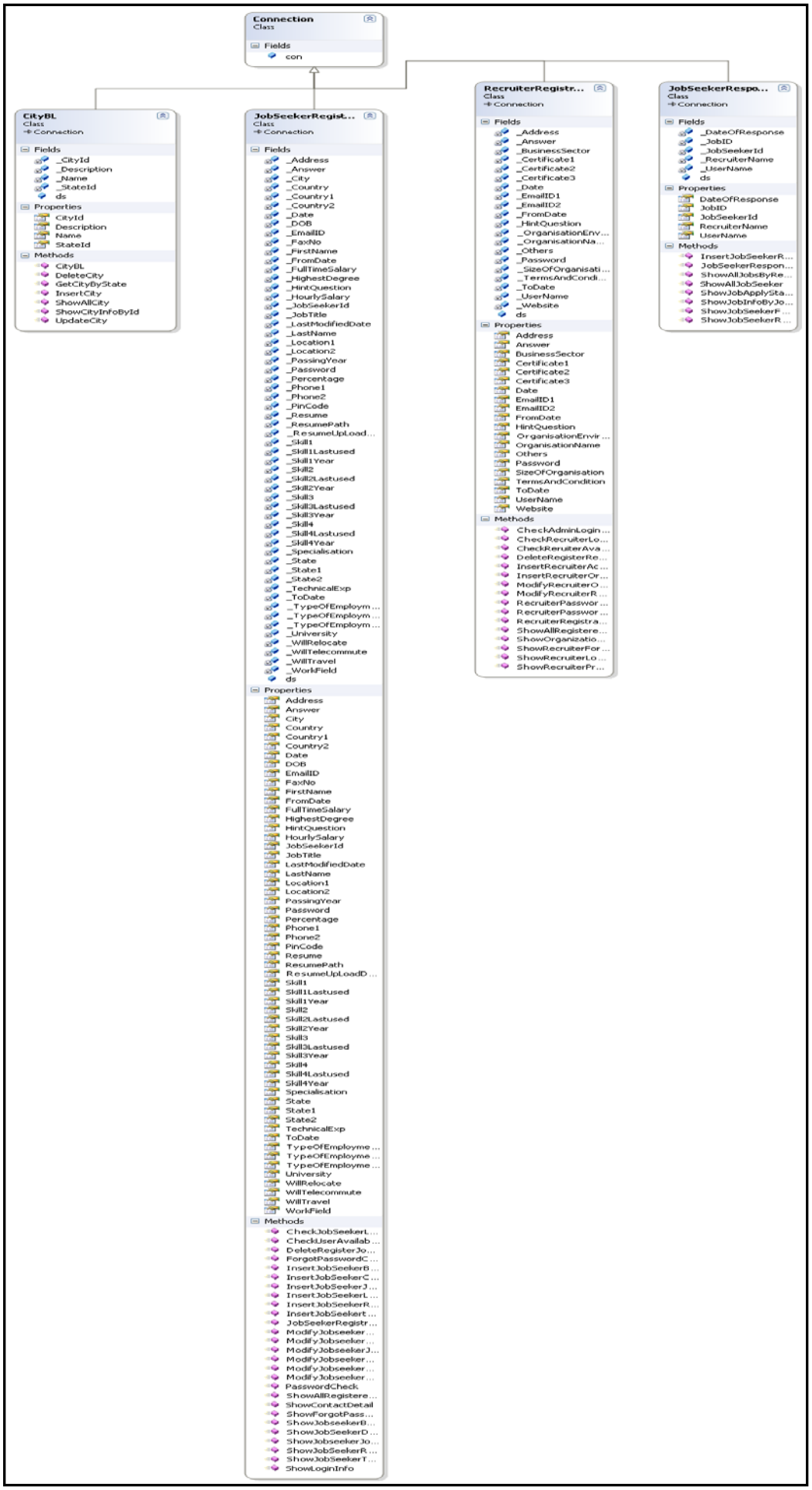
- Developing the system, which meets the SRS and solving all the requirements of the system?
- Demonstrating the system and installing the system at client's location after the acceptance testing is successful.
- Submitting the required user manual describing the system interfaces to work on it and also the documents of the system.
- Conducting any user training that might be needed for using the system.
- Maintaining the system for a period of one year after installation.

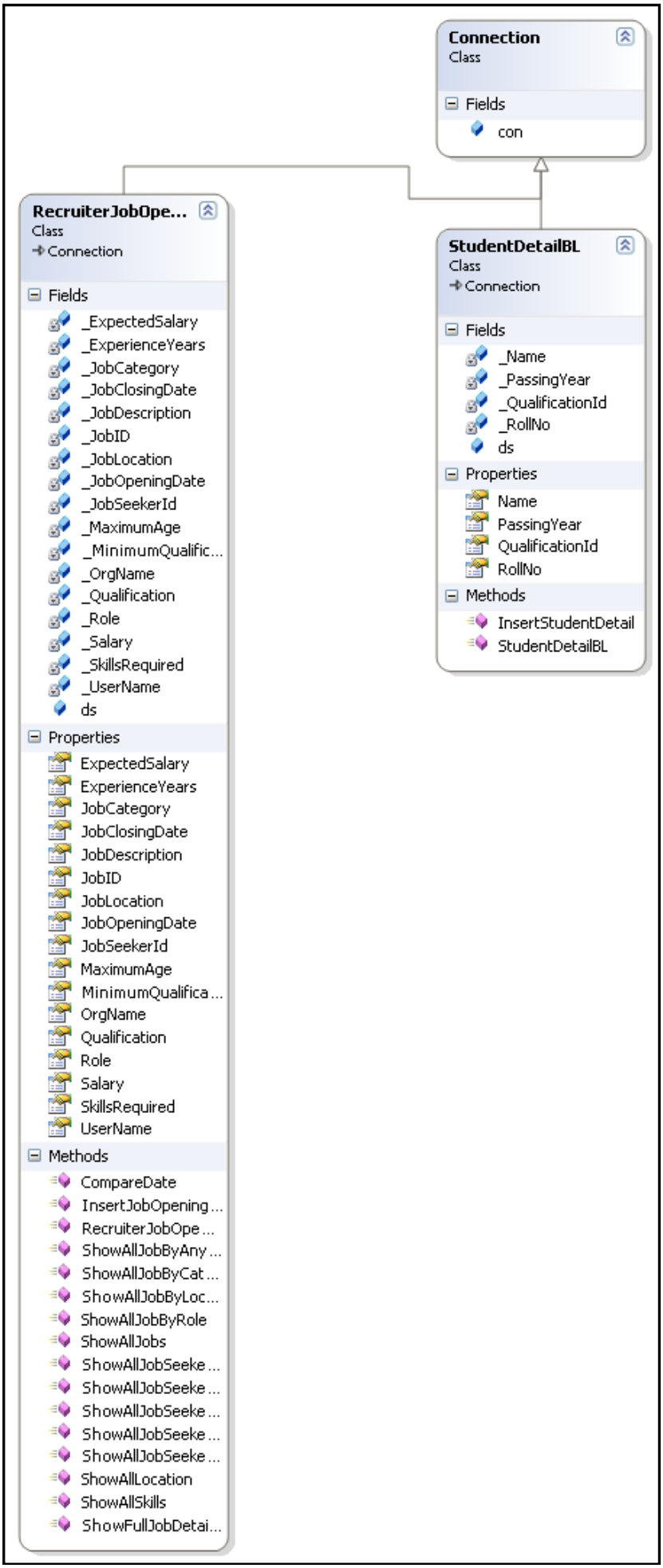
2.1 Project/Application Architecture

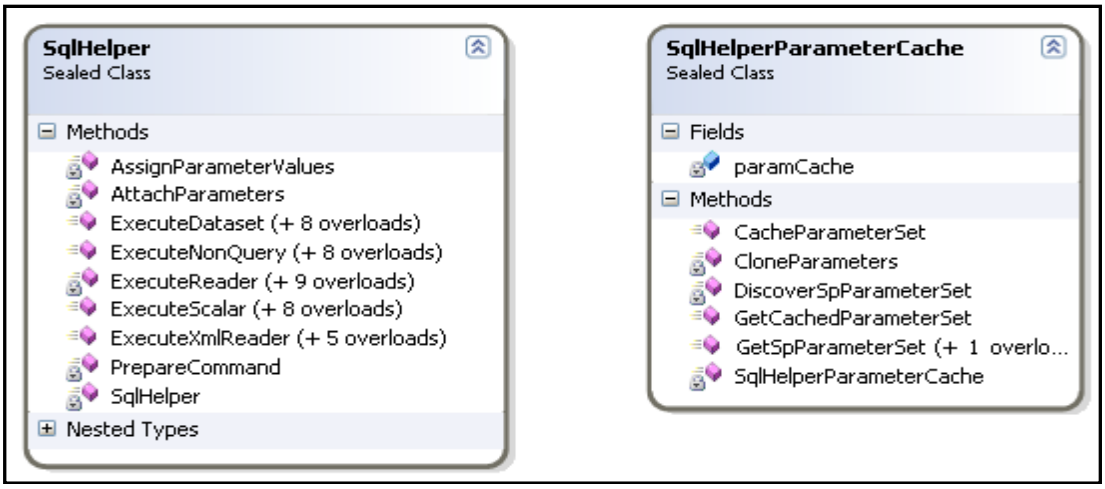
Class Diagram:

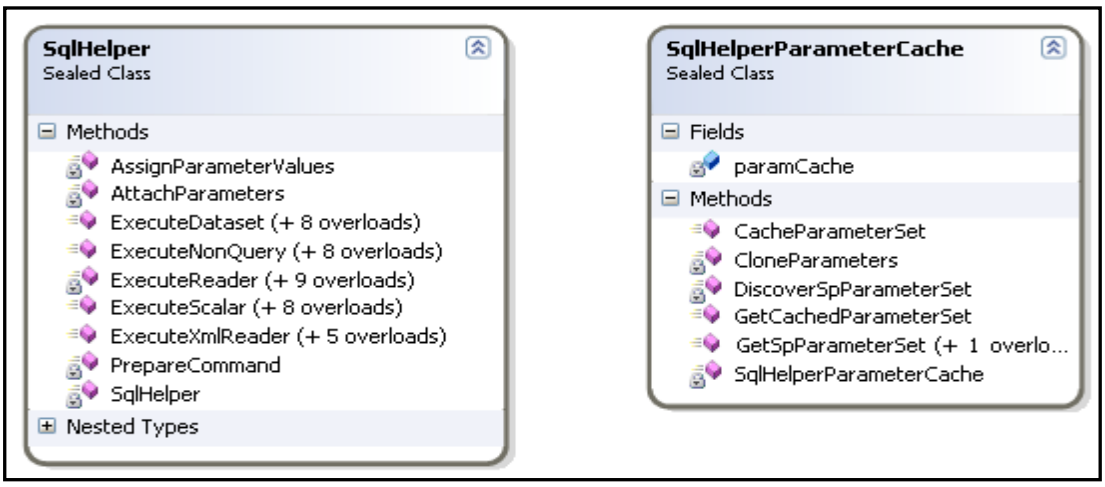



Connection
Class
Fields
con
ExperianceBL
Class
Connection
Fields
_Duration
_ExpId
_ExpType
ds
Properties
Duration
ExpId
ExpType
Methods
DeleteExperience
ExperianceBL
InsertExperiance
ShowExperience
ShowExperienceI...
UpdateExperience
QualificationBL
Class
Connection
Fields
_Description
_Qualification
_QualificationId
_QualificationLevel
ds
Properties
Description
Qualification
QualificationId
QualificationLevel
Methods
DeleteQualification
InsertQualification
QualificationBL
ShowQualification
ShowQualificatio...
UpdateQualification
StateBL
Class
Connection
Fields
_Description
_Name
_StateId
countryId
ds
Properties
CountryId
Description
Name
StateId
Methods
DeleteState
GetStateByCount...
InsertState
ShowAllState
ShowStateInfoById
StateBL
UpdateState
FunctionalAreaBL
Class
Connection
Fields
_Description
_FunctionalArea
_FunctionalId
ds
Properties
Description
FunctionalArea
FunctionalId
Methods
DeleteFunctional...
FunctionalAreaBL
InsertQualification
ShowFunctionalA...
ShowFunctionInf...
UpdateFunctional...


Connection
Class
Fields
con
CityBL
Class
Connection
Fields
_CityId
_Description
_Name
_StateId
ds
Properties
CityId
Description
Name
StateId
Methods
CityBL
DeleteCity
GetCityByState
InsertCity
ShowAllCity
ShowCityInfoById
UpdateCity
JobSeekerRegist...
Class
Connection
Fields
_Address
_Answer
_City
_Country
_Country1
_Country2
_Date
_DOB
_EmailID
_FaxNo
_FirstName
_FromDate
_FullTimeSalary
_HighestDegree
_HintQuestion
_HourlySalary
_JobSeekerId
_JobTitle
_LastModifiedDate
_LastName
_Location1
_Location2
_PassingYear
_Password
_Percentage
_Phone1
_Phone2
_PinCode
_Resume
_ResumePath
_ResumeUpLoad...
_Skill1
_Skill1Lastused
_Skill1Year
_Skill2
_Skill2Lastused
_Skill2Year
_Skill3
_Skill3Lastused
_Skill3Year
_Skill4
_Skill4Lastused
_Skill4Year
_Specialisation
_State
_State1
_State2
_TechnicalExp
_ToDate
_TypeOfEmploym...
_TypeOfEmploym...
_TypeOfEmploym...
_University
_WillRelocate
_WillTelecommute
_WillTravel
_WorkField
ds
Properties
Address
Answer
City
Country
Country1
Country2
Date
DOB
EmailID
FaxNo
FirstName
FromDate
FullTimeSalary
HighestDegree
HintQuestion
HourlySalary
JobSeekerId
JobTitle
LastModifiedDate
LastName
Location1
Location2
PassingYear
Password
Percentage
Phone1
Phone2
PinCode
Resume
ResumePath
ResumeUpLoadD...
Skill1
Skill1Lastused
Skill1Year
Skill2
Skill2Lastused
Skill2Year
Skill3
Skill3Lastused
Skill3Year
Skill4
Skill4Lastused
Skill4Year
Specialisation
State
State1
State2
TechnicalExp
ToDate
TypeOfEmployme...
TypeOfEmployme...
TypeOfEmployme...
University
WillRelocate
WillTelecommute
WillTravel
WorkField
Methods
CheckJobSeekerL...
CheckUserAvailab...
DeleteRegisterJo...
ForgotPasswordC...
InsertJobSeekerB...
InsertJobSeekerC...
InsertJobSeekerJ...
InsertJobSeekerL...
InsertJobSeekerR...
InsertJobSeekert...
JobSeekerRegistr...
ModifyJobseeker...
ModifyJobseeker...
ModifyJobseekerJ...
ModifyJobseeker...
ModifyJobseeker...
ModifyJobseeker...
PasswordCheck
ShowAllRegistere...
ShowContactDetail
ShowForgotPass...
ShowJobseekerB...
ShowJobSeekerD...
ShowJobseekerJo...
ShowJobSeekerR...
ShowJobSeekerT...
ShowLoginInfo
RecruiterRegistr...
Class
Connection
Fields
_Address
_Answer
_BusinessSector
_Certificate1
_Certificate2
_Certificate3
_Date
_EmailID1
_EmailID2
_FromDate
_HintQuestion
_OrganisationEnv...
_OrganisationNa...
_Others
_Password
_SizeOfOrganisati...
_TermsAndCondi...
_ToDate
_UserName
_Website
ds
Properties
Address
Answer
BusinessSector
Certificate1
Certificate2
Certificate3
Date
EmailID1
EmailID2
FromDate
HintQuestion
OrganisationEnvir...
OrganisationName
Others
Password
SizeOfOrganisation
TermsAndCondition
ToDate
UserName
Website
Methods
CheckAdminLogin...
CheckRecruiterLo...
CheckReruiterAva...
DeleteRegisterRe...
InsertRecruiterAc...
InsertRecruiterOr...
ModifyRecruiterO...
ModifyRecruiterR...
RecruiterPasswor...
RecruiterPasswor...
RecruiterRegistra...
ShowAllRegistere...
ShowOrganizatio...
ShowRecruiterFor...
ShowRecruiterLo...
ShowRecruiterPr...
JobSeekerRespo...
Class
Connection
Fields
_DateOfResponse
_JobID
_JobSeekerId
_RecruiterName
_UserName
ds
Properties
DateOfResponse
JobID
JobSeekerId
RecruiterName
UserName
Methods
InsertJobSeekerR...
JobSeekerRespon...
ShowAllJobsByRe...
ShowAllJobSeeker
ShowJobApplySta...
ShowJobInfoByJo...
ShowJobSeekerF...
ShowJobSeekerR...

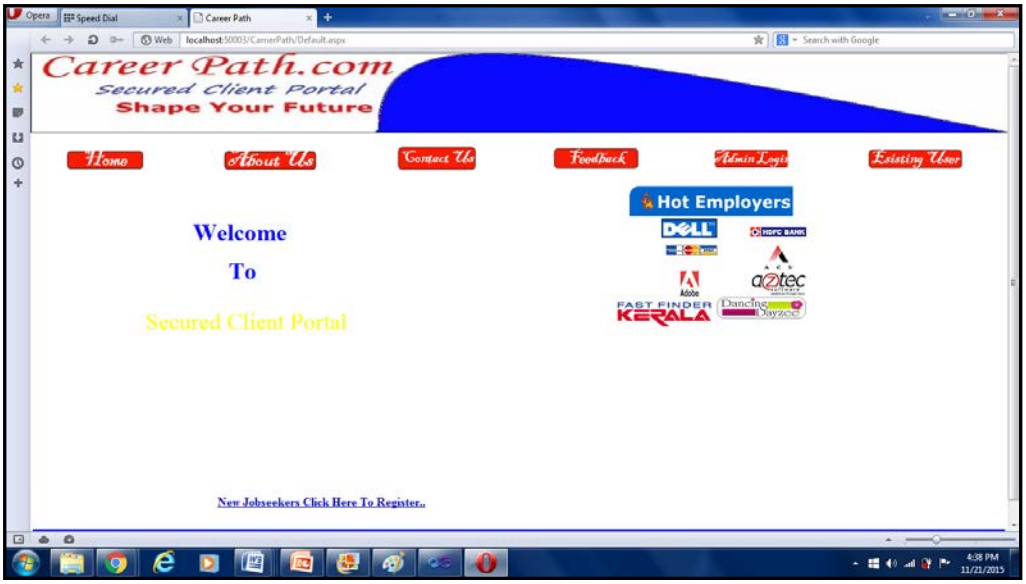

Connection
Class
Fields
con
RecruiterJobOpe...
Class
Connection
Fields
_ExpectedSalary
_ExperienceYears
_JobCategory
_JobClosingDate
_JobDescription
_JobID
_JobLocation
_JobOpeningDate
_JobSeekerId
_MaximumAge
_MinimumQualific...
_OrgName
_Qualification
_Role
_Salary
_SkillsRequired
_UserName
ds
Properties
ExpectedSalary
ExperienceYears
JobCategory
JobClosingDate
JobDescription
JobID
JobLocation
JobOpeningDate
JobSeekerId
MaximumAge
MinimumQualifica...
OrgName
Qualification
Role
Salary
SkillsRequired
UserName
Methods
CompareDate
InsertJobOpening...
RecruiterJobOpe...
ShowAllJobByAny...
ShowAllJobByCat...
ShowAllJobByLoc...
ShowAllJobByRole
ShowAllJobs
ShowAllJobSeeke...
ShowAllJobSeeke...
ShowAllJobSeeke...
ShowAllJobSeeke...
ShowAllJobSeeke...
ShowAllLocation
ShowAllSkills
ShowFullJobDetai...
StudentDetailBL
Class
Connection
Fields
_Name
_PassingYear
_QualificationId
_RollNo
ds
Properties
Name
PassingYear
QualificationId
RollNo
Methods
InsertStudentDetail
StudentDetailBL





Page name: Home page

Fields: None

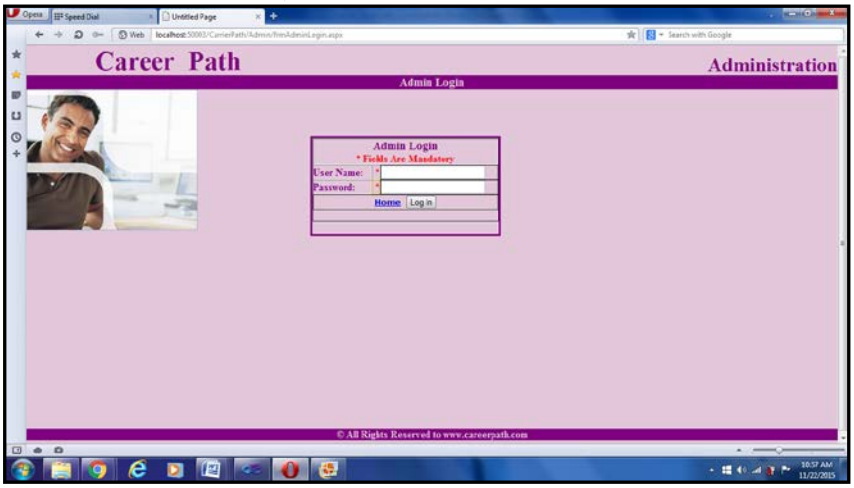


Page Name: About Us
Fields: None

Page name : Contact us
Fields : Name , Email id, Contact no , Subject , Message.

Page name: Feedback
Fields : User name , Email id , Feedback.

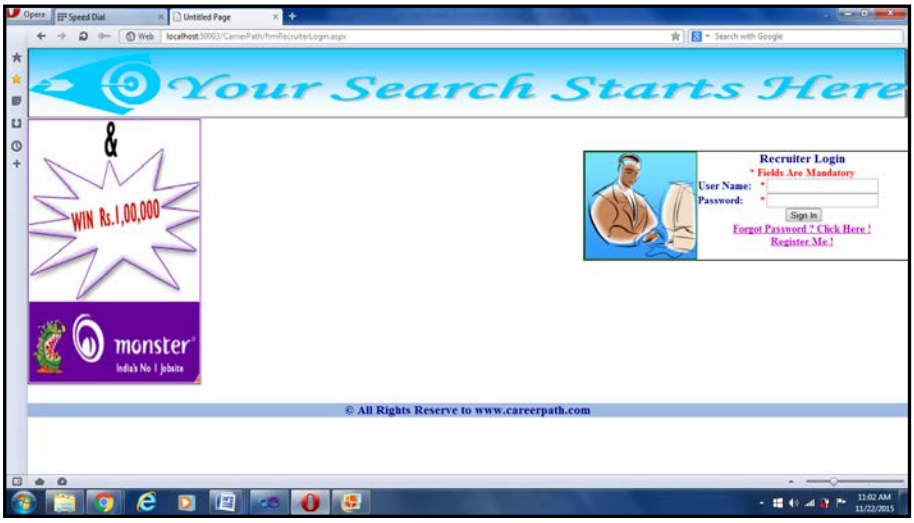
Page name : Admin login
Fields: User name , Password.

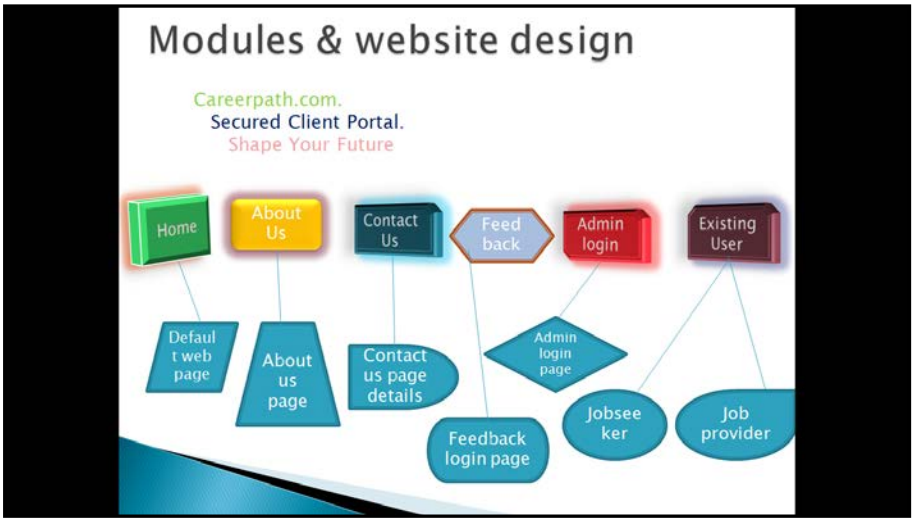


Page Name: Existing user
Fields: None

Page Name: Jobseeker login
Fields: Username, Password.

Page Name: Job recruiter login
Fields: Username , Password.





2.2 Project/Application Information flows

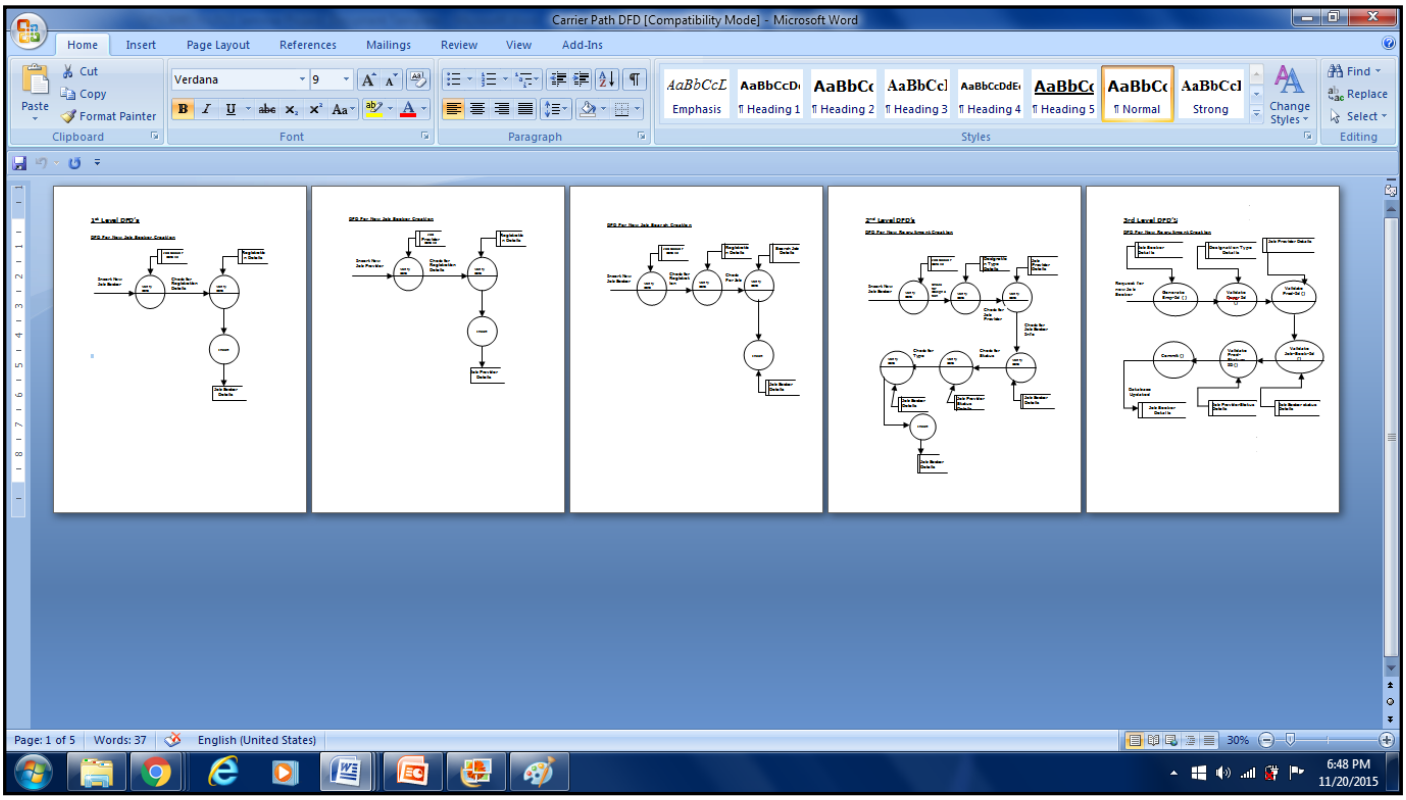
Data Flow

- A Data Flow has only one direction of flow between symbols. It may flow in both directions between a process and a data store to show a read before an update. The later is usually indicated however by two separate arrows since these happen at different type.
- A join in DFD means that exactly the same data comes from any of two or more different processes data store or sink to a common location.
- A data flow cannot go directly back to the same process it leads. There must be at least one other process that handles the data flow produce some other data flow returns the original data into the beginning process.
- A Data flow to a data store means update (delete or change).
- A data Flow from a data store means retrieve or use.

A data flow has a noun phrase label more than one data flow noun phrase can appear on a single arrow as long as all of the flows on the same arrow move together as one package.

Our Project Dataflow Diagram: Dataflow Diagram(DFD) is referred in the below screenshot.

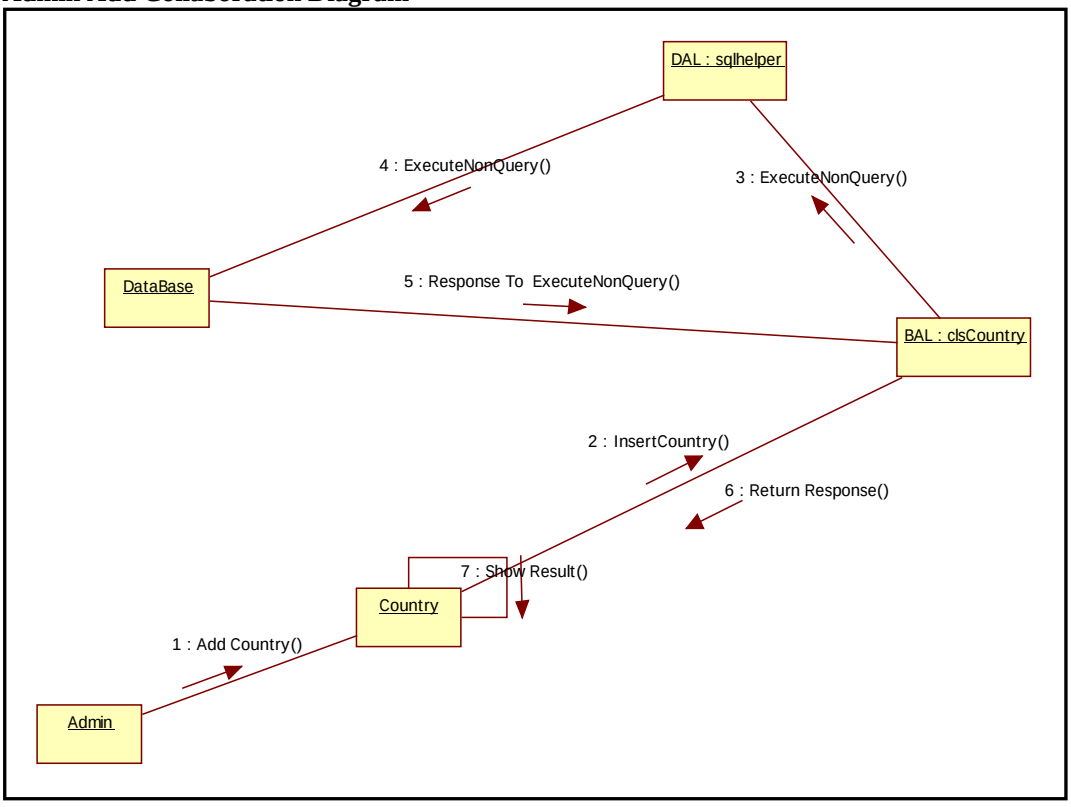
Screen Shot of DFD:



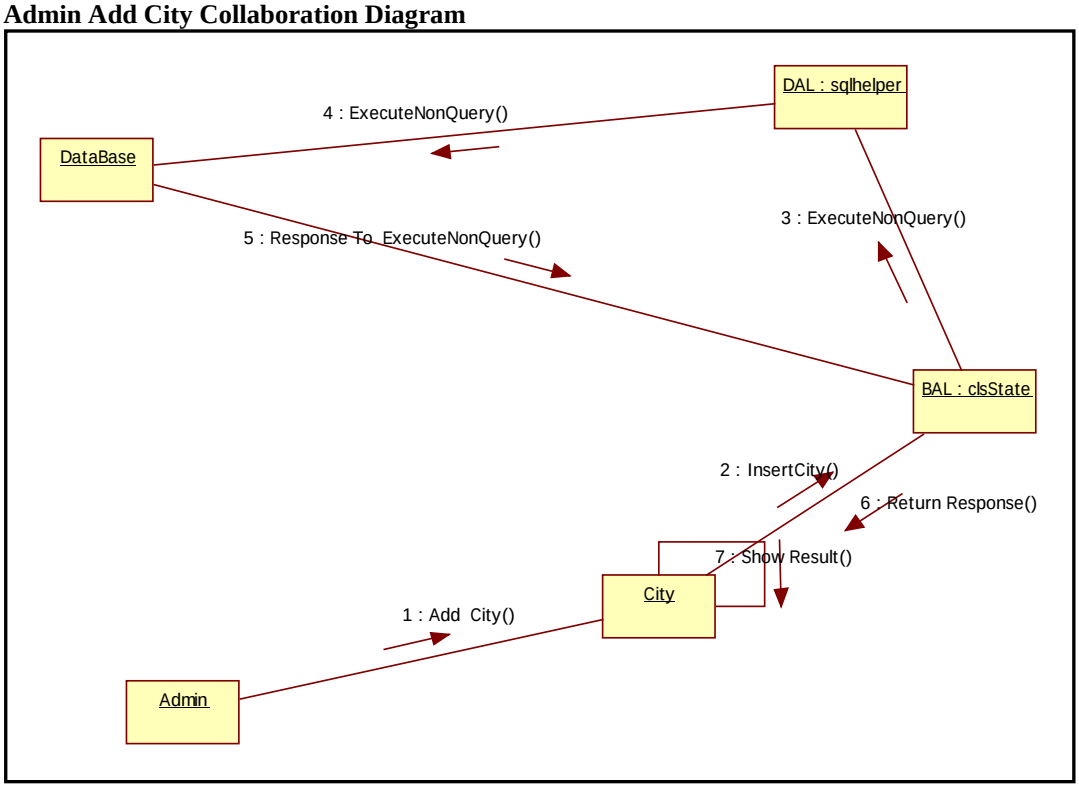
Collaboration Diagrams

Admin Login

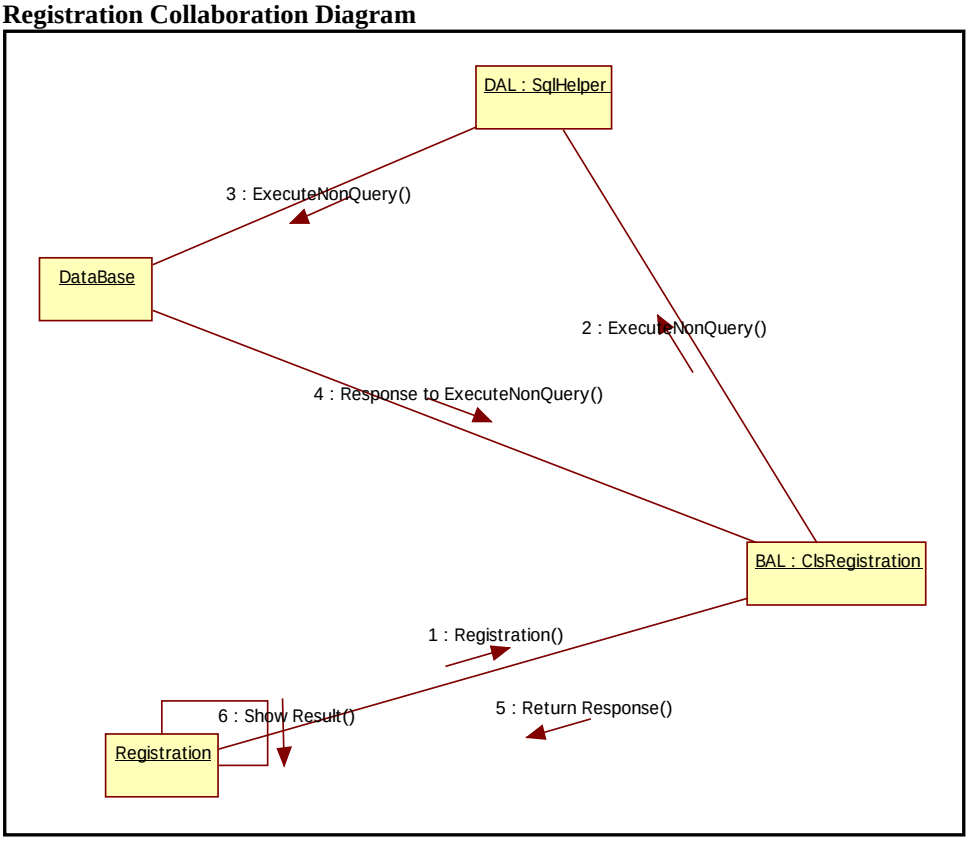
Admin Add Collaboration Diagram



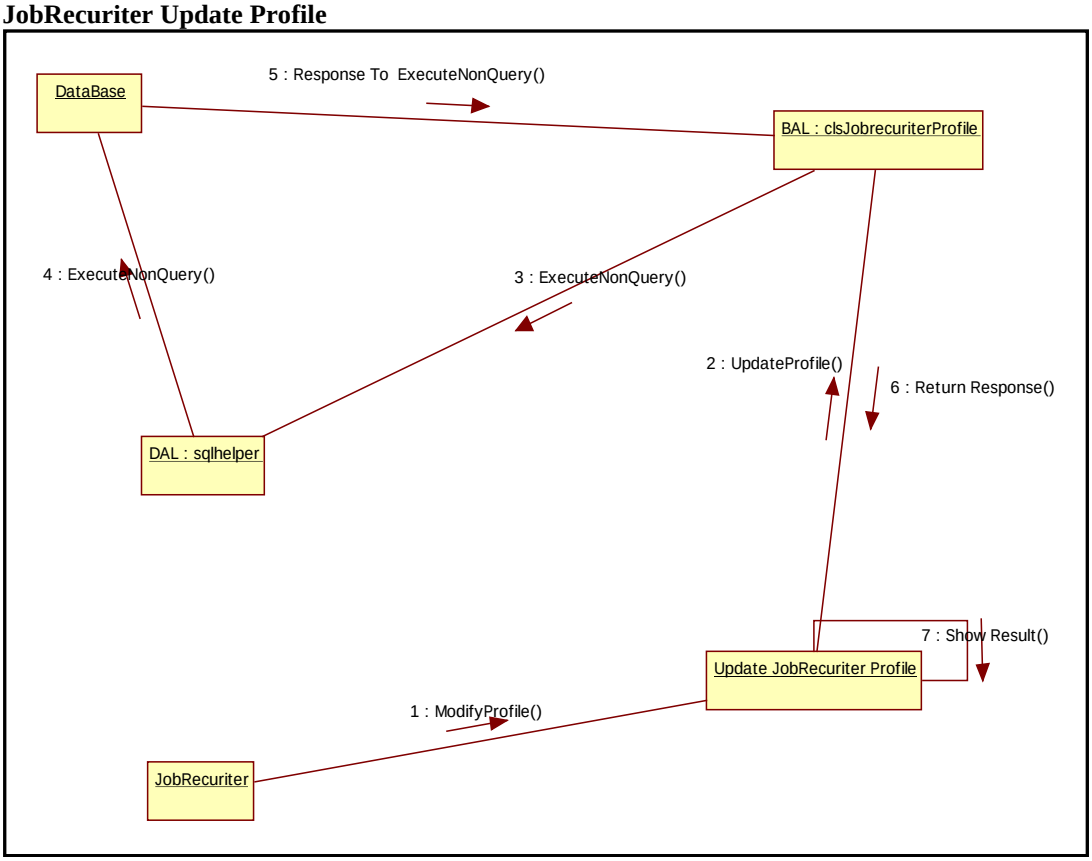
Admin Add City Collaboration Diagram



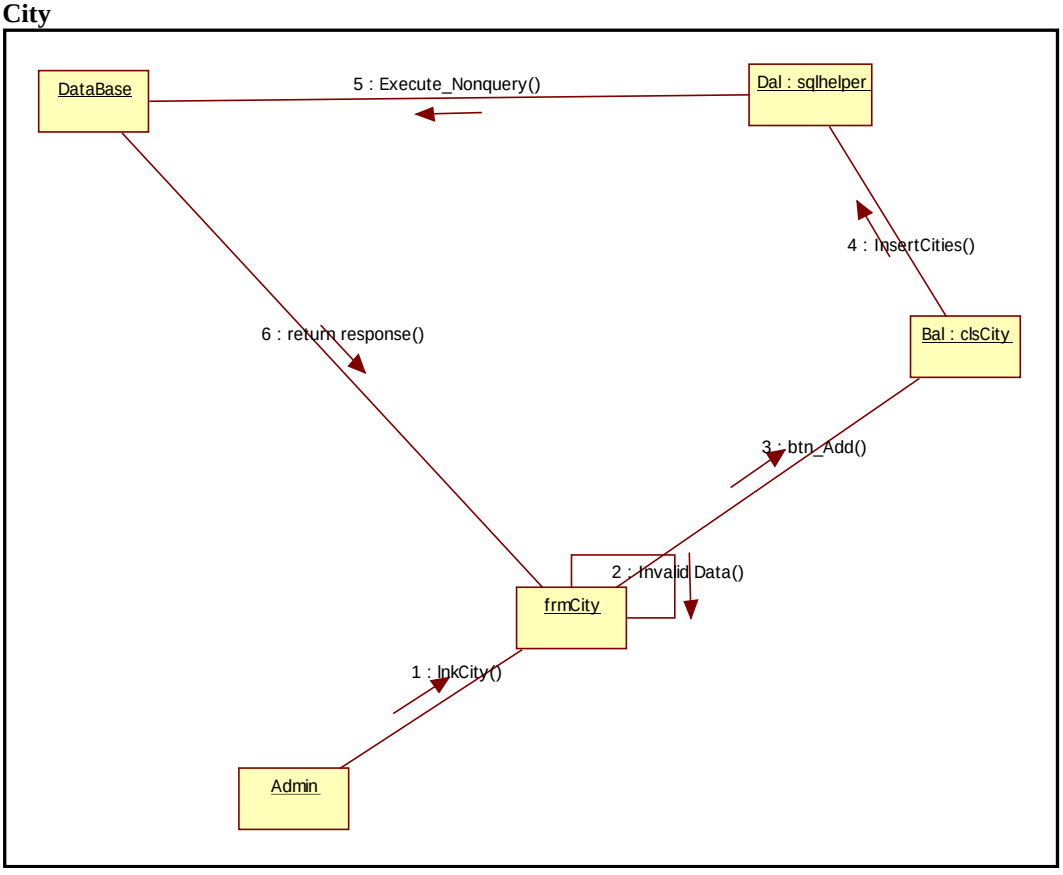
Registration Collaboration Diagram



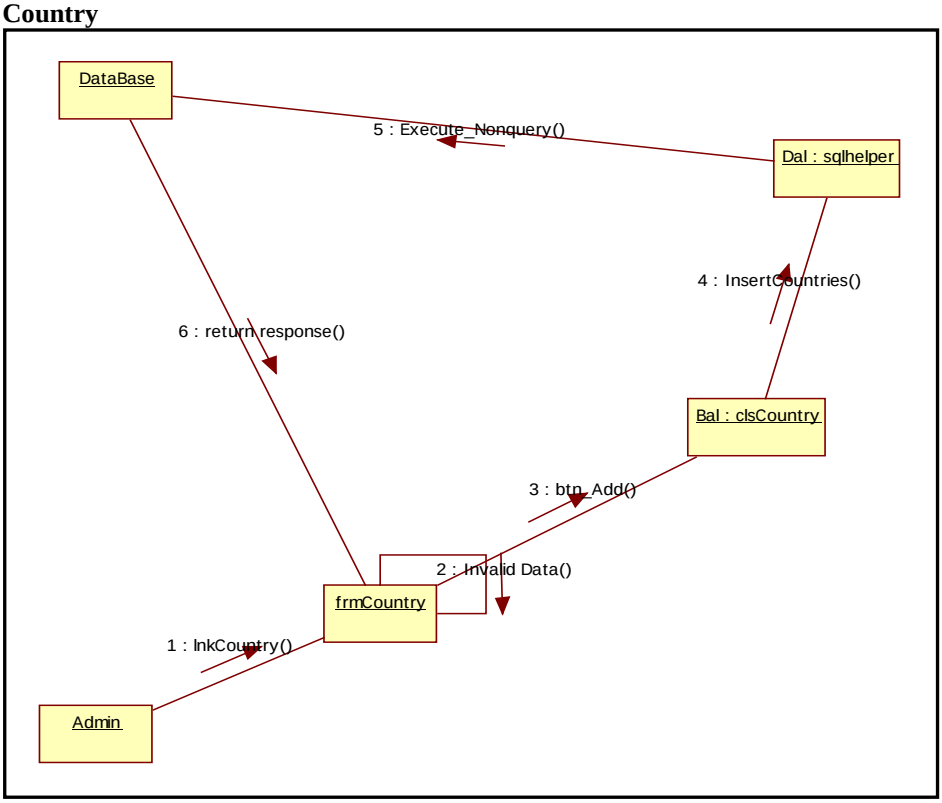
JobRecuriter Update Profile



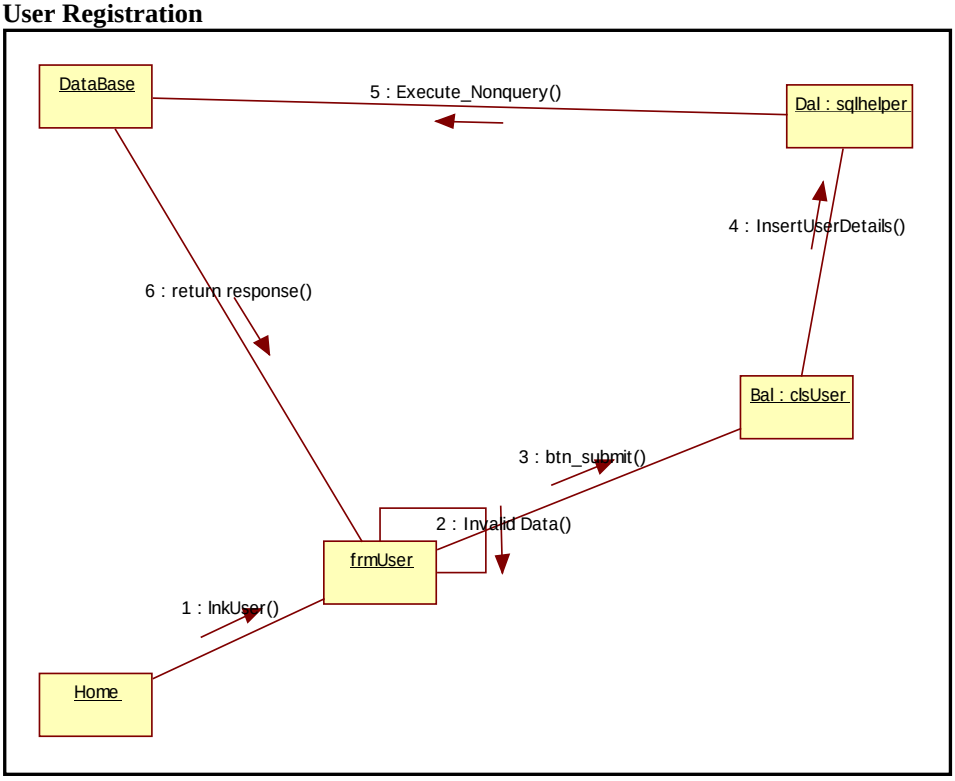
City



Country



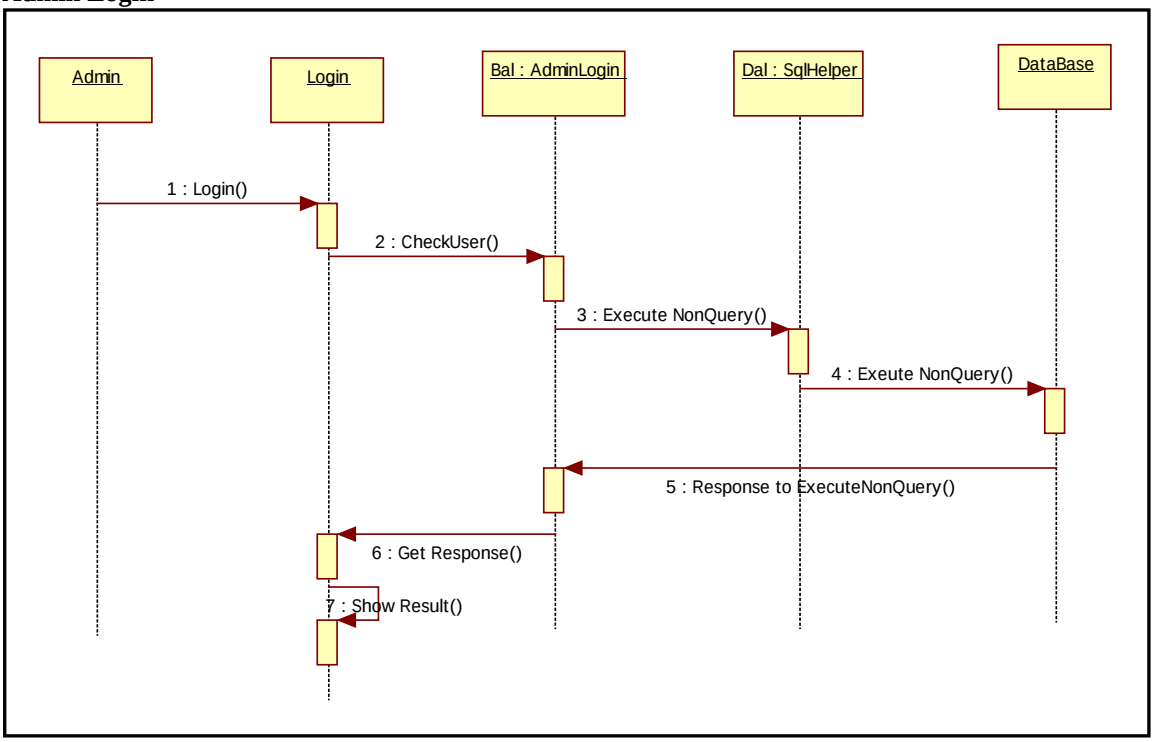
User Registration



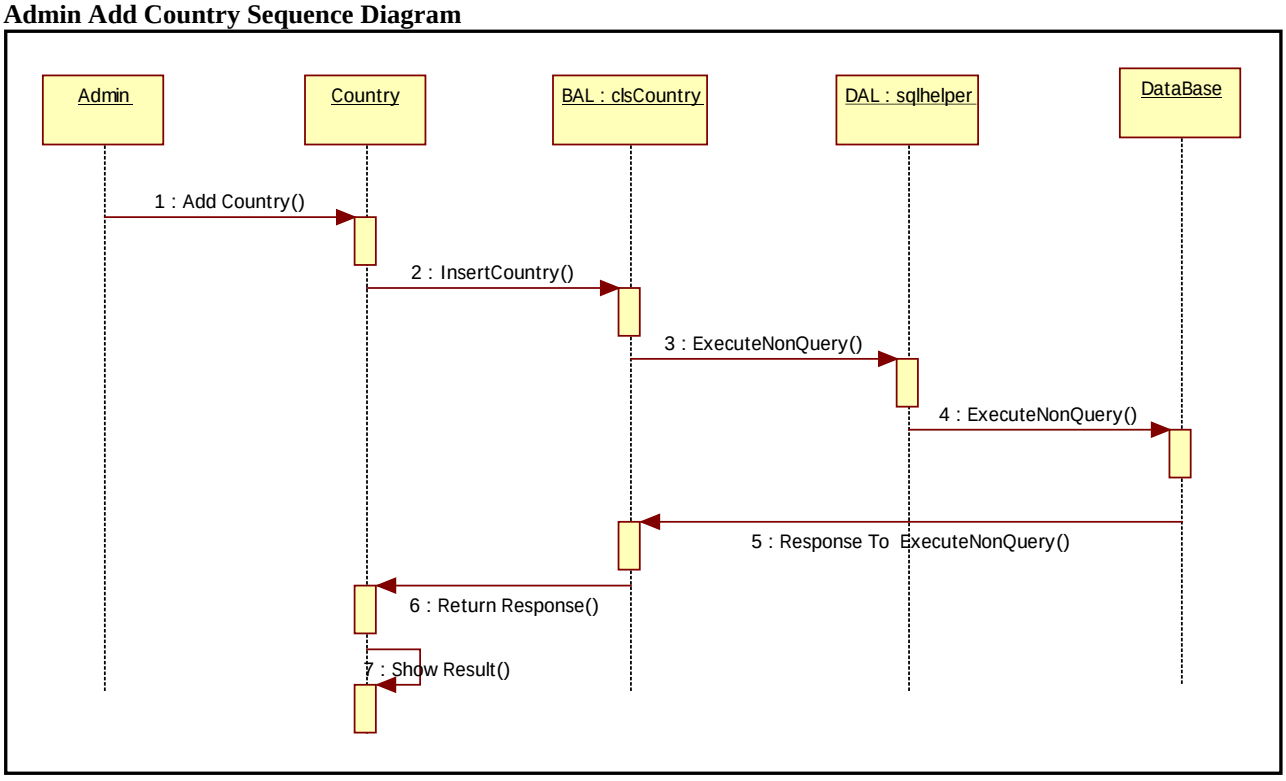
Sequence Diagrams:

Sequence Diagrams Represent the objects participating the interaction horizontally and time vertically.

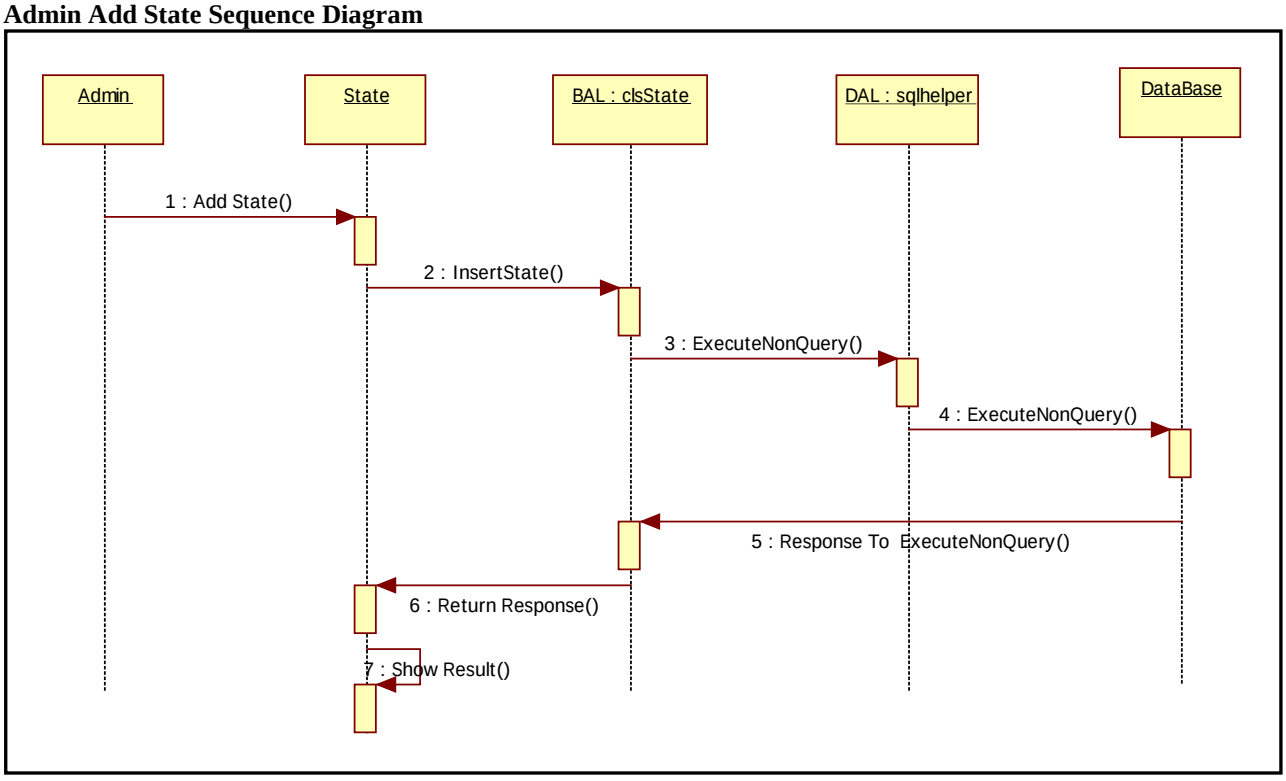
Admin Login



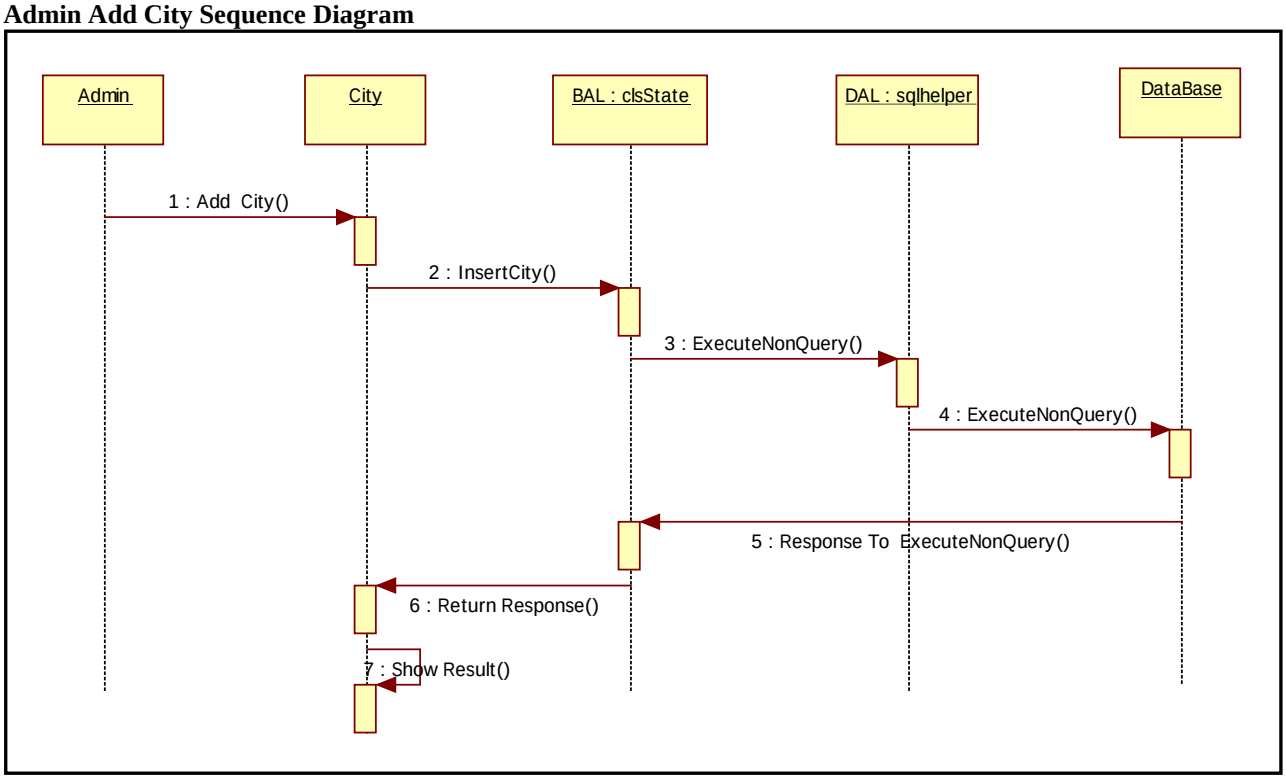
Admin Add Country Sequence Diagram



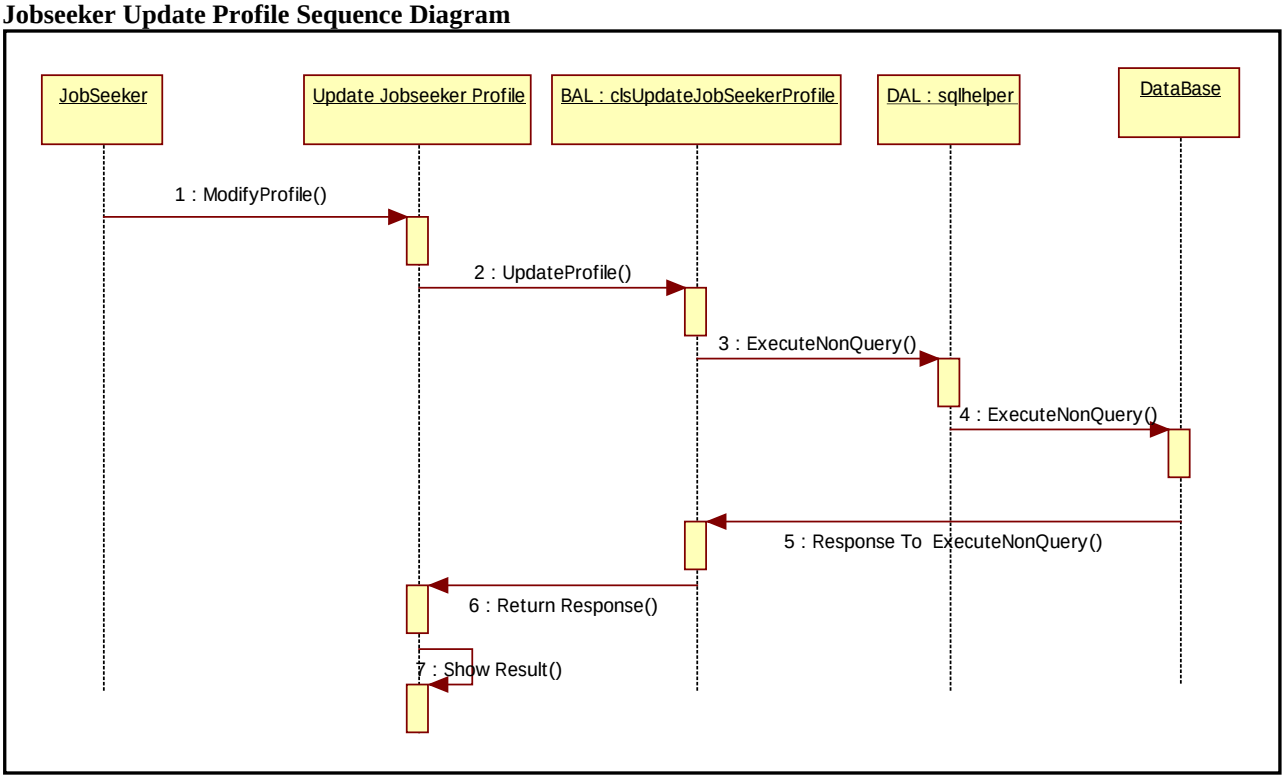
Admin Add State Sequence Diagram



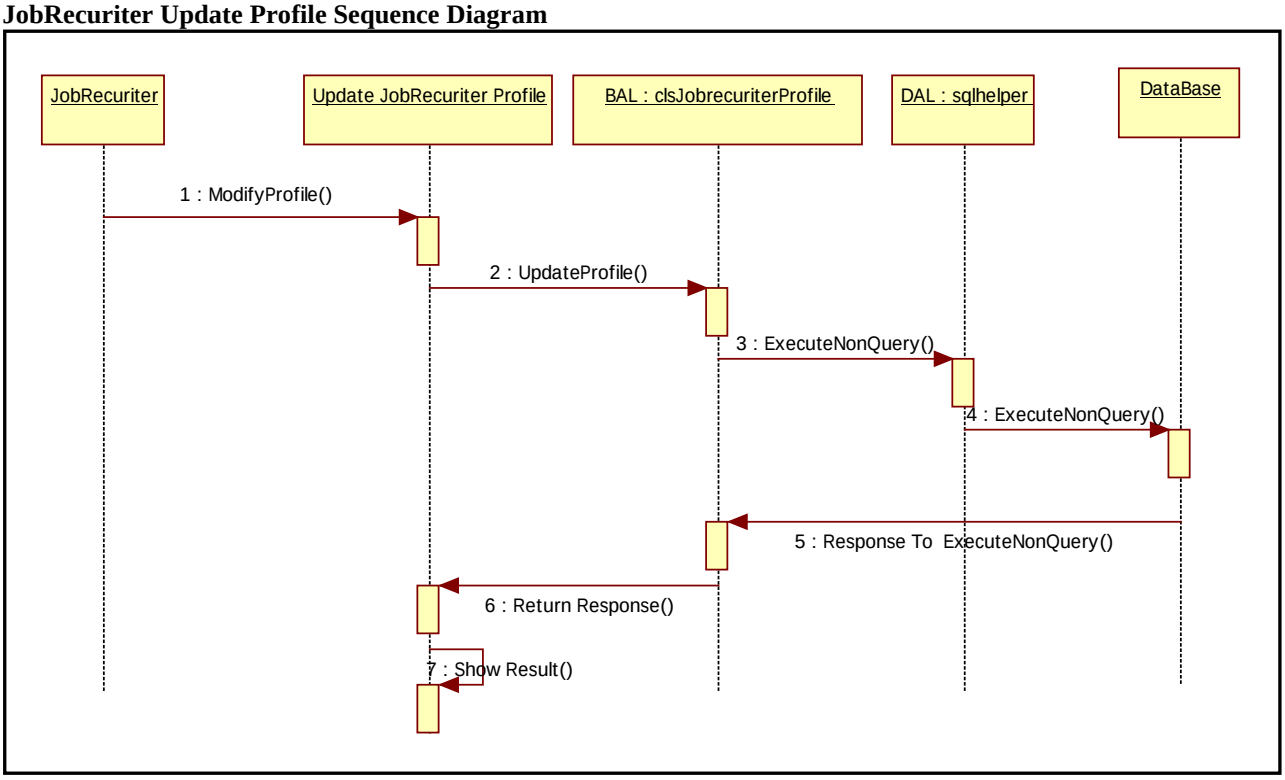
Admin Add City Sequence Diagram



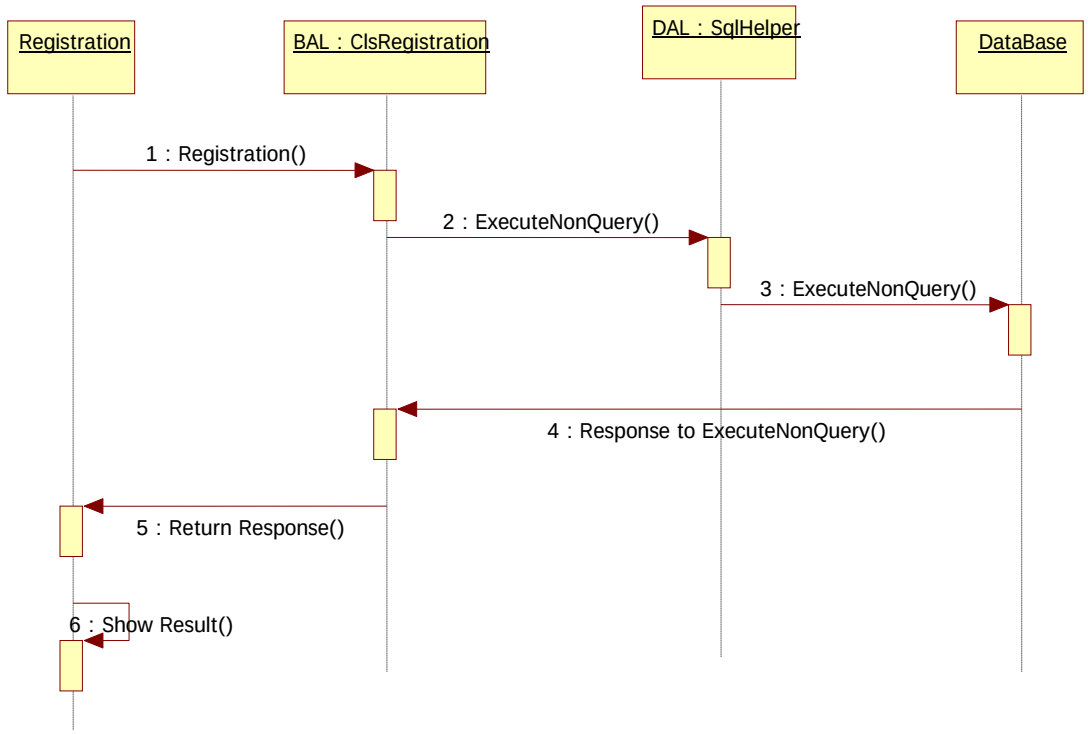
Jobseeker Update Profile Sequence Diagram



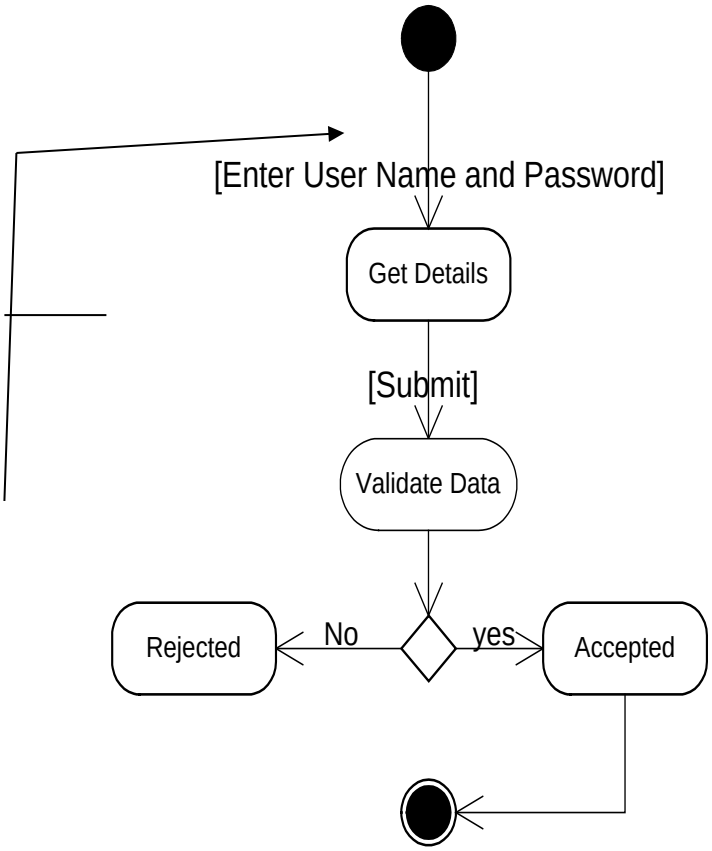
JobRecuriter Update Profile Sequence Diagram



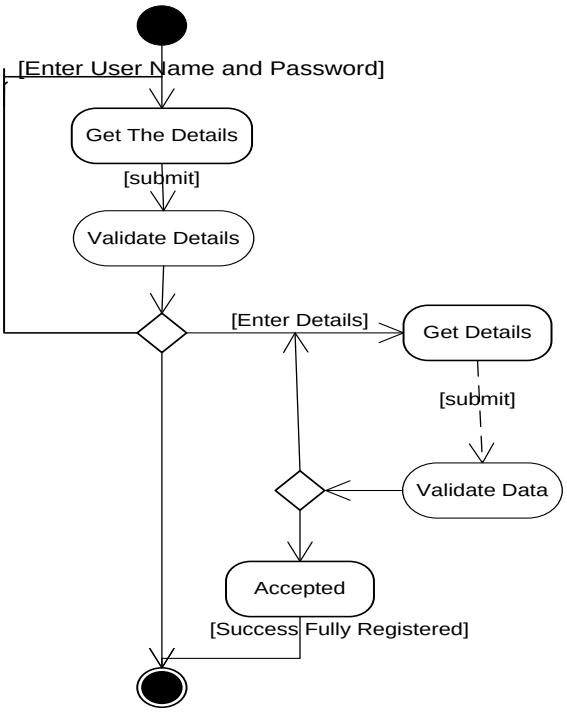
Registration Sequence Diagram



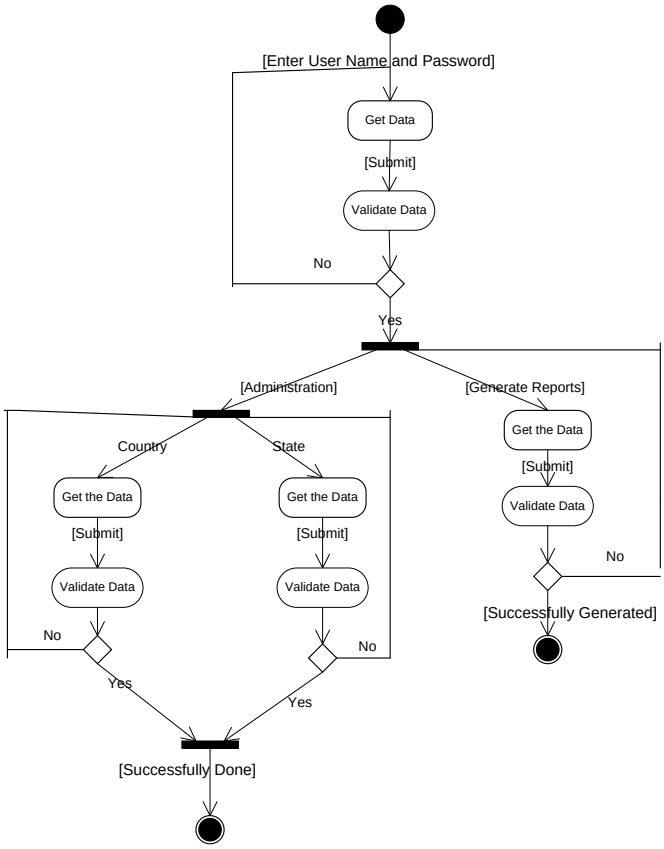
Activity Diagrams:
Login Activity



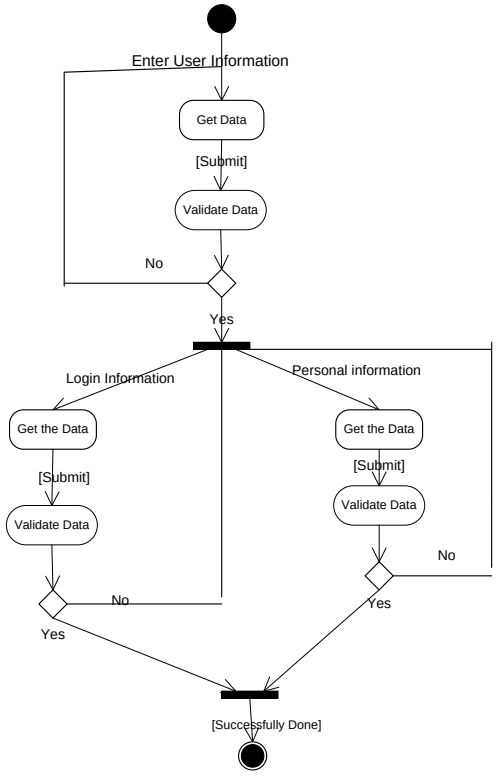
Jobseeker Registration Activity



Admin Activity Diagram:



Job Recruiter Activity Diagram:



2.3 Interactions with other Projects (if Any)

This project does not have any interaction with other Projects.

2.4 Interactions with other Applications

This project does not have any interactions with other applications.

2.5 Capabilities

Hardware and Software Specifications.

S/w

- Microsoft .NET framework 4.0
- Language: C#.Net, Asp.net
- Microsoft Visual Studio 2008IDE or higher
- OS: Microsoft Windows 7 or higher

H/w

- P IV Processor or higher
- 512 MB RAM minimum
- Secondary memory of 20 – 50 GB minimum

Introduction to .NET Framework

The .NET Framework is a new computing platform that simplifies application development in the highly distributed environment of the Internet. The .NET Framework is designed to fulfill the following objectives:

- To provide a consistent object-oriented programming environment whether object code is stored and executed locally, executed locally but Internet-distributed, or executed remotely.
- To provide a code-execution environment that minimizes software deployment and versioning conflicts.
- To provide a code-execution environment that guarantees safe execution of code, including code created by an unknown or semi-trusted third party.
- To provide a code-execution environment that eliminates the performance problems of scripted or interpreted environments.
- To make the developer experience consistent across widely varying types of applications, such as Windows-based applications and Web-based applications.
- To build all communication on industry standards to ensure that code based on the .NET Framework can integrate with any other code.

The .NET Framework has two main components: the common language runtime and the .NET Framework class library. The common language runtime is the foundation of the .NET Framework. You can think of the runtime as an agent that manages code at execution time, providing core services such as memory management, thread management, and Remoting, while also enforcing strict type safety and other forms of code accuracy that ensure security and robustness. In fact, the concept of code management is a fundamental principle of the runtime. Code that targets the runtime is known as managed code, while code that does not target the runtime is known as unmanaged code. The class library, the other main component of the .NET Framework, is a comprehensive, object-oriented collection of reusable types that you can use to develop applications ranging from traditional command-line or graphical user interface (GUI) applications to applications based on the latest innovations provided by ASP.NET, such as Web Forms and XML Web services.

The .NET Framework can be hosted by unmanaged components that load the common language runtime into their processes and initiate the execution of managed code, thereby creating a software environment that can exploit both managed and unmanaged features. The .NET Framework not only provides several runtime hosts, but also supports the development of third-party runtime hosts.

For example, ASP.NET hosts the runtime to provide a scalable, server-side environment for managed code. ASP.NET works directly with the runtime to enable Web Forms applications and XML Web services, both of which are discussed later in this topic.

Internet Explorer is an example of an unmanaged application that hosts the runtime (in the form of a MIME type extension). Using Internet Explorer to host the runtime enables you to embed managed components or Windows Forms controls in HTML documents. Hosting the runtime in this way makes managed mobile code (similar to Microsoft® ActiveX® controls) possible, but with significant improvements that only managed code can offer, such as semi-trusted execution and secure isolated file storage.

The following illustration shows the relationship of the common language runtime and the class library to your applications and to the overall system. The illustration also shows how managed code operates within a larger architecture.

Features of the common language runtime

The common language runtime manages memory, thread execution, code execution, code safety verification, compilation, and other system services. These features are intrinsic to the managed code that runs on the common language runtime.

With regards to security, managed components are awarded varying degrees of trust, depending on a number of factors that include their origin (such as the Internet, enterprise network, or local computer). This means that a managed

component might or might not be able to perform file-access operations, registry-access operations, or other sensitive functions, even if it is being used in the same active application.

The runtime enforces code access security. For example, users can trust that an executable embedded in a Web page can play an animation on screen or sing a song, but cannot access their personal data, file system, or network. The security features of the runtime thus enable legitimate Internet-deployed software to be exceptionally featuring rich.

The runtime also enforces code robustness by implementing a strict type- and code-verification infrastructure called the common type system (CTS). The CTS ensures that all managed code is self-describing. The various Microsoft and third-party language compilers

Generate managed code that conforms to the CTS. This means that managed code can consume other managed types and instances, while strictly enforcing type fidelity and type safety.

In addition, the managed environment of the runtime eliminates many common software issues. For example, the runtime automatically handles object layout and manages references to objects, releasing them when they are no longer being used. This automatic memory management resolves the two most common application errors, memory leaks and invalid memory references.

The runtime also accelerates developer productivity. For example, programmers can write applications in their development language of choice, yet take full advantage of the runtime, the class library, and components written in other languages by other developers. Any compiler vendor who chooses to target the runtime can do so. Language compilers that target the .NET Framework make the features of the .NET Framework available to existing code written in that language, greatly easing the migration process for existing applications.

While the runtime is designed for the software of the future, it also supports software of today and yesterday. Interoperability between managed and unmanaged code enables developers to continue to use necessary COM components and DLLs.

The runtime is designed to enhance performance. Although the common language runtime provides many standard runtime services, managed code is never interpreted. A feature called just-in-time (JIT) compiling enables all managed code to run in the native machine language of the system on which it is executing. Meanwhile, the memory manager removes the possibilities of fragmented memory and increases memory locality-of-reference to further increase performance.

Finally, the runtime can be hosted by high-performance, server-side applications, such as Microsoft® SQL Server™ and Internet Information Services (IIS). This infrastructure enables you to use managed code to write your business logic, while still enjoying the superior performance of the industry's best enterprise servers that support runtime hosting.

.NET Framework class library

The .NET Framework class library is a collection of reusable types that tightly integrate with the common language runtime. The class library is object oriented, providing types from which your own managed code can derive functionality. This not only makes the .NET Framework types easy to use, but also reduces the time associated with learning new features of the .NET Framework. In addition, third-party components can integrate seamlessly with classes in the .NET Framework.

For example, the .NET Framework collection classes implement a set of interfaces that you can use to develop your own collection classes. Your collection classes will blend seamlessly with the classes in the .NET Framework.

As you would expect from an object-oriented class library, the .NET Framework types enable you to accomplish a range of common programming tasks, including tasks such as string management, data collection, database connectivity, and file access. In addition to these common tasks, the class library includes types that support a variety of specialized development scenarios. For example, you can use the .NET Framework to develop the following types of applications and services:

- Console applications.
- Scripted or hosted applications.
- Windows GUI applications (Windows Forms).
- ASP.NET applications.
- XML Web services.
- Windows services.

For example, the Windows Forms classes are a comprehensive set of reusable types that vastly simplify Windows GUI development. If you write an ASP.NET Web Form application, you can use the Web Forms classes.

Client application development

Client applications are the closest to a traditional style of application in Windows-based programming. These are the types of applications that display windows or forms on the desktop, enabling a user to perform a task. Client applications include applications such as word processors and spreadsheets, as well as custom business applications such as data-entry tools, reporting tools, and so on. Client applications usually employ windows, menus, buttons, and other GUI elements, and they likely access local resources such as the file system and peripherals such as printers.

Another kind of client application is the traditional ActiveX control (now replaced by the managed Windows Forms control) deployed over the Internet as a Web page. This application is much like other client applications: it is executed natively, has access to local resources, and includes graphical elements.

In the past, developers created such applications using C/C++ in conjunction with the Microsoft Foundation Classes (MFC) or with a rapid application development (RAD) environment such as Microsoft® Visual Basic®. The .NET Framework incorporates aspects of these existing products into a single, consistent development environment that drastically simplifies the development of client applications.

The Windows Forms classes contained in the .NET Framework are designed to be used for GUI development. You can easily create command windows, buttons, menus, toolbars, and other screen elements with the flexibility necessary to accommodate shifting business needs.

For example, the .NET Framework provides simple properties to adjust visual attributes associated with forms. In some cases the underlying operating system does not support changing these attributes directly, and in these cases the .NET Framework automatically recreates the forms. This is one of many ways in which the .NET Framework integrates the developer interface, making coding simpler and more consistent.

Unlike ActiveX controls, Windows Forms controls have semi-trusted access to a user's computer. This means that binary or natively executing code can access some of the resources on the user's system (such as GUI elements and limited file access) without being able to access or compromise other resources. Because of code access security, many applications that once needed to be installed on a user's system can now be safely deployed through the Web. Your applications can implement the features of a local application while being deployed like a Web page.

ASP.NET

Server Application Development

Server-side applications in the managed world are implemented through runtime hosts. Unmanaged applications host the common language runtime, which allows your custom managed code to control the behavior of the server. This model provides you with all the features of the common language runtime and class library while gaining the performance and scalability of the host server.

The following illustration shows a basic network schema with managed code running in different server environments. Servers such as IIS and SQL Server can perform standard operations while your application logic executes through the managed code.

Server-side managed code

ASP.NET is the hosting environment that enables developers to use the .NET Framework to target Web-based applications. However, ASP.NET is more than just a runtime host; it is a complete architecture for developing Web sites and Internet-distributed objects using managed code. Both Web Forms and XML Web services use IIS and ASP.NET as the publishing mechanism for applications, and both have a collection of supporting classes in the .NET Framework.

XML Web services, an important evolution in Web-based technology, are distributed, server-side application components similar to common Web sites. However, unlike Web-based applications, XML Web services components have no UI and are not targeted for browsers such as Internet Explorer and Netscape Navigator. Instead, XML Web services consist of reusable software components designed to be consumed by other applications, such as traditional client applications, Web-based applications, or even other XML Web services. As a result, XML Web services technology is rapidly moving application development and deployment into the highly distributed environment of the Internet.

If you have used earlier versions of ASP technology, you will immediately notice the improvements that ASP.NET and Web Forms offers. For example, you can develop Web Forms pages in any language that supports the .NET Framework. In addition, your code no longer needs to share the same file with your HTTP text (although it can continue to do so if you prefer). Web Forms pages execute in native machine language because, like any other managed application, they take full advantage of the runtime. In contrast, unmanaged ASP pages are always scripted and interpreted. ASP.NET pages are faster, more functional, and easier to develop than unmanaged ASP pages because they interact with the runtime like any managed application.

The .NET Framework also provides a collection of classes and tools to aid in development and consumption of XML Web services applications. XML Web services are built on standards such as SOAP (a remote procedure-call protocol), XML (an extensible data format), and WSDL (the Web Services Description Language). The .NET Framework is built on these standards to promote interoperability with non-Microsoft solutions.

For example, the Web Services Description Language tool included with the .NET Framework SDK can query an XML Web service published on the Web, parse its WSDL description, and produce C# or Visual Basic source code that your application can use to become a client of the XML Web service. The source code can create classes derived from classes in the class library that handle all the underlying communication using SOAP and XML parsing. Although you can use the class library to consume XML Web services directly, the Web Services Description Language tool and the other tools contained in the SDK facilitate your development efforts with the .NET Framework.

If you develop and publish your own XML Web service, the .NET Framework provides a set of classes that conform to all the underlying communication standards, such as SOAP, WSDL, and XML. Using those classes enables you to focus on the logic of your service, without concerning yourself with the communications infrastructure required by distributed software development.

Finally, like Web Forms pages in the managed environment, your XML Web service will run with the speed of native machine language using the scalable communication of IIS.

Active Server Pages.NET

ASP.NET is a programming framework built on the common language runtime that can be used on a server to build powerful Web applications. ASP.NET offers several important advantages over previous Web development models:

- **Enhanced Performance.** ASP.NET is compiled common language runtime code running on the server. Unlike its interpreted predecessors, ASP.NET can take advantage of early binding, just-in-time compilation, native optimization, and caching services right out of the box. This amounts to dramatically better performance before you ever write a line of code.
- **World-Class Tool Support.** The ASP.NET framework is complemented by a rich toolbox and designer in the Visual Studio integrated development environment. WYSIWYG editing, drag-and-drop server controls, and automatic deployment are just a few of the features this powerful tool provides.
- **Power and Flexibility.** Because ASP.NET is based on the common language runtime, the power and flexibility of that entire platform is available to Web application developers. The .NET Framework class library, Messaging, and Data Access solutions are all seamlessly accessible from the Web. ASP.NET is also language-independent, so you can choose the language that best applies to your application or partition your application across many languages. Further, common language runtime interoperability guarantees that your existing investment in COM-based development is preserved when migrating to ASP.NET.
- **Simplicity.** ASP.NET makes it easy to perform common tasks, from simple form submission and client authentication to deployment and site configuration. For example, the ASP.NET page framework allows you to build user interfaces that cleanly separate application logic from presentation code and to handle events in a simple, Visual Basic - like forms processing model. Additionally, the common language runtime simplifies development, with managed code services such as automatic reference counting and garbage collection.
- **Manageability.** ASP.NET employs a text-based, hierarchical configuration system, which simplifies applying settings to your server environment and Web applications. Because configuration information is stored as plain text, new settings may be applied without the aid of local administration tools. This "zero local administration" philosophy extends to deploying ASP.NET Framework applications as well. An ASP.NET Framework application is deployed to a server simply by copying the necessary files to the server. No server restart is required, even to deploy or replace running compiled code.
- **Scalability and Availability.** ASP.NET has been designed with scalability in mind, with features specifically tailored to improve performance in clustered and multiprocessor environments. Further, processes are closely monitored and managed by the ASP.NET runtime, so that if one misbehaves (leaks, deadlocks), a new process can be created in its place, which helps keep your application constantly available to handle requests.
- **Customizability and Extensibility.** ASP.NET delivers a well-factored architecture that allows developers to "plug-in" their code at the appropriate level. In fact, it is possible to extend or replace any subcomponent of the ASP.NET runtime with your own custom-written component. Implementing custom authentication or state services has never been easier.
- **Security.** With built in Windows authentication and per-application configuration, you can be assured that your applications are secure.

Languages Support

The Microsoft .NET Platform currently offers built-in support for three languages: C#, Visual Basic, and JScript.

What is ASP.NET web forms?

The ASP.NET Web Forms page framework is a scalable common language runtime programming model that can be used on the server to dynamically generate Web pages.

Intended as a logical evolution of ASP (ASP.NET provides syntax compatibility with existing pages), the ASP.NET Web Forms framework has been specifically designed to address a number of key deficiencies in the previous model. In particular, it provides:

- The ability to create and use reusable UI controls that can encapsulate common functionality and thus reduce the amount of code that a page developer has to write.
- The ability for developers to cleanly structure their page logic in an orderly fashion (not "spaghetti code").
- The ability for development tools to provide strong WYSIWYG design support for pages (existing ASP code is opaque to tools).

ASP.NET Web Forms pages are text files with an .aspx file name extension. They can be deployed throughout an IIS virtual root directory tree. When a browser client requests .aspx resources, the ASP.NET runtime parses and compiles the target file into a .NET Framework class. This class can then be used to dynamically process incoming requests. (Note that the .aspx file is compiled only the first time it is accessed; the compiled type instance is then reused across multiple requests).

An ASP.NET page can be created simply by taking an existing HTML file and changing its file name extension to .aspx (no modification of code is required). For example, the following sample demonstrates a simple HTML page that collects a user's name and category preference and then performs a form post back to the originating page when a button is clicked:

ASP.NET provides syntax compatibility with existing ASP pages. This includes support for <% %> code render blocks that can be intermixed with HTML content within an .aspx file. These code blocks execute in a top-down manner at page render time.

Code-behind Web Forms

ASP.NET supports two methods of authoring dynamic pages. The first is the method shown in the preceding samples, where the page code is physically declared within the originating .aspx file. An alternative approach--known as the code-behind method--enables the page code to be more cleanly separated from the HTML content into an entirely separate file.

Introduction to ASP.NET Server controls

In addition to (or instead of) using <% %> code blocks to program dynamic content, ASP.NET page developers can use ASP.NET server controls to program Web pages. Server controls are declared within an .aspx file using custom tags or intrinsic HTML tags that contain a **runat="server"** attributes value. Intrinsic HTML tags are handled by one of the controls in the **System.Web.UI.HtmlControls** namespace. Any tag that doesn't explicitly map to one of the controls is assigned the type of **System.Web.UI.HtmlControls.HtmlGenericControl**.

Server controls automatically maintain any client-entered values between round trips to the server. This control state is not stored on the server (it is instead stored within an **<input type="hidden">** form field that is round-tripped between requests). Note also that no client-side script is required.

In addition to supporting standard HTML input controls, ASP.NET enables developers to utilize richer custom controls on their pages. For example, the following sample demonstrates how the **<asp:adrotator>** control can be used to dynamically display rotating ads on a page.

1. ASP.NET Web Forms provide an easy and powerful way to build dynamic Web UI.
2. ASP.NET Web Forms pages can target any browser client (there are no script library or cookie requirements).
3. ASP.NET Web Forms pages provide syntax compatibility with existing ASP pages.
4. ASP.NET server controls provide an easy way to encapsulate common functionality.
5. ASP.NET ships with 45 built-in server controls. Developers can also use controls built by third parties.
6. ASP.NET server controls can automatically project both up level and down level HTML.
7. ASP.NET templates provide an easy way to customize the look and feel of list server controls.
8. ASP.NET validation controls provide an easy way to do declarative client or server data validation.

ADO.NET Overview

ADO.NET is an evolution of the ADO data access model that directly addresses user requirements for developing scalable applications. It was designed specifically for the web with scalability, statelessness, and XML in mind.

ADO.NET uses some ADO objects, such as the **Connection** and **Command** objects, and also introduces new objects. Key new ADO.NET objects include the **DataSet**, **DataReader**, and **DataAdapter**.

The important distinction between this evolved stage of ADO.NET and previous data architectures is that there exists an object -- the **DataSet** -- that is separate and distinct from any data stores. Because of that, the **DataSet** functions as a standalone entity. You can think of the DataSet as an always disconnected record set that knows nothing about the source or destination of the data it contains. Inside a **DataSet**, much like in a database, there are tables, columns, relationships, constraints, views, and so forth.

A **DataAdapter** is the object that connects to the database to fill the **DataSet**. Then, it connects back to the database to update the data there, based on operations performed while the **DataSet** held the data. In the past, data processing has been primarily connection-based. Now, in an effort to make multi-tiered apps more efficient, data processing is turning to a message-based approach that revolves around chunks of information. At the center of this approach is the **DataAdapter**, which provides a bridge to retrieve and save data between a **DataSet** and its source data store. It accomplishes this by means of requests to the appropriate SQL commands made against the data store.

The XML-based **DataSet** object provides a consistent programming model that works with all models of data storage: flat, relational, and hierarchical. It does this by having no 'knowledge' of the source of its data, and by representing the data that it holds as collections and data types. No matter what the source of the data within the **DataSet** is, it is manipulated through the same set of standard APIs exposed through the **DataSet** and its subordinate objects.

While the **DataSet** has no knowledge of the source of its data, the managed provider has detailed and specific information. The role of the managed provider is to connect, fill, and persist the **DataSet** to and from data stores. The OLE DB and SQL Server .NET Data Providers (System.Data.OleDb and System.Data.SqlClient) that are part of the .Net Framework provide four basic objects: the **Command**, **Connection**, **DataReader** and **DataAdapter**. In the remaining sections of this document, we'll walk through each part of the **DataSet** and the OLE DB/SQL Server .NET Data Providers explaining what they are, and how to program against them.

The following sections will introduce you to some objects that have evolved, and some that are new. These objects are:

- **Connections**. For connection to and managing transactions against a database.
- **Commands**. For issuing SQL commands against a database.
- **DataReaders**. For reading a forward-only stream of data records from a SQL Server data source.
- **DataSets**. For storing, Remoting and programming against flat data, XML data and relational data.
- **DataAdapters**. For pushing data into a **DataSet**, and reconciling data against a database.

When dealing with connections to a database, there are two different options: SQL Server .NET Data Provider (System.Data.SqlClient) and OLE DB .NET Data Provider (System.Data.OleDb). In these samples we will use the SQL Server .NET Data Provider. These are written to talk directly to Microsoft SQL Server. The OLE DB .NET Data Provider is used to talk to any OLE DB provider (as it uses OLE DB underneath).

Connections:

Connections are used to 'talk to' databases, and are represented by provider-specific classes such as **SqlConnection**. Commands travel over connections and result sets are returned in the form of streams which can be read by a **DataReader** object, or pushed into a **DataSet** object.

Commands:

Commands contain the information that is submitted to a database, and are represented by provider-specific classes such as **SqlCommand**. A command can be a stored procedure call, an UPDATE statement, or a statement that returns results. You can also use input and output parameters, and return values as part of your command syntax. The example below shows how to issue an INSERT statement against the **Northwind** database.

DataReaders:

The **DataReader** object is somewhat synonymous with a read-only/forward-only cursor over data. The **DataReader** API supports flat as well as hierarchical data. A **DataReader** object is returned after executing a command against a database. The format of the returned **DataReader** object is different from a record set. For example, you might use the **DataReader** to show the results of a search list in a web page.

Datasets and DataAdapters

DataSets

The **DataSet** object is similar to the ADO **Record set** object, but more powerful, and with one other important distinction: the **DataSet** is always disconnected. The **DataSet** object represents a cache of data, with database-like structures such as tables, columns, relationships, and constraints. However, though a **DataSet** can and does behave much like a database, it is important to remember that **DataSet** objects do not interact directly with databases, or other source data. This allows the developer to work with a programming model that is always consistent, regardless of where the source data resides. Data coming from a database, an XML file, from code, or user input can all be placed into **DataSet** objects. Then, as changes are made to the **Dataset** they can be tracked and verified before updating the source data. The **GetChanges** method of the **DataSet** object actually creates a second **Dataset** that contains only the changes to the data. This **DataSet** is then used by a **DataAdapter** (or other objects) to update the original data source.

The **Dataset** has many XML characteristics, including the ability to produce and consume XML data and XML schemas. XML schemas can be used to describe schemas interchanged via Web Services. In fact, a **DataSet** with a schema can actually be compiled for type safety and statement completion.

Data adapters (OLEDB/SQL)

The **Data Adapter** object works as a bridge between the **DataSet** and the source data. Using the provider-specific **SqlDataAdapter** (along with its associated **SqlCommand** and **SqlConnection**) can increase overall performance when working with a Microsoft SQL Server databases. For other OLE DB-supported databases, you would use the **OleDbDataAdapter** object and its associated **OleDbCommand** and **OleDbConnection** objects.

The **DataAdapter** object uses commands to update the data source after changes have been made to the **DataSet**. Using the **Fill** method of the **DataAdapter** calls the SELECT command; using the **Update** method calls the INSERT, UPDATE or DELETE command for each changed row. You can explicitly set these commands in order to control the statements used at runtime to resolve changes, including the use of stored procedures. For ad-hoc scenarios, a **CommandBuilder** object can generate these at run-time based upon a select statement. However, this run-time generation requires an extra round-trip to the server in order to gather required metadata, so explicitly providing the INSERT, UPDATE, and DELETE commands at design time will result in better run-time performance.

- ADO.NET is the next evolution of ADO for the .Net Framework.
- ADO.NET was created with n-Tier, statelessness and XML in the forefront. Two new objects, the **DataSet** and **Data Adapter**, are provided for these scenarios.
- ADO.NET can be used to get data from a stream, or to store data in a cache for updates.
- There is a lot more information about ADO.NET in the documentation.
- Remember, you can execute a command directly against the database in order to do inserts, updates, and deletes. You don't need to first put data into a **Dataset** in order to insert, update, or delete it.
- Also, you can use a **Dataset** to bind to the data, move through the data, and navigate data relationships

SQL Server

A database management, or DBMS, gives the user access to their data and helps them transform the data into information. Such database management systems include dBase, paradox, IMS, SQL Server and SQL Server. These systems allow users to create, update and extract information from their database.

A database is a structured collection of data. Data refers to the characteristics of people, things and events. SQL Server stores each data item in its own fields. In SQL Server, the fields relating to a particular person, thing or event are bundled together to form a single complete unit of data, called a record (it can also be referred to as raw or an occurrence). Each record is made up of a number of fields. No two fields in a record can have the same field name.

During an SQL Server Database design project, the analysis of your business needs identifies all the fields or attributes of interest. If your business needs change over time, you define any additional fields or change the definition of existing fields.

SQL Server Tables

SQL Server stores records relating to each other in a table. Different tables are created for the various groups of information. Related tables are grouped together to form a database.

Primary Key

Every table in SQL Server has a field or a combination of fields that uniquely identifies each record in the table. The Unique identifier is called the Primary Key, or simply the Key. The primary key provides the means to distinguish one record from all other in a table. It allows the user and the database system to identify, locate and refer to one particular record in the database.

Relational Database

Sometimes all the information of interest to a business operation can be stored in one table. SQL Server makes it very easy to link the data in multiple tables. Matching an employee to the department in which they work is one example. This is what makes SQL Server a relational database management system, or RDBMS. It stores data in two or more tables and enables you to define relationships between the table and enables you to define relationships between the tables.

Foreign Key

When a field is one table matches the primary key of another field is referred to as a foreign key. A foreign key is a field or a group of fields in one table whose values match those of the primary key of another table.

Referential Integrity

Not only does SQL Server allow you to link multiple tables, it also maintains consistency between them. Ensuring that the data among related tables is correctly matched is referred to as maintaining referential integrity.

Data Abstraction

A major purpose of a database system is to provide users with an abstract view of the data. This system hides certain details of how the data is stored and maintained. Data abstraction is divided into three levels.

Physical level: This is the lowest level of abstraction at which one describes how the data are actually stored.

Conceptual Level: At this level of database abstraction all the attributed and what data are actually stored is described and entries and relationship among them.

View level: This is the highest level of abstraction at which one describes only part of the database.

Advantages of RDBMS

- Redundancy can be avoided
- Inconsistency can be eliminated
- Data can be Shared
- Standards can be enforced
- Security restrictions can be applied
- Integrity can be maintained
- Conflicting requirements can be balanced
- Data independence can be achieved.

Disadvantages of DBMS

A significant disadvantage of the DBMS system is cost. In addition to the cost of purchasing of developing the software, the hardware has to be upgraded to allow for the extensive programs and the workspace required for their execution and storage. While centralization reduces duplication, the lack of duplication requires that the database be adequately backed up so that in case of failure the data can be recovered.

Features of SQL Server (RDBMS)

SQL SERVER is one of the leading database management systems (DBMS) because it is the only Database that meets the uncompromising requirements of today's most demanding information systems. From complex decision support systems (DSS) to the most rigorous online transaction processing (OLTP) application, even application that require simultaneous DSS and OLTP access to the same critical data, SQL Server leads the industry in both performance and capability.

SQL Server is a truly portable, distributed, and open DBMS that delivers unmatched performance, continuous operation and support for every database. SQL Server RDBMS is high performance fault tolerant DBMS which is specially designed for online transactions processing and for handling large database application.

SQL SERVER with transactions processing option offers two features which contribute to very high level of transaction processing throughput, which are

- The row level lock manager

Enterprise wide data sharing

The unrivaled portability and connectivity of the SQL SERVER DBMS enables all the systems in the organization to be linked into a singular, integrated computing resource.

Portability

SQL SERVER is fully portable to more than 80 distinct hardware and operating systems platforms, including UNIX, MSDOS, OS/2, Macintosh and dozens of proprietary platforms. This portability gives complete freedom to choose the database server platform that meets the system requirements.

Open Systems

SQL SERVER offers a leading implementation of industry –standard SQL. SQL Server's open architecture integrates SQL SERVER and non –SQL SERVER DBMS with industry's most comprehensive collection of tools, application, and third party software products SQL Server's Open architecture provides transparent access to data from other relational database and even non-relational database.

Distributed Data Sharing

SQL Server's networking and distributed database capabilities to access data stored on remote server with the same ease as if the information was stored on a single local computer. A single SQL statement can access data at multiple sites. You can store data where system requirements such as performance, security or availability dictate.

Unmatched Performance

The most advanced architecture in the industry allows the SQL SERVER DBMS to deliver unmatched performance.

Sophisticated Concurrency Control

Real World applications demand access to critical data. With most database Systems application becomes "contention bound" – which performance is limited not by the CPU power or by disk I/O, but user waiting on one another for data access . SQL Server employs full, unrestricted row-level locking and contention free queries to minimize and in many cases entirely eliminates contention wait times.

No I/O Bottlenecks

SQL Server's fast commit groups commit and deferred write technologies dramatically reduce disk I/O bottlenecks. While some database write whole data block to disk at commit time, SQL Server commits transactions with at most sequential log file on disk at commit time, On high throughput systems, one sequential writes typically group commit multiple transactions. Data read by the transaction remains as shared memory so that other transactions may access that data without reading it again from disk. Since fast commits write all data necessary to the recovery to the log file, modified blocks are written back to the database independently of the transaction commit, when written from memory to disk.

2.6 Risk Assessment and Management

The below is the risk associated to this project :

When an unauthorized user tries to delete/insert/modify the data an email will be triggered and will be send to the admin.

3 Project Requirements

3.1 Identification of Requirements

<Career path-SCP2015_1_Output Design-Capability-"000001">

Output design

Outputs from computer systems are required primarily to communicate the results of processing to users. They are also used to provides a permanent copy of the results for later consultation. The various types of outputs in general are:

- External Outputs, whose destination is outside the organization.
- Internal Outputs whose destination is within organization and they are the
- User's main interface with the computer.
- Operational outputs whose use is purely within the computer department.

- Interface outputs, which involve the user in communicating directly with

<Career path-SCP2015_1_Output Definition-Capability-“000002”>

The outputs should be defined in terms of the following points:

- Type of the output
- Content of the output
- Format of the output
- Location of the output
- Frequency of the output
- Volume of the output
- Sequence of the output

It is not always desirable to print or display data as it is held on a computer. It should be decided as which form of the output is the most suitable.

For Example

- Will decimal points need to be inserted
- Should leading zeros be suppressed.

<Career path-SCP2015_1_Output Media-Capability-“000003”>

In the next stage it is to be decided that which medium is the most appropriate for the output. The main considerations when deciding about the output media are:

- The response time required.
- The location of the users
- The software and hardware available.

Keeping in view the above description the project is to have outputs mainly coming under the category of internal outputs. The main outputs desired according to the requirement specification are:

The outputs were needed to be generated as a hot copy and as well as queries to be viewed on the screen. Keeping in view these outputs, the format for the output is taken from the outputs, which are currently being obtained after manual processing. The standard printer is to be used as output media for hard copies.

<Career path-SCP2015_1_Input Design-Capability-“000004”>

Input design is a part of overall system design. The main objective during the input design is as given below:

- To produce a cost-effective method of input.
- To achieve the highest possible level of accuracy.
- To ensure that the input is acceptable and understood by the user.

<Career path-SCP2015_1_Input Stages-Capability-“000005”>

The main input stages can be listed as below:

- Data recording
- Data transcription
- Data conversion
- Data verification
- Data control
- Data transmission

- Data validation
- Data correction

<Career path-SCP2015_1_Input Types-Capability-“000006”>

It is necessary to determine the various types of inputs. Inputs can be categorized as follows:

- External inputs, which are prime inputs for the system.
- Internal inputs, which are user communications with the system.
- Operational, which are computer department’s communications to the system?
- Interactive, which are inputs entered during a dialogue.

<Career path-SCP2015_1_Input Media-Capability-“000007”>

At this stage choice has to be made about the input media. To conclude about the input media consideration has to be given to:

- Type of input
- Flexibility of format
- Speed
- Accuracy
- Verification methods
- Rejection rates
- Ease of correction
- Storage and handling requirements
- Security
- Easy to use
- Portability

Keeping in view the above description of the input types and input media, it can be said that most of the inputs are of the form of internal and interactive. As Input data is to be the directly keyed in by the user, the keyboard can be considered to be the most suitable input device.

<Career path-SCP2015_1_Error Avoidance-Capability-“000008”>

Avoiding errors

- At this stage care is to be taken to ensure that input data remains accurate form the stage at which it is recorded upto the stage in which the data is accepted by the system. This can be achieved only by means of careful control each time the data is handled.

<Career path-SCP2015_1_Error Detection-Capability-“000009”>

Detecting Errors

- Even though every effort is make to avoid the occurrence of errors, still a small proportion of errors is always likely to occur, these types of errors can be discovered by using validations to check the input data

<Career path-SCP2015_1_Data Validation-Capability-“0000010”>

Validating the data

- Procedures are designed to detect errors in data at a lower level of detail. Data validations have been included in the system in almost every area where there is a possibility for the user to commit errors. The system will not accept invalid

data. Whenever an invalid data is keyed in, the system immediately prompts the user and the user has to again key in the data and the system will accept the data only if the data is correct. Validations have been included where necessary.

- The system is designed to be a user friendly one. In other words the system has been designed to communicate effectively with the user. The system has been designed with popup menus.

<Career path-SCP2015_1_User Interface Design-Capability-"0000011">

User Interface Design

- It is essential to consult the system users and discuss their needs while designing the user interface

<Career path-SCP2015_1_User Interface Systems Can be Broadly Classified As-Capability-"0000012">

User Interface Systems Can be Broadly Classified As

- User initiated interface the user is in charge, controlling the progress of the user/computer dialogue. In the computer-initiated interface, the computer selects the next stage in the interaction.
- Computer initiated interfaces
- In the computer initiated interfaces the computer guides the progress of the user/computer dialogue. I Information is displayed and the user response of the computer takes action or displays further information.

<Career path-SCP2015_1_User Initiated Interfaces -Capability-"0000013">

User Initiated Interfaces

User initiated interfaces fall into tow approximate classes:

- Command driven interfaces: In this type of interface the user inputs commands or queries which are interpreted by the computer.
- Forms oriented interface: The user calls up an image of the form to his/her screen and fills in the form. The forms oriented interface is chosen because it is the best choice.

<Career path-SCP2015_1_Computer-Initiated Interfaces-Capability-"0000014">

Computer-Initiated Interfaces

The following computer – initiated interfaces were used:

- The menu system for the user is presented with a list of alternatives and the user chooses one; of alternatives.
- Questions – answer type dialog system where the computer asks question and takes action based on the basis of the users reply.

Right from the start the system is going to be menu driven, the opening menu displays the available options. Choosing one option gives another popup menu with more options. In this way every option leads the users to data entry form where the user can key in the data.

<Career path-SCP2015_1_Error Message Design-Capability-"0000014">

Error Message Design

- The design of error messages is an important part of the user interface design. As user is bound to commit some errors or other while designing a system the system should be designed to be helpful by providing the user with information regarding the error he/she has committed.
- This application must be able to produce output at different modules for different inputs.

3.2 Operations, Administration, Maintenance and Provisioning (OAM&P)

1. A person should be able to
    - Access/ Search CVs/information from the first page (only read access).

- login to the system through the first page of the application
- change the password after logging into the system
- Upload his/her CV.
- See/change his/her details.
- Get help about the application on how to use the different features of the system.

2. An admin login should be present who can read as well as remove any uploads.

In the flexibility of the uses the interface has been developed a graphics concept in mind, associated through a browser interface. The GUI'S at the top level have been categorized as

- Administrative user interface
- The operational or generic user interface

The administrative user interface concentrates on the consistent information that is practically, part of the organizational activities and which needs proper authentication for the data collection. The interfaces help the administrations with all the transactional states like Data insertion, Data deletion and Date updation along with the extensive data search capabilities.

The operational or generic user interface helps the users upon the system in transactions through the existing data and required services. The operational user interface also helps the ordinary users in managing their own information helps the ordinary users in managing their own information in a customized manner as per the assisted flexibilities.

Number of Modules

The system after careful analysis has been identified to be presented with the following modules:

The modules involved are:

- Admin
- Job Seeker
- Job Provider
- Notification
- Search
- Report
- Authentication

Admin

In this module Admin will add all the qualifications, skill, experience, city, state, country and update and delete information about the job provider or job seeker he can also search for the job seeker and he can send mail to offer the job to job seeker and he can also see the jobs add by the job provider.

Job Seeker

In this module Job Seeker register himself and upload his resume and fill the profile give by admin and after login he will search for the job on various conditions and he can change his profiles and resume and he can apply for the jobs based on various conditions. He can see the response of the company and he can call the company person for the interview.

Job provider

In this module Job Provider register himself and his company and after login he will add new job and he can search for the job seekers on various condition and he can offer the job to job seeker according to the job profile and he can also see the response from the job seekers and send the mail.

Notification

In this module admin and job provider send the notification to the job seeker in the form of email.

Reports

This module contains all the information about the reports generated by the admin based on the particular job seeker, particular job provider, all job seeker and job provider, all jobs generated by the job providers.

Authentication

This module contains all the information about the authenticated user. User without his username and password can't enter into the login if he is only the authenticated user then he can enter to his login.

Inputs & Outputs

The main inputs, outputs and major functions of the system are as follows.

Inputs:

- Admin enters his or her user id and password.
- Users enter his or her user id and password.
- User while registration can add complete education details, skills..etc.,
- User requests the reports.
- User requests the search.
- Admin can edit the queries based on the jobs.

Outputs:

- Admin receives survey details.
- Users receive gives the survey details.
- Users can see the requested information check the status of that.
- user receive the requested question and they give the answer
- Displays requested result.

Advantages

- Access/ Search CVs/information from the first page (only read access).
- login to the system through the first page of the application
- change the password after logging into the system
- Upload his/her CV.
- See/change his/her details.
- Get help about the application on how to use the different features of the system.

SDLC Methodologies

This Document plays a vital role in the development life cycle (SDLC) as it describes the complete requirement of the system. It is meant for use by the developers and will be the basic during testing phase. Any changes made to the requirements in the future will have to go through formal change approval process.

WATER FALL MODEL was being chosen because all requirements were known beforehand and the objective of our software development is the computerization/automation of an already existing manual working system.

3.3 Security and Fraud Prevention

The protection of computer based resources that includes hardware, software, data, procedures and people against unauthorized use or natural

Disaster is known as System Security.

System Security can be divided into four related issues:

- Security
- Integrity
- Privacy
- Confidentiality

System Security refers to the technical innovations and procedures applied to the hardware and operation systems to protect against deliberate or accidental damage from a defined threat.

Data Security is the protection of data from loss, disclosure, modification and destruction.

System Integrity refers to the power functioning of hardware and programs, appropriate physical security and safety against external threats such as eavesdropping and wiretapping.

Privacy defines the rights of the user or organizations to determine what information they are willing to share with or accept from others and how the organization can be protected against unwelcome, unfair or excessive dissemination of information about it.

Confidentiality is a special status given to sensitive information in a database to minimize the possible invasion of privacy. It is an attribute of information that characterizes its need for protection.

Security in software

System security refers to various validations on data in form of checks and controls to avoid the system from failing. It is always important to ensure that only valid data is entered and only valid operations are performed on the system. The system employees two types of checks and controls:

Client Side Validation

Various client side validations are used to ensure on the client side that only valid data is entered. Client side validation saves server time and load to handle invalid data. Some checks imposed are:

- VBScript in used to ensure those required fields are filled with suitable data only. Maximum lengths of the fields of the forms are appropriately defined.
- Forms cannot be submitted without filling up the mandatory data so that manual mistakes of submitting empty fields that are mandatory can be sorted out at the client side to save the server time and load.
- Tab-indexes are set according to the need and taking into account the ease of user while working with the system.

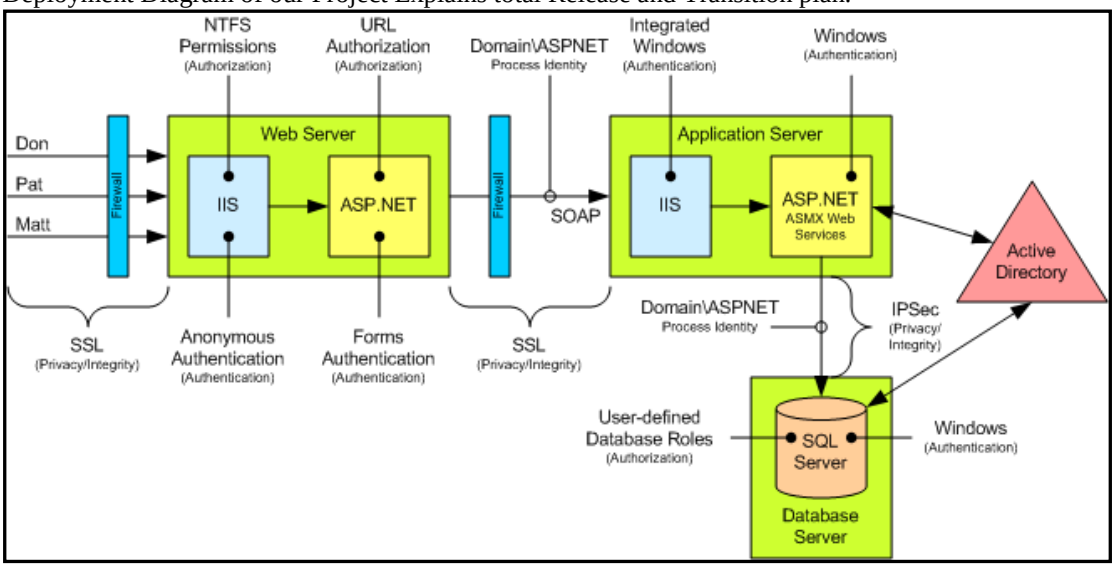
Server Side Validation

Some checks cannot be applied at client side. Server side checks are necessary to save the system from failing and intimating the user that some invalid operation has been performed or the performed operation is restricted. Some of the server side checks imposed is:

- Server side constraint has been imposed to check for the validity of primary key and foreign key. A primary key value cannot be duplicated. Any attempt to duplicate the primary value results into a message intimating the user about those values through the forms using foreign key can be updated only of the existing foreign key values.
- User is intimating through appropriate messages about the successful operations or exceptions occurring at server side.
- Various Access Control Mechanisms have been built so that one user may not agitate upon another. Access permissions to various types of users are controlled according to the organizational structure. Only permitted users can log on to the system and can have access according to their category. User- name, passwords and permissions are controlled o the server side.
- Using server side validation, constraints on several restricted operations are imposed.

3.4 Release and Transition Plan

Deployment Diagram of our Project Explains total Release and Transition plan.



4 Project Design Description

Software design sits at the technical kernel of the software engineering process and is applied regardless of the development paradigm and area of application. Design is the first step in the development phase for any engineered product or system. The designer's goal is to produce a model or representation of an entity that will later be built. Beginning, once system requirement have been specified and analyzed, system design is the first of the three technical activities -design, code and test that is required to build and verify software.

The importance can be stated with a single word "Quality". Design is the place where quality is fostered in software development. Design provides us with representations of software that can assess for quality. Design is the only way that we can accurately translate a customer's view into a finished software product or system. Software design serves as a foundation for all the software engineering steps that follow. Without a strong design we risk building an unstable system – one that will be difficult to test, one whose quality cannot be assessed until the last stage.

During design, progressive refinement of data structure, program structure, and procedural details are developed reviewed and documented. System design can be viewed from either technical or project management perspective. From the technical point of view, design is comprised of four activities – architectural design, data structure design, interface design and procedural design

Output Design

Outputs from computer systems are required primarily to communicate the results of processing to users. They are also used to provides a permanent copy of the results for later consultation. The various types of outputs in general are:

- External Outputs, whose destination is outside the organization.
- Internal Outputs whose destination is within organization and they are the
- User's main interface with the computer.
- Operational outputs whose use is purely within the computer department.
- Interface outputs, which involve the user in communicating directly with

Output Definition

The outputs should be defined in terms of the following points:

- Type of the output
- Content of the output
- Format of the output
- Location of the output
- Frequency of the output
- Volume of the output
- Sequence of the output

It is not always desirable to print or display data as it is held on a computer. It should be decided as which form of the output is the most suitable.

For Example

- Will decimal points need to be inserted
- Should leading zeros be suppressed.

Output Media:

In the next stage it is to be decided that which medium is the most appropriate for the output. The main considerations when deciding about the output media are:

- The suitability for the device to the particular application.
- The need for a hard copy.
- The response time required.
- The location of the users
- The software and hardware available.

Keeping in view the above description the project is to have outputs mainly coming under the category of internal outputs. The main outputs desired according to the requirement specification are:

The outputs were needed to be generated as a hot copy and as well as queries to be viewed on the screen. Keeping in view these outputs, the format for the output is taken from the outputs, which are currently being obtained after manual processing. The standard printer is to be used as output media for hard copies.

Input Design

Input design is a part of overall system design. The main objective during the input design is as given below:

- To produce a cost-effective method of input.
- To achieve the highest possible level of accuracy.
- To ensure that the input is acceptable and understood by the user.

Input Stages

The main input stages can be listed as below:

- Data recording
- Data transcription
- Data conversion
- Data verification
- Data control
- Data transmission
- Data validation

- Data correction

Input Types

It is necessary to determine the various types of inputs. Inputs can be categorized as follows:

- External inputs, which are prime inputs for the system.
- Internal inputs, which are user communications with the system.
- Operational, which are computer department's communications to the system?
- Interactive, which are inputs entered during a dialogue.

Input Media

At this stage choice has to be made about the input media. To conclude about the input media consideration has to be given to:

- Type of input
- Flexibility of format
- Speed
- Accuracy
- Verification methods
- Rejection rates
- Ease of correction
- Storage and handling requirements
- Security
- Easy to use
- Portability

Keeping in view the above description of the input types and input media, it can be said that most of the inputs are of the form of internal and interactive. As

Input data is to be the directly keyed in by the user, the keyboard can be considered to be the most suitable input device.

Error Avoidance

At this stage care is to be taken to ensure that input data remains accurate form the stage at which it is recorded upto the stage in which the data is accepted by the system. This can be achieved only by means of careful control each time the data is handled.

Error Detection

Even though every effort is make to avoid the occurrence of errors, still a small proportion of errors is always likely to occur, these types of errors can be discovered by using validations to check the input data.

Data Validation

Procedures are designed to detect errors in data at a lower level of detail. Data validations have been included in the system in almost every area where there is a possibility for the user to commit errors. The system will not accept invalid data. Whenever an invalid data is keyed in, the system immediately prompts the user and the user has to again key in the data and the system will accept the data only if the data is correct. Validations have been included where necessary.

The system is designed to be a user friendly one. In other words the system has been designed to communicate effectively with the user. The system has been designed with popup menus.

Normalization

It is a process of converting a relation to a standard form. The process is used to handle the problems that can arise due to data redundancy i.e. repetition of data in the database, maintain data integrity as well as handling problems that can arise due to insertion, updating, deletion anomalies.

Decomposing is the process of splitting relations into multiple relations to eliminate anomalies and maintain anomalies and maintain data integrity. To do this we use normal forms or rules for structuring relation.

Insertion anomaly: Inability to add data to the database due to absence of other data.

Deletion anomaly: Unintended loss of data due to deletion of other data.

Update anomaly: Data inconsistency resulting from data redundancy and partial update

Normal Forms: These are the rules for structuring relations that eliminate anomalies.

First Normal Form

A relation is said to be in first normal form if the values in the relation are atomic for every attribute in the relation. By this we mean simply that no attribute value can be a set of values or, as it is sometimes expressed, a repeating group.

Second Normal form

A relation is said to be in second Normal form is it is in first normal form and it should satisfy any one of the following rules.

- Primary key is a not a composite primary key
- No non key attributes are present
- Every non key attribute is fully functionally dependent on full set of primary key.

Third Normal Form

A relation is said to be in third normal form if their exits no transitive dependencies.

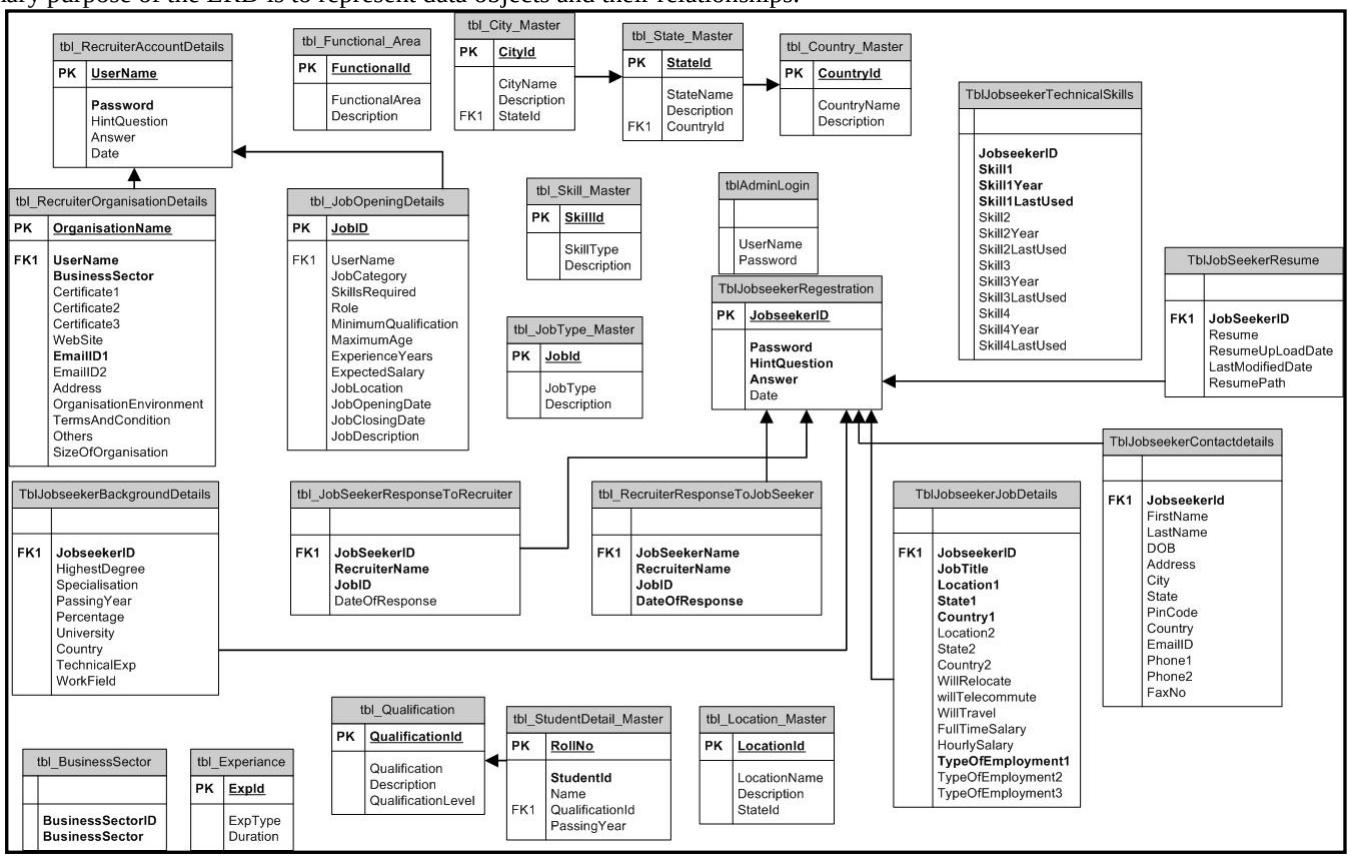
Transitive Dependency: If two non key attributes depend on each other as well as on the primary key then they are said to be transitively dependent.

The above normalization principles were applied to decompose the data in multiple tables thereby making the data to be maintained in a consistent state.

E – R Diagrams

- The relation upon the system is structure through a conceptual ER-Diagram, which not only specifics the existential entities but also the standard relations through which the system exists and the cardinalities that are necessary for the system state to continue.
- The entity Relationship Diagram (ERD) depicts the relationship between the data objects. The ERD is the notation that is used to conduct the date modeling activity the attributes of each data object noted is the ERD can be described resign a data object descriptions.
- The set of primary components that are identified by the ERD are
  - ◆ Data object ◆ Relationships
  - ◆ Attributes ◆ Various types of indicators.

The primary purpose of the ERD is to represent data objects and their relationships.



Unified Modeling Language Diagrams

- The unified modeling language allows the software engineer to express an analysis model using the modeling notation that is governed by a set of syntactic semantic and pragmatic rules.
- A UML system is represented using five different views that describe the system from distinctly different perspective. Each view is defined by a set of diagram, which is as follows.
- User Model View

    i. This view represents the system from the users perspective.

    ii. The analysis representation describes a usage scenario from the end-users perspective.

Structural model view

- In this model the data and functionality are arrived from inside the system.
- This model view models the static structures.

Behavioral Model View

- It represents the dynamic of behavioral as parts of the system, depicting the interactions of collection between various structural elements described in the user model and structural model view.

Implementation Model View

- In this the structural and behavioral as parts of the system are represented as they are to be built.

Environmental Model View

In this the structural and behavioral aspects of the environment in which the system is to be implemented are represented.

UML is specifically constructed through two different domains they are

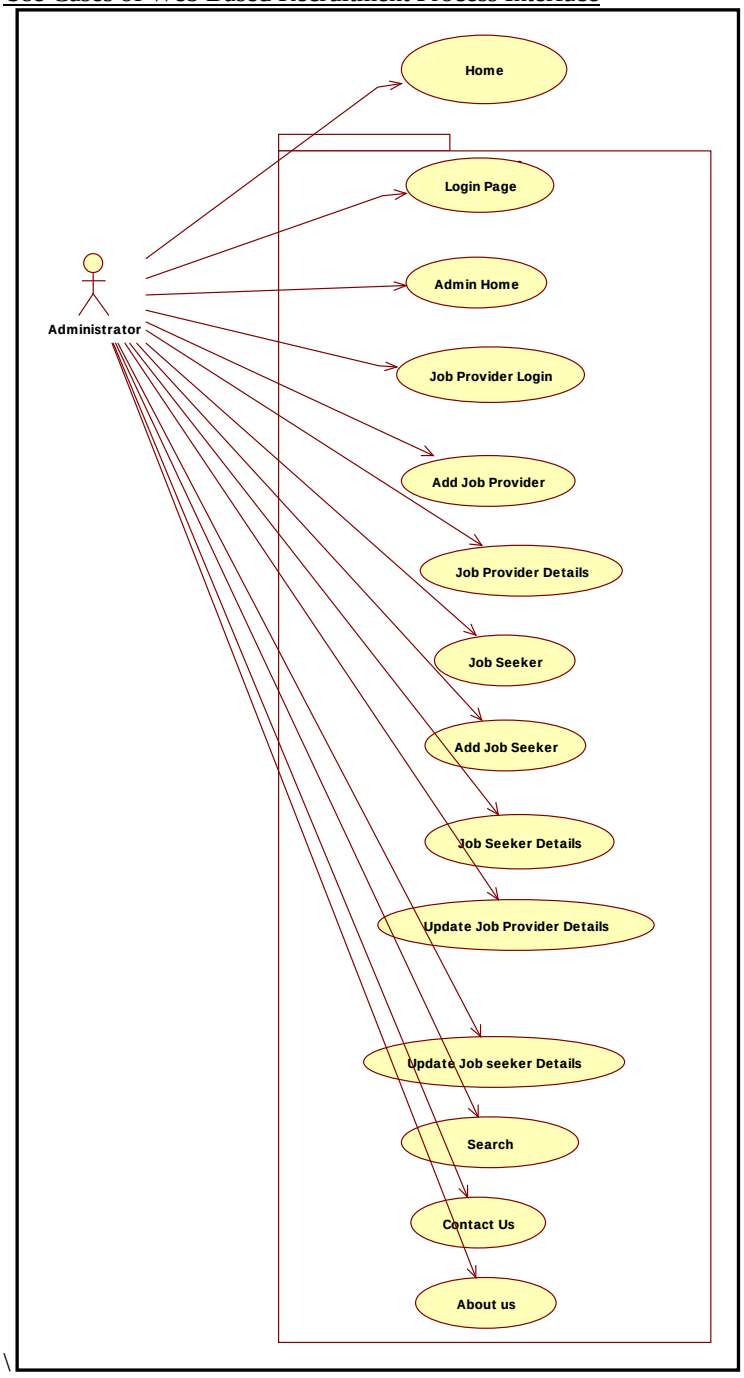
- UML Analysis modeling, which focuses on the user model and structural model views of the system.
- UML design modeling, which focuses on the behavioral modeling, implementation modeling and environmental model views.

Use case Diagrams represent the functionality of the system from a user's point of view. Use cases are used during requirements elicitation and analysis to represent the functionality of the system. Use cases focus on the behavior of the system from external point of view.

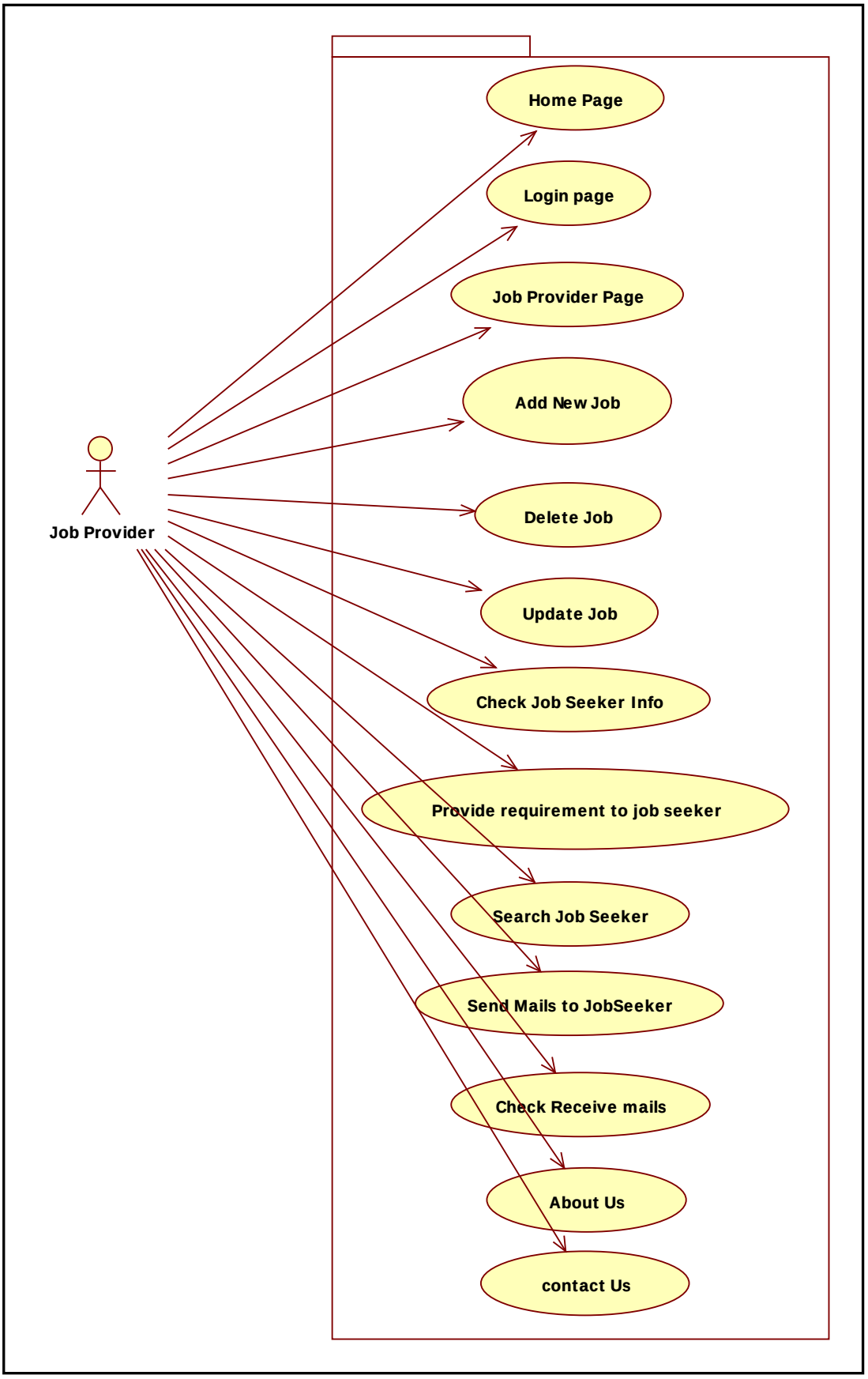
Actors are external entities that interact with the system. Examples of actors include users like administrator, bank customer …etc., or another system like central database.

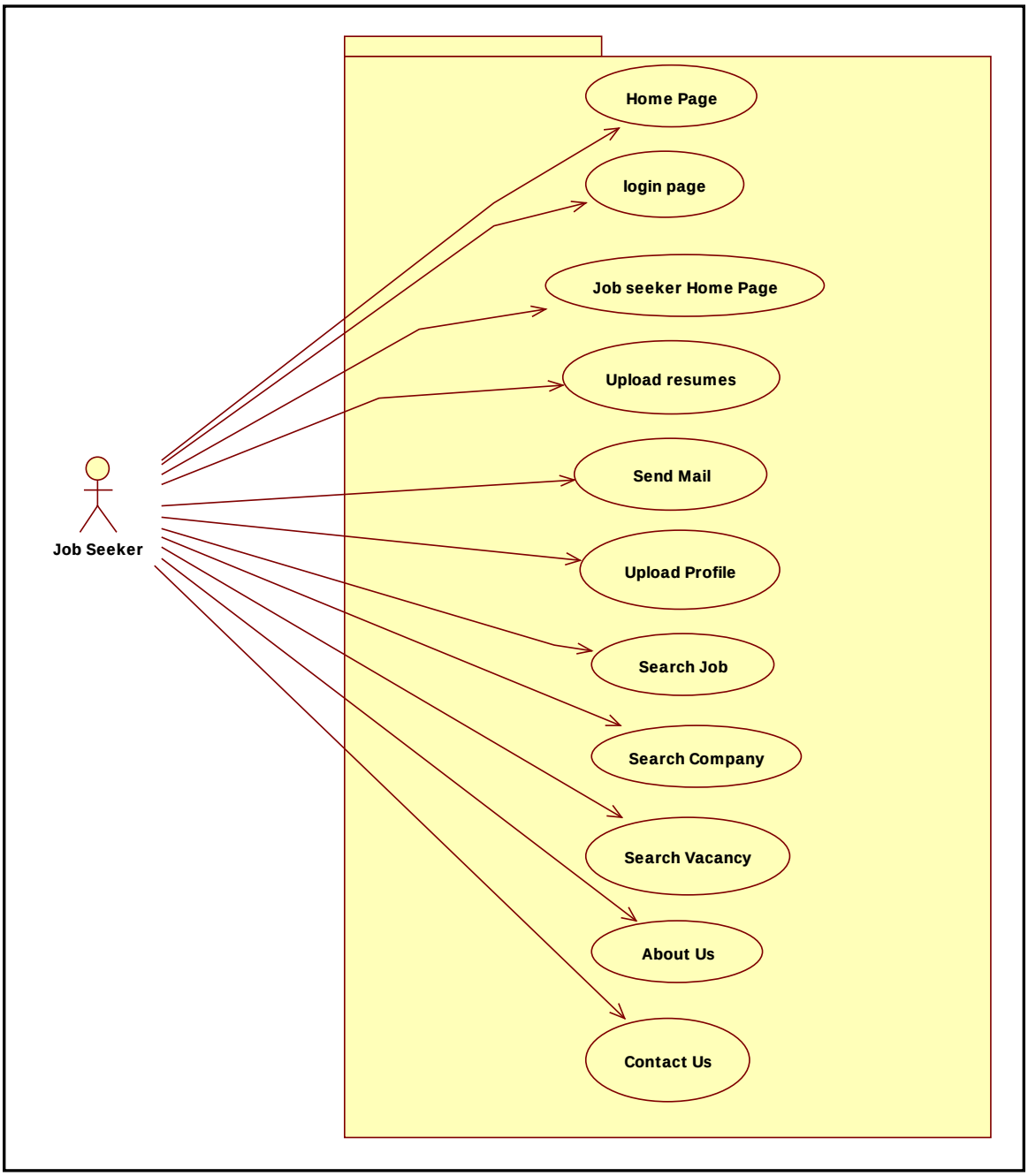
Use case Model

Use Cases of Web Based Recruitment Process Interface



\


Home Page
Login page
Job Provider Page
Add New Job
Job Provider
Delete Job
Update Job
Check Job Seeker Info
Provide requirement to job seeker
Search Job Seeker
Send Mails to JobSeeker
Check Receive mails
About Us
contact Us


Home Page
login page
Job seeker Home Page
Upload resumes
Send Mail
Upload Profile
Search Job
Search Company
Search Vacancy
About Us
Contact Us
Job Seeker

Use case name	Login
Participating actors	Admin, Job Provider, Job Seeker
Flow of events	The Actor will give the user name and password to the system. The system will verify the authentication.
Entry Condition	The actor will enter the system by using username and password
Exit condition	If un authenticated should be exited
Quality Requirements	Password must satisfy the complexity requirements.

Use case name	Admin Registration
Participating actors	Admin
Flow of events	The Admin will submit all the details and place in the application.
Entry Condition	Must satisfy all the norms given by the Carrier Path interface site.
Exit condition	Successful or Un successful completion of creation of account.
Quality Requirements	All fields are mandatory.

Use case name	Job Provider Registration
Participating actors	Job Provider
Flow of events	The Job Provider must enter all his personal details.
Entry Condition	View Home page
Exit condition	Registered Job Provider should be successfully logged out. Error Message should be displayed on Un successful creation.
Quality Requirements	Best Error Handling techniques. Check on Mandatory fields.

Use case name	Job Seeker Registration
Participating actors	Job Seeker
Flow of events	The Job Seeker must enter all his personal details
Entry Condition	View Home Page
Exit condition	Registered Job Seeker should be successfully logged out. Error Message should be displayed on Un successful creation.
Quality Requirements	Best Error Handling techniques. Check on Mandatory fields.

5 Project Internal/external Interface Impacts and Specification

GUI'S

In the flexibility of the uses the interface has been developed a graphics concept in mind, associated through a browses interface. The GUI'S at the top level have been categorized as

Administrative user interface

The operational or generic user interface

The administrative user interface concentrates on the consistent information that is practically, part of the organizational activities and which needs proper authentication for the data collection. The interfaces help the administrations with all the transactional states like Data insertion, Data deletion and Date updation along with the extensive data search capabilities.

The operational or generic user interface helps the users upon the system in transactions through the existing data and required services. The operational user interface also helps the ordinary users in managing their own information helps the ordinary users in managing their own information in a customized manner as per the assisted flexibilities.

6 Project Design Units Impacts

Input and Output

The main inputs, outputs and major functions of the system are as follows

Inputs

Head operator enters his or her user id and password.

Operators enter his or her user id and password.

- Technicians enter his or her user id and password.
- Sub technicians enter his or her user id and password.
- User requests the reports.
- User requests the search.
- Head operator can edits the personal details and so on.

Outputs

- Head operator receives personal details.
- Operator receives the personal details.
- Technicians receive personal and technical details.
- Users receive requested reports.
- Displays search result.

A person should be able to

- Access/ Search CVs/information from the first page (only read access).
- login to the system through the first page of the application
- change the password after logging into the system
- Upload his/her CV.
- See/change his/her details.
- Get help about the application on how to use the different features of the system.

An admin login should be present who can read as well as remove any uploads.

In the flexibility of the uses the interface has been developed a graphics concept in mind, associated through a browser interface. The GUI'S at the top level have been categorized as

Administrative user interface

The operational or generic user interface

The administrative user interface concentrates on the consistent information that is practically, part of the organizational activities and which needs proper authentication for the data collection. The interfaces help the administrations with all the transactional states like Data insertion, Data deletion and Date updation along with the extensive data search capabilities.

The operational or generic user interface helps the users upon the system in transactions through the existing data and required services. The operational user interface also helps the ordinary users in managing their own information helps the ordinary users in managing their own information in a customized manner as per the assisted flexibilities.

6.1 Functional Area/Design Unit A

6.1.1 Functional Overview

Software design sits at the technical kernel of the software engineering process and is applied regardless of the development paradigm and area of application. Design is the first step in the development phase for any engineered product or system. The designer's goal is to produce a model or representation of an entity that will later be built. Beginning, once system requirement have been specified and analyzed, system design is the first of the three technical activities -design, code and test that is required to build and verify software.

The importance can be stated with a single word "Quality". Design is the place where quality is fostered in software development. Design provides us with representations of software that can assess for quality. Design is the only way that we can accurately translate a customer's view into a finished software product or system. Software design serves as a foundation for all the software engineering steps that follow. Without a strong design we risk building an unstable system – one that will be difficult to test, one whose quality cannot be assessed until the last stage.

During design, progressive refinement of data structure, program structure, and procedural details are developed reviewed and documented. System design can be viewed from either technical or project management perspective. From the technical point of view, design is comprised of four activities – architectural design, data structure design, interface design and procedural design.

- Based on the given requirements, conceptualize the Solution Architecture.
- Choose the domain of your interest otherwise develop the application for ultimatedotnet.com.
- Depict the various architectural components, show interactions and connectedness and show internal and external elements.
- Design the web services, web methods and database infrastructure needed both and client and server.
- Provide an environment for upgradation of application for newer versions that are available in the same domain as web service target.

6.1.2 Impacts

This project does not have any functional area impacts but the below are the mandatory requirements for developing this project.

Software and Hardware Requirements:

Software Specifications

- Microsoft .NET framework 4.0
- Language: C#.Net, Asp.net
- Microsoft Visual Studio 2008IDE or higher
- OS: Microsoft Windows 7 or higher

Hardware Specifications.

- P IV Processor or higher
- 512 MB RAM minimum
- Secondary memory of 20 – 50 GB minimum

6.1.3 Requirements

Purpose of the system

This system can be used as an Online Job Portal for the Placements providing to the un employees who are seeking for a job placement. Job Seeker logging into the system and he can should be able to upload their information in the form of a CV. Visitors/Company representatives logging in may also access/search any information put up by Job Seeker.

Problems in the existing system

It is limited to a single system. It is less user-friendly. It is having lots of manual work (Manual system does not mean that you are working with pen and paper, it also include working on spread sheets and other simple software's). The present system is very less secure. It is unable to see the different kinds of survey. It doesn't have the mail and file upload feature.

Solution to these Problems

The development of the new system contains the following activities, which try to automate the entire process keeping in view of the database integration approach.

User friendliness is provided in the application with various controls. The system makes the overall project management much easier and flexible. It can be accessed over the Internet. Various classes have been used to provide file upload and mail features. There is no risk of data mismanagement at any level while the project development is under process. Report generation feature is provided using grid view control in ASP.NET.

Advantages

The project can be easily used in the process of decision making. Different types of surveys can be generated which help the management to take correct decision and reduce the time delay which automatically increases the company's work standards as well as the economical state of the company.

Functional Requirements

A person should be able to

- Access/ Search CVs/information from the first page (only read access).
- login to the system through the first page of the application
- change the password after logging into the system
- Upload his/her CV.
- See/change his/her details.
- Get help about the application on how to use the different features of the system.

An admin login should be present who can read as well as remove any uploads.

Inputs & Outputs

The main inputs, outputs and major functions of the system are as follows.

Inputs:

- Admin enters his or her user id and password.
- Users enter his or her user id and password.
- User while registration can add complete education details, skills..etc.,
- User requests the reports.
- User requests the search.
- Admin can edit the queries based on the jobs.

Outputs:

- Admin receives survey details.

- Users receive gives the survey details.
- Users can see the requested information check the status of that.
- user receive the requested question and they give the answer
- Displays requested result.

Advantages

- Access/ Search CVs/information from the first page (only read access).
- login to the system through the first page of the application
- change the password after logging into the system
- Upload his/her CV.
- See/change his/her details.
- Get help about the application on how to use the different features of the system.

SDLC Methodologies

This Document plays a vital role in the development life cycle (SDLC) as it describes the complete requirement of the system. It is meant for use by the developers and will be the basic during testing phase. Any changes made to the requirements in the future will have to go through formal change approval process.

WATER FALL MODEL was being chosen because all requirements were known beforehand and the objective of our software development is the computerization/automation of an already existing manual working system.

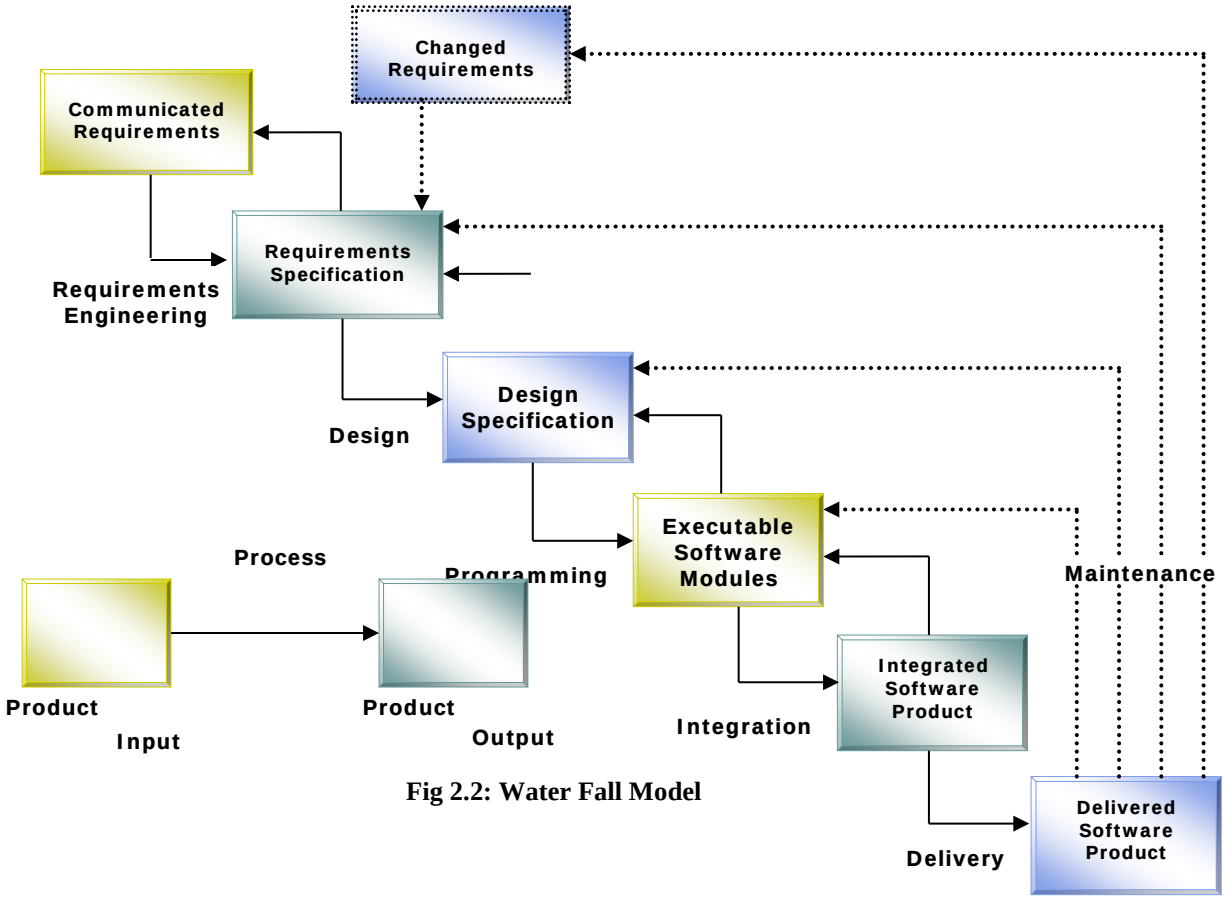


Fig 2.2: Water Fall Model

Data dictionary

After carefully understanding the requirements of the client the entire data storage requirements are divided into tables. The below tables are normalized to avoid any anomalies during the course of data entry.

Column Name	Data Type	Length	Allow Nulls
BusinessSectorID	int	4	
BusinessSector	varchar	50	

City Details

Column Name	Data Type	Length	Allow Nulls
CityId	int	4	
CityName	varchar	50	✓
Description	varchar	80	✓
StateId	int	4	✓

Country Details

Column Name	Data Type	Length	Allow Nulls
CountryId	int	4	
CountryName	varchar	50	✓
Description	varchar	80	✓

Experience Detail

Column Name	Data Type	Length	Allow Nulls
ExpId	int	4	
ExpType	varchar	50	✓
Duration	char	10	✓

Functional Area

Column Name	Data Type	Length	Allow Nulls
FunctionalId	int	4	
FunctionalArea	varchar	50	✓
Description	varchar	50	✓

Job Opening Detail

Column Name	Data Type	Length	Allow Nulls
UserName	nvarchar	50	✓
JobID	nvarchar	50	
JobCategory	nvarchar	50	✓
SkillsRequired	nvarchar	100	✓
Role	nvarchar	15	✓
MinimumQualification	nvarchar	50	✓
MaximumAge	tinyint	1	✓
ExperienceYears	tinyint	1	✓
ExpectedSalary	money	8	✓
JobLocation	nvarchar	50	✓
JobOpeningDate	datetime	8	✓
JobClosingDate	datetime	8	✓
JobDescription	nvarchar	2000	✓

Jobseeker Response to Recruiter

Column Name	Data Type	Length	Allow Nulls
JobSeekerID	varchar	50	
RecruiterName	varchar	50	
JobID	varchar	50	
DateOfResponse	datetime	8	✓

Job Type Detail

Column Name	Data Type	Length	Allow Nulls
JobId	int	4	
JobType	varchar	50	✓
Description	varchar	80	✓

Location Master

Column Name	Data Type	Length	Allow Nulls
LocationId	int	4	
LocationName	varchar	50	✓
Description	varchar	80	✓
StateId	int	4	✓

Qualification

Column Name	Data Type	Length	Allow Nulls
QualificationId	int	4	
Qualification	varchar	50	✓
Description	varchar	50	✓
QualificationLevel	int	4	✓

Recruiter Account Details

Column Name	Data Type	Length	Allow Nulls
UserName	nvarchar	50	
Password	nvarchar	50	
HintQuestion	nvarchar	100	✓
Answer	nvarchar	100	✓
[Date]	datetime	8	✓

Recruiter Organization Details

Column Name	Data Type	Length	Allow Nulls
UserName	nvarchar	50	
OrganisationName	nvarchar	50	
BusinessSector	nvarchar	50	
Certificate1	nvarchar	50	✓
Certificate2	nvarchar	50	✓
Certificate3	nvarchar	50	✓
WebSite	nvarchar	100	✓
EmailID1	nvarchar	100	
EmailID2	nvarchar	100	✓
Address	nvarchar	1024	✓
OrganisationEnvironm	nvarchar	2000	✓
TermsAndCondition	nvarchar	2000	✓
Others	nvarchar	2000	✓
SizeOfOrganisation	int	4	✓

Recruiter Response to Jobseeker

Column Name	Data Type	Length	Allow Nulls
JobSeekerName	varchar	50	
RecruiterName	varchar	50	
JobID	varchar	50	
DateOfResponse	datetime	8	

Skill Master

Column Name	Data Type	Length	Allow Nulls
SkillId	int	4	
SkillType	varchar	50	✓
Description	varchar	50	✓

State detail

Column Name	Data Type	Length	Allow Nulls
StateId	int	4	
StateName	varchar	50	✓
Description	varchar	80	✓
CountryId	int	4	✓

Student Detail

Column Name	Data Type	Length	Allow Nulls
StudentId	int	4	
RollNo	varchar	50	
Name	varchar	50	✓
QualificationId	int	4	✓
PassingYear	varchar	50	✓

Admin Login

Column Name	Data Type	Length	Allow Nulls
UserName	varchar	50	✓
Password	varchar	50	✓

Jobseeker Background Details

Column Name	Data Type	Length	Allow Nulls
JobseekerID	varchar	50	
HighestDegree	varchar	50	✓
Specialisation	varchar	50	✓
PassingYear	int	4	✓
Percentage	float	8	✓
University	varchar	50	✓
Country	varchar	50	✓
TechnicalExp	varchar	50	✓
WorkField	varchar	25	✓

Contact Details

Column Name	Data Type	Length	Allow Nulls
JobseekerId	varchar	50	
FirstName	char	20	✓
LastName	char	20	✓
DOB	datetime	8	✓
Address	varchar	50	✓
City	varchar	50	✓
State	varchar	50	✓
PinCode	int	4	✓
Country	varchar	50	✓
EmailID	varchar	25	✓
Phone1	varchar	20	✓
Phone2	varchar	20	✓
FaxNo	varchar	20	✓

Jobseeker Job Details

Column Name	Data Type	Length	Allow Nulls
JobseekerID	varchar	50	
JobTitle	varchar	50	
Location1	varchar	50	
State1	varchar	50	
Country1	varchar	50	
Location2	varchar	50	✓
State2	varchar	50	✓
Country2	varchar	50	✓
WillRelocate	varchar	50	✓
willTelecommute	varchar	50	✓
WillTravel	varchar	50	✓
FullTimeSalary	varchar	50	✓
HourlySalary	varchar	50	✓
TypeOfEmployment1	varchar	80	
TypeOfEmployment2	varchar	80	✓
TypeOfEmployment3	varchar	80	✓

Jobseeker Registration

Column Name	Data Type	Length	Allow Nulls
JobseekerID	varchar	50	
Password	varchar	50	
HintQuestion	varchar	50	
Answer	varchar	50	
[Date]	datetime	8	✓

Jobseeker Resume

Column Name	Data Type	Length	Allow Nulls
JobSeekerID	varchar	50	
Resume	ntext	16	✓
ResumeUpLoadDate	datetime	8	✓
LastModifiedDate	datetime	8	✓
ResumePath	varchar	100	✓

Jobseeker Technical Details

Column Name	Data Type	Length	Allow Nulls
JobseekerID	varchar	50	
Skill1	varchar	20	
Skill1Year	int	4	
Skill1LastUsed	int	4	
Skill2	varchar	20	✓
Skill2Year	int	4	✓
Skill2LastUsed	int	4	✓
Skill3	varchar	20	✓
Skill3Year	int	4	✓
Skill3LastUsed	int	4	✓
Skill4	varchar	20	✓
Skill4Year	int	4	✓
Skill4LastUsed	int	4	✓

6.2 Functional Area/Design Unit B

6.2.1 Functional Overview

6.2.2 Feasibility Report

Preliminary investigation examine project feasibility, the likelihood the system will be useful to the organization. The main objective of the feasibility study is to test the Technical, Operational and Economical feasibility for adding new modules and debugging old running system. All system is feasible if they are unlimited resources and infinite time. There are aspects in the feasibility study portion of the preliminary investigation:

- Technical Feasibility
- Operation Feasibility
- Economical Feasibility

Technical Feasibility

The technical issue usually raised during the feasibility stage of the investigation includes the following:

- Does the necessary technology exist to do what is suggested?
- Do the proposed equipments have the technical capacity to hold the data required to use the new system?
- Will the proposed system provide adequate response to inquiries, regardless of the number or location of users?
- Can the system be upgraded if developed?

Are there technical guarantees of accuracy, reliability, ease of access and data security?

Earlier no system existed to cater to the needs of 'Secure Infrastructure Implementation System'. The current system developed is technically feasible. It is a web based user interface for audit workflow at NIC-CSD. Thus it provides an easy access to the users. The database's purpose is to create, establish and maintain a workflow among various entities in order to facilitate all concerned users in their various capacities or roles. Permission to the users would be granted based on the roles specified. Therefore, it provides the technical guarantee of accuracy, reliability and security. The software and hard requirements for the development of this project are not many and are already available in-house at NIC or are available as free as open source. The work for the project is done with the

current equipment and existing software technology. Necessary bandwidth exists for providing a fast feedback to the users irrespective of the number of users using the system.

Operational Feasibility

Proposed projects are beneficial only if they can be turned out into information system. That will meet the organization's operating requirements. Operational feasibility aspects of the project are to be taken as an important part of the project implementation. Some of the important issues raised are to test the operational feasibility of a project includes the following: -

- Is there sufficient support for the management from the users?
- Will the system be used and work properly if it is being developed and implemented?
- Will there be any resistance from the user that will undermine the possible application benefits?

This system is targeted to be in accordance with the above-mentioned issues. Beforehand, the management issues and user requirements have been taken into consideration. So there is no question of resistance from the users that can undermine the possible application benefits.

The well-planned design would ensure the optimal utilization of the computer resources and would help in the improvement of performance status.

Economic Feasibility

A system can be developed technically and that will be used if installed must still be a good investment for the organization. In the economical feasibility, the development cost in creating the system is evaluated against the ultimate benefit derived from the new systems. Financial benefits must equal or exceed the costs.

The system is economically feasible. It does not require any addition hardware or software. Since the interface for this system is developed using the existing resources and technologies available at NIC, There is nominal expenditure and economical feasibility for certain.

6.2.3 System Testing and Implementation

Introduction

Software testing is a critical element of software quality assurance and represents the ultimate review of specification, design and coding. In fact, testing is the one step in the software engineering process that could be viewed as destructive rather than constructive.

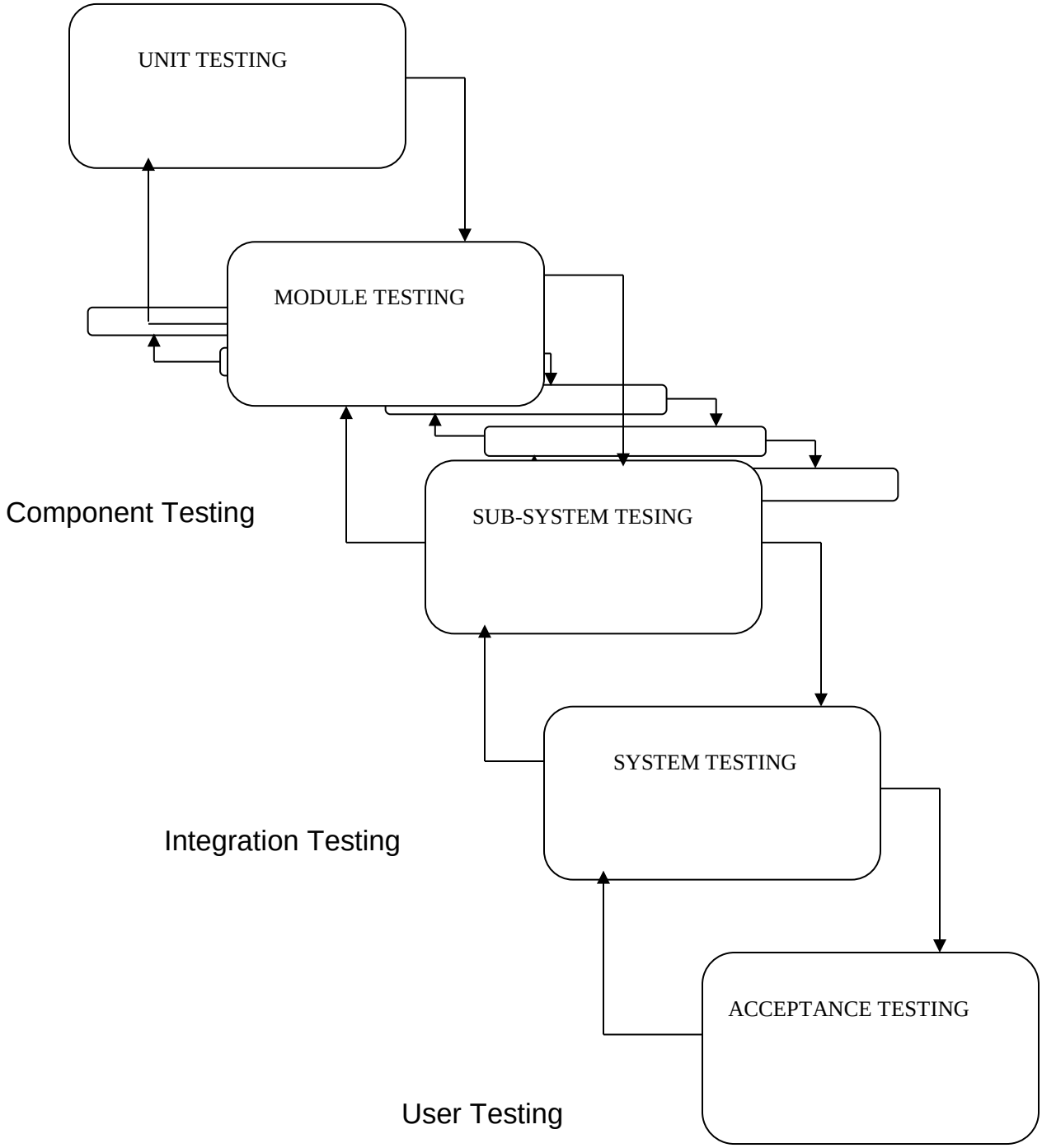
A strategy for software testing integrates software test case design methods into a well-planned series of steps that result in the successful construction of software. Testing is the set of activities that can be planned in advance and conducted systematically. The underlying motivation of program testing is to affirm software quality with methods that can economically and effectively apply to both strategic to both large and small-scale systems.

Strategic approach to software testing

The software engineering process can be viewed as a spiral. Initially system engineering defines the role of software and leads to software requirement analysis where the information domain, functions, behavior, performance, constraints and validation criteria for software are established. Moving inward along the spiral, we come to design and finally to coding. To develop computer software we spiral in along streamlines that decrease the level of abstraction on each turn.

A strategy for software testing may also be viewed in the context of the spiral. Unit testing begins at the vertex of the spiral and concentrates on each unit of the software as implemented in source code. Testing progress by moving outward along the spiral to integration testing, where the focus is on the design and the construction of the software architecture. Talking another turn on outward on the spiral we encounter validation testing where requirements established as part of software

requirements analysis are validated against the software that has been constructed. Finally we arrive at system testing, where the software and other system elements are tested as a whole.



Unit Testing

Unit testing focuses verification effort on the smallest unit of software design, the module. The unit testing we have is white box oriented and some modules the steps are conducted in parallel.

White Box Testing

This type of testing ensures that

- All independent paths have been exercised at least once
- All logical decisions have been exercised on their true and false sides

- All loops are executed at their boundaries and within their operational bounds
- All internal data structures have been exercised to assure their validity.

To follow the concept of white box testing we have tested each form .we have created independently to verify that Data flow is correct, All conditions are exercised to check their validity, All loops are executed on their boundaries.

Basic path testing:

Established technique of flow graph with Cyclomatic complexity was used to derive test cases for all the functions. The main steps in deriving test cases were:

Use the design of the code and draw correspondent flow graph.

Determine the Cyclomatic complexity of resultant flow graph, using formula:

V(G)=E-N+2 or

V(G)=P+1 or

V(G)=Number Of Regions

Where V(G) is Cyclomatic complexity,

E is the number of edges,

N is the number of flow graph nodes,

P is the number of predicate nodes.

Determine the basis of set of linearly independent paths.

Conditional testing

In this part of the testing each of the conditions were tested to both true and false aspects. And all the resulting paths were tested. So that each path that may be generate on particular condition is traced to uncover any possible errors.

Data flow testing

This type of testing selects the path of the program according to the location of definition and use of variables. This kind of testing was used only when some local variable were declared. The *definition-use chain* method was used in this type of testing. These were particularly useful in nested statements.

Loop testing

In this type of testing all the loops are tested to all the limits possible. The following exercise was adopted for all loops:

- All the loops were tested at their limits, just above them and just below them.
- All the loops were skipped at least once.
- For nested loops test the inner most loop first and then work outwards.
- For concatenated loops the values of dependent loops were set with the help of connected loop.
- Unstructured loops were resolved into nested loops or concatenated loops and tested as above.

Each unit has been separately tested by the development team itself and all the input have been validated.

7 Open Issues

This Project does not have any open issues.

8 Acknowledgements

We drive our great pleasure in expressing our sincere gratitude to advisor Dr.Soon Oak Park and Professor N.CHEN for their timely suggestions, which helped us to complete the project work successfully.

We also very much thankful to the members of the Computer Science Department and also the members of our team for their extreme support and guidance in making this project successful.

9 References

- **For .NET installation**
  www.support.mircosoft.com
- **For Deployment and packing on server**
  www.developer.com
  www.15seconds.com
- **For Sql**
  www.msdn.microsoft.com
- **For Asp.net**
  www.msdn.microsoft.com/net/quickstart/aspplus/default.com
  www.asp.net
  www.fmexpense.com/quickstart/aspplus/default.com
  www.asptoday.com
  www.aspfree.com
  www.4guysfromrolla.com/index.aspx

10 Appendices

NA