# Vidareutveckling av projekthanteringssystem

Jonatan Nygård

Examensarbete för ingenjör (YH)-examen

Utbildningsprogrammet för informationsteknik

Vasa 2015

EXAMENSARBETE


Författare: Jonatan Nygård

Utbildningsprogram och ort: Informationsteknik, Vasa

Handledare: Susanne Österholm


Titel: *Vidareutveckling av projekthanteringssystem*

---


Datum 28.04.2015 Sidantal 33


---


## Abstrakt

Syftet med detta examensarbete var att vidareutveckla Indevit Solutions egna webbaserade projekthanteringssystem. Webbapplikationen skulle utvecklas till att ha en responsiv webbdesign med asynkron funktionalitet, samt till att förenkla rapporteringen inom byggnadsbranschen. Byggnadsbranschens rapportskyldighet till skatteverket fastslogs 01.07.2014, och skall göras för alla byggnads- och renoveringsprojekt vars värde överstiger 15 000 € eller som behöver bygglov.

Alla som har behörighet till ett projekt, skall med webbapplikationen kunna lägga till sina egna arbetsprestationer. Baserat på dessa prestationer skall administratorn eller projektansvarig kunna generera en rapport för skatteverket.

Webbapplikationen konstruerades med hjälp av teknikerna ASP.NET MVC 5 och SQL. Användargränssnittet utvecklades med hjälp av ramverket Bootstrap, samt ASP.NET MVC extensionen DevExpress.

Examensarbetet blev en responsiv webbapplikation med asynkron funktionalitet och med de grundfunktioner som behövs för att administrera ett projekt. En rapport på de användare som har arbetat med projektet under en specifik månad går att generera.


---


Språk: svenska Nyckelord: *byggnadsbranschen,* responsiv webbdesign, ASP.NET MVC


---

BACHELOR'S THESIS


| | |
|---|---|
| Author: | Jonatan Nygård |
| Degree programme and location: | Information Technology, Vaasa |
| Supervisor: | Susanne Österholm |


Title: *Further development of a project management system*

---


Date 28.04.2015 Number of pages 33


---


## Abstract

The purpose of this thesis was to further develop Indevit Solutions' own web-based project management system. The web application was to be developed to have a responsive web design with asynchronous functionality, and it should simplify reporting in construction industry. The construction industry's duty to report to the tax office was established on 01.07.2014 and reports must be made for all building and renovation projects whose value exceeds € 15,000, or which need a building permit.

All who have access to a project shall, with the web application, be able to add their own work performance. Based on these achievements, the administrator or project manager can generate a report for submission to the tax office.

The web application was constructed using the techniques ASP.NET MVC 5 and SQL. The user interface was developed using the Bootstrap framework and the DevExpress ASP.NET MVC extension.

The work became a responsive web application with asynchronous functionality and with the basic functions needed to manage a project. A report on those users who have worked on the project during a specific month can be generated.


---


Language: Swedish Keywords: construction industry, responsive web design, ASP .NET MVC


---

# Innehållsförteckning

Förklarningar

| | |
|---|---|
| Ajax | Ajax är en samling av olika webbteknologier som används t.ex. för att skicka eller hämta information från en server, utan att störa användargränssnittet. |
| ASP | Förkortning för Active Server Pages, är en teknik utvecklad av Microsoft för att skapa dynamiska webbserverapplikationer. |
| Asynkron | Uppdatera informationen utan att störa användargränssnittet. |
| Bootstrap | Bootstrap är ett ramverk för webbutveckling |
| DevExpress | DevExpress är en tredjepartskomponent för att skapa användargränssnitt. |
| JQuery | JQuery är ett JavaScript-bibliotek som förenklar användningen av JavaScript. |
| MVC | Förkortning för Model-View-Controller, ett designmönster inom systemutveckling. |
| .NET | .NET Framework är en systemkomponent i Microsoft Windows. Den består av ett stort klassbibliotek och ger driftskompatibilitet över flera programmeringsspråk. |
| Ramverk | Ramverk är samling av t.ex. klasser som underlättar för utvecklare. |
| Responsiv design | Responsiv webbdesign innebär bl.a. att designen förändras beroende på vilken skärmstorlek användaren har. |
| SQL | Structured Query Language är ett frågespråk för relationsdatabaser. |

# 1 Inledning

Arbetet gick ut på att vidareutveckla ett webbaserat projekthanteringssystem till att ha en responsiv webbdesign och en asynkron funktionalitet. Systemet skulle även utvidgas till en produkt för byggnadsbranschen.

## 1.1 Uppdragsgivare

Indevit Solutions är ett privat aktiebolag som grundades 2011 av Peter Hertsbacka och Christoffer Smeds. Företaget är beläget i Vasa och har två heltidsanställda. Indevit Solutions utvecklar både tekniska lösningar och kundens arbetsprocesser, där huvudmålet är att den tekniska lösningen skall anpassas efter den optimala arbetsprocessen. Utvecklingsarbetet sker genom ett tätt samarbete med nyckelpersoner hos kunderna, som oftast finns inom produktionsindustrin. De tekniska lösningarna är oftast skräddarsydda webb- och skrivbordslösningar, men mobila lösningar är också en viktig del. Indevit Solutions utvecklar även kundanpassade programlösningar för mindre kunder.

Ett exempel på ett av företagets senaste större projekt var att utveckla arbetsprocesserna och programlösningar för ett nytt lager i USA. Kunden, ett större finländskt företag med dotterbolag i USA, ville utlokalisera sitt lager till annan ort. All lagerhantering för både inkommande och utgående varor, samt lagerhållningen behövde utvecklas. Flytten av deras nuvarande lager behövdes optimeras så att deras kunder hela tiden kunde förlita sig på att få varor från endera lagren. Även integrationer till fraktbolag blev implementerade. [9]

För att hantera alla olika projekt, har Indevit Solutions utvecklat ett webbaserat projekthanteringssystem, som möjliggör för kunden att följa med projektets utveckling i realtid. Detta projekthanteringssystem är baserat på Scrum-metodiken.

## 1.2 Bakgrund

Projekthanteringssystemet är ämnat för projekt inom programutveckling och följer Scrum-metodiken, som används bland annat inom programutveckling. Projekthanteringssystemet är utvecklat med ASP.NET MVC3 samt med extensionen DevExpress ASP.NET MVC.

Systemet är uppbyggt så att funktioner för att skapa, editera och radera finns på separata sidor.

För att editera grunduppgifter för ett specifikt projekt, skickas användaren till en sida med alla pågående projekt. När man har valt vilket projekt man vill editera skickas användaren på nytt till en ny sida där man gör sina ändringar.

Projekthanteringssystemet delar in projektets arbetsuppgifter och resurser i kategorier. Till dessa arbetsuppgifter kan användaren lägga till arbetsprestationer med en kort beskrivning över vad som gjorts.

Åtkomsten till olika sidor såsom editeringssidor, är konstruerad med funktionsrättigheter. För att en användare ska kunna ta sig till en editeringssida innebär det att användaren behöver rättigheter till den specifika sidan. Även tillgång till systemets funktioner är konstruerad på likadant sätt.

Användargränssnittet i systemet använder sig inte av en responsiv webbdesign, vilket innebär att sidorna inte anpassar sig till den enhet man använder sig av.

# 2 Uppgift

Uppdraget som beställdes av Indevit Solutions var att vidareutveckla det webbaserade projekthanteringssystemet. Kraven för vidareutvecklingen var att webbapplikationen skulle följa en responsiv webbdesign med hjälp av ramverket Bootstrap som finns inkluderat i ASP.NET MVC5. Webbapplikationens användargränssnitt skulle även visas asynkront, vilket innebär att man skickar och hämtar informationen i bakgrunden, utan att avbryta den sida som användaren ser. Det andra kravet var att utvidga systemet till en produkt för byggnadsbranschen för att underlätta de nya rapporteringsskyldigheterna.

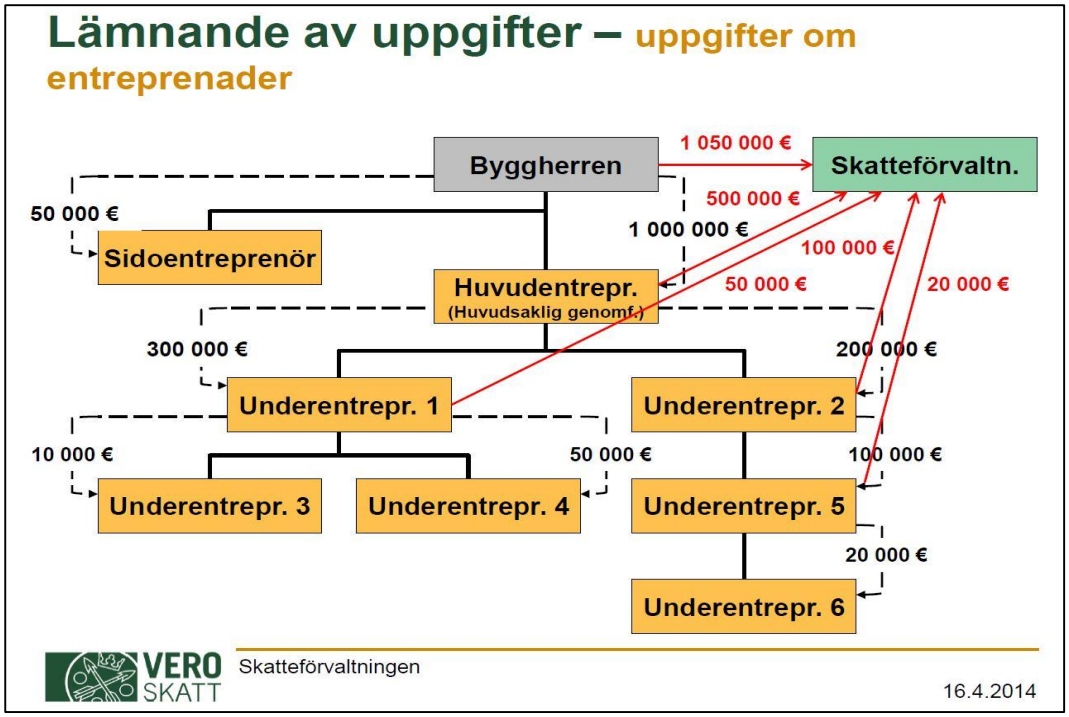
Logiken, för själva projekthanteringen, skulle följa det ursprungliga systemet med att dela in projektets innehåll i kategorier och uppgifter. Databasschemat skulle följa den ursprungliga databasen med vissa modifieringar.

## 2.1 Anmälningar om byggande

Skyldighet att anmäla byggnads- och renoveringsprojekt till skattemyndigheten trädde i kraft 01.07.2014, i syfte att minska ”grå ekonomi”, som finns inom byggnadsbranschen.

Företag och egenföretagare, som beställer byggentreprenader eller byggarbetskraft, är skyldiga att lämna uppgifter till skattemyndigheten. Anmälningar behöver endast göras om projektets värde överstiger 15 000 euro eller om det krävs ett bygglov.

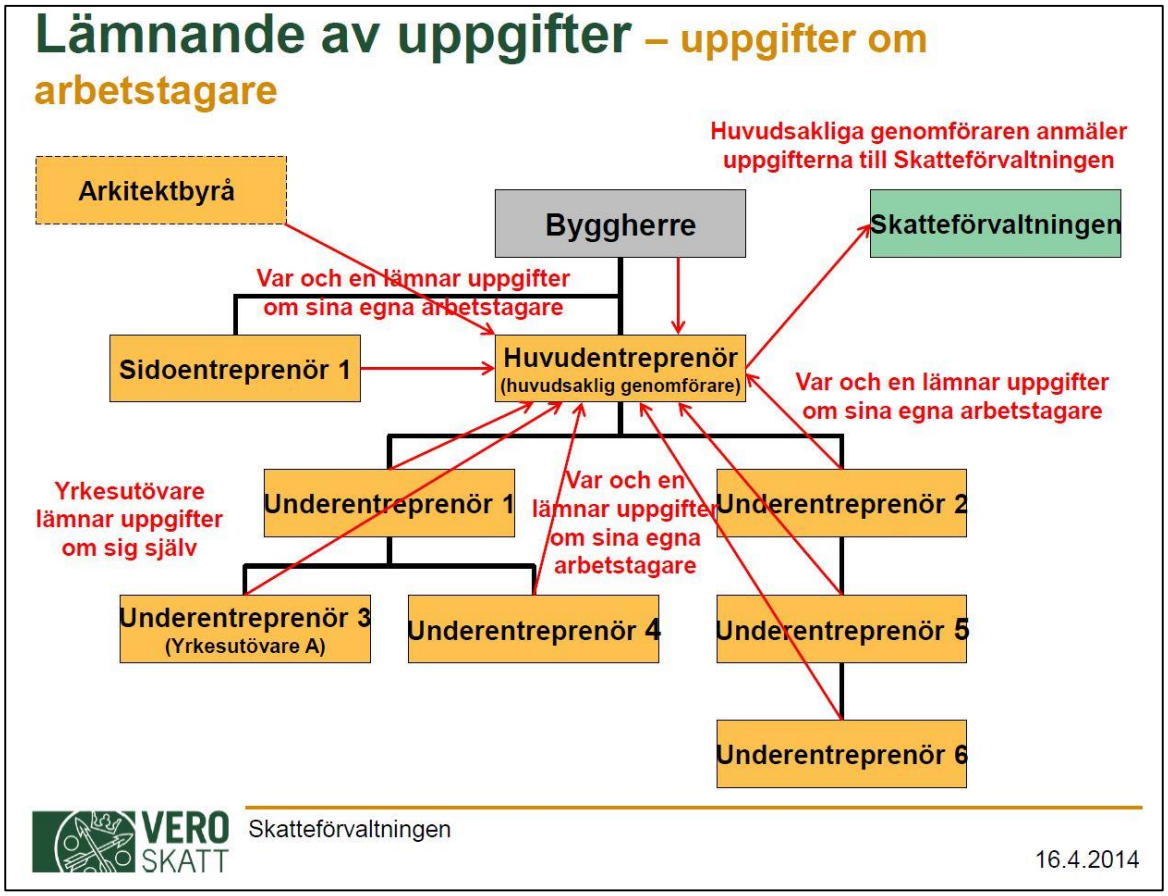
Hushållen, som beställer ett byggnads- eller renoveringsprojekt, behöver endast lämna uppgifter om entreprenad före slutsynen ifall arbetet kräver bygglov. När byggnadsinspektören gör slutsynen uppvisas ett intyg från skatteförvaltningen att skyldigheten att lämna uppgifter har fullgjorts av beställaren [2]. Företag eller egenföretagare skall anmäla varje månad om varje byggnadsentreprenad skilt för sig.




*Figur 1. Lämnande av uppgifter om entreprenader. [1]*

Enligt figur 1, skall beställaren av entreprenaden lämna uppgifter till skatteförvaltningen om de avtal som beställts sammanlagt. Exempel: Byggherren beställer en entreprenad av en firma för 1 miljon euro och en annan entreprenad av en annan firma. Byggherren är då skyldig att lämna uppgifter om dessa två entreprenader till skatteförvaltningen. Huvudentreprenören gör ett antal beställningar från olika underentreprenörer och är då skyldig att lämna uppgifter om dessa entreprenader till skatteförvaltningen. Sidoentreprenören utför hela sitt arbete själv och gör inga beställningar från någon underentreprenör och behöver därför inte lämna några uppgifter om entreprenaden.

Om det uppstår en gemensam byggarbetsplats, måste den huvudsakliga genomföraren eller byggnadsherren månatligen rapportera om arbetstagarna, alla som befinner sig på byggarbetsplatsen samt de personer som har arbetat med projektet på annat håll, t.ex. planerare och arkitekter.




*Figur 2. Lämnande av uppgifter om arbetstagare. [1]*

Se figur 2. På en gemensam byggarbetsplats är det den huvudsakliga genomföraren som är skyldig att lämna uppgifter om arbetstagarna som befinner sig på arbetsplatsen. Om ingen huvudsaklig genomförare har bestämts är det byggherren som är skyldig att lämna uppgifterna till skatteförvaltningen. Alla entreprenörer som ingår i projektet skall lämna uppgifter om sina arbetstagare till den huvudsakliga genomföraren.

Även sidoentreprenörer, som inte har kontakt med den huvudsakliga genomföraren, skall lämna in uppgifter om sina arbetstagare. Enskilda yrkesutövare lämnar in uppgifter om sig själv. [1]

Alla dessa rapporter skall lämnas in till skattemyndigheten senast den femte följande månad i elektronisk form. För att lämna in rapporterna kan man använda sig av en webblankett som finns på webbplatsen suomi.fi.

Programvara som skapar rapporter måste omvandla rapporterna till textfiler. Informationen skall bestå av kod: uppgiftspar vilka finns definierade på webbplatsen ilmoitin.fi.
Kod: uppgiftspar innebär att alla uppgifter som skall rapporteras har en motsvarande kod.

Exempel på hur kod: uppgiftsparen ser ut.

- 010: Identifikationsuppgift för den uppgiftsskyldige
- 200: Kontaktpersonens efternamn
- 201: Kontaktpersonens förnamn
- 202: Telefonnummer för uppgiftsskyldige
- 250: Byggarbetsplatsens uppgifter
- 251: Byggarbetsplatsens nummer
- 500: Arbetsgivarens uppgifter
- 501: Arbetsgivarens FO-nummer
- 550: Efternamn för arbetsgivarens representant
- 551: Förnamn för arbetsgivarens representant
- 552: Telefonnummer för arbetsgivarens representant

De omvandlade rapporterna skall laddas upp till webbtjänsten ilmoitin.fi. [8]

## 2.2 Det ursprungliga systemets funktionalitet

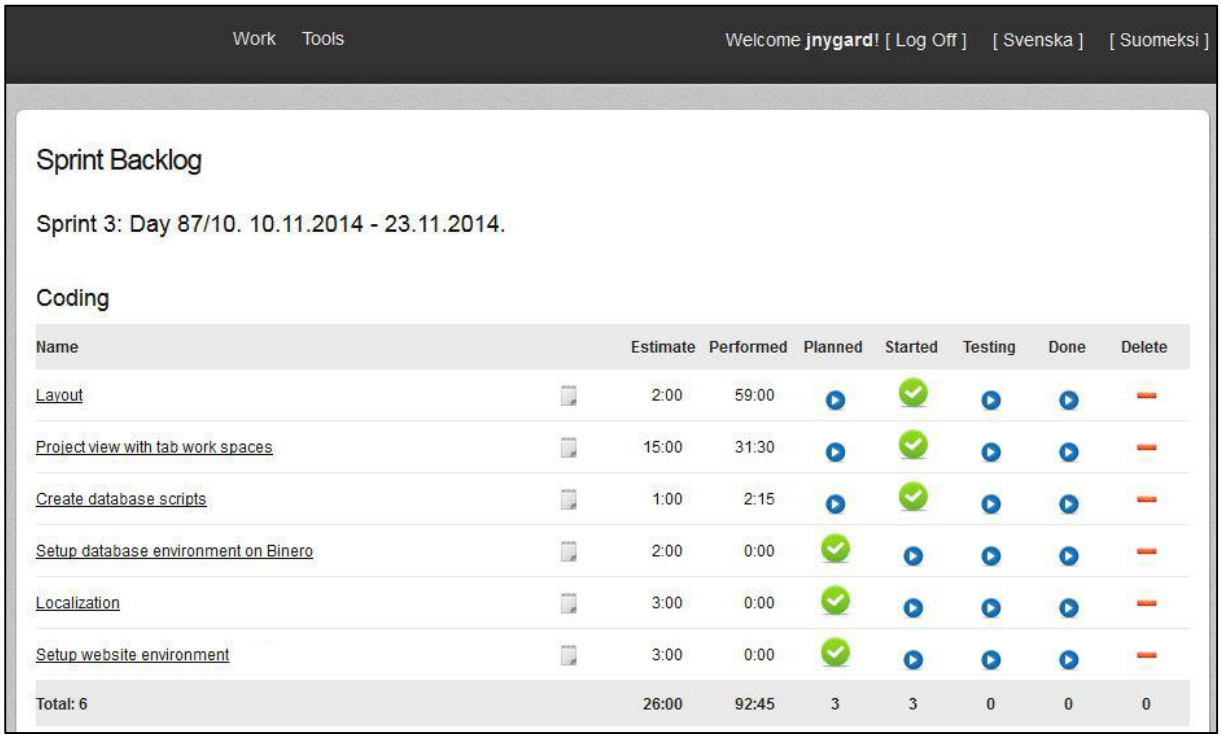
Innan ett projekt skapas måste kunden registreras, eftersom ett projekt inte kan existera om det inte är kopplat till en kund.

Projektet skapas av projektledaren eller administratorn, så att grunduppgifter inmatas och en kund kopplas till projektet från kundregistret. Grunduppgifterna för projektet består av namnet på projektet, om man skall kunna fakturera projektet, samt om projektet skall följa Scrum-metodiken. Följer projektet Scrum-metodiken skall även sprintlängden definieras.

Syftet med inställningen för Scrum-metodiken är om det skall följas eller inte. Att Scrum-metodiken är aktiverad för ett projekt innebär att man måste skapa en sprint och lägga till vilka uppgifter som skall finnas med i sprinten. Endast de uppgifter som finns med i en sprint kan hanteras. Om Scrum-metodiken inte är aktiverad innebär det att alla kategorier och uppgifter är aktiva och kan hanteras.

Efter att projektets grunduppgifter har sparats, skapar man kategorier med ett namn och om det är en återkommande kategori. Till de olika kategorierna lägger man till uppgifter. Uppgifterna skapas med namn, beskrivning, deadline, vem som är ansvarig utvecklare, vem som är ansvarig testare, uppskattad tidsåtgång, vem som begärt arbetsuppgiften och vem som har godkänt den.

Om projektet har Scrum-metodiken aktiverad, skapar man en sprint med start och slutdatum för sprinten. Till sprinten lägger man till vilka uppgifter som skall finnas med i sprinten. Om man startar en ny sprint, utan att uppgifterna från föregående sprint är avslutade, flyttas de över till den nya sprinten automatiskt. Hanteringen av en pågående sprint involverar tillägg av arbetsprestationer, tillägg av kommentarer till en arbetsuppgift och statushantering av uppgifter såsom startad, testning och klar. I figur 3 visas en sprint som är aktiv.




*Figur 3. Arbetsuppgifter i en sprint.*

I ett projekt, som inte har Scrum-metodiken aktiverad, finns inte sprint-funktionen tillgänglig och alla kategorier och uppgifter går att hantera. Statushanteringen finns inte tillgänglig i ett projekt där Scrum-metodiken inte är aktiverad.

Nästa steg är att tilldela funktionsrättigheter till användarna, som skall ha tillgång till projektet. Alla funktioner i applikationen kräver att användaren har rättigheterna till den specifika funktionen.

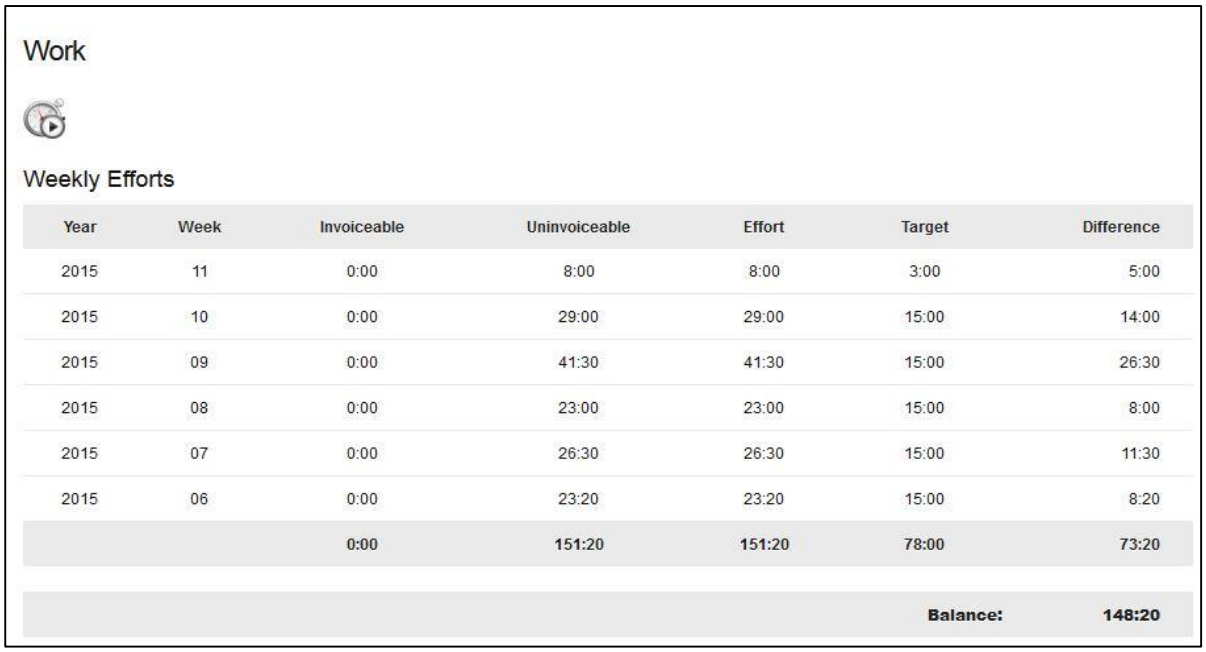
En användare, som inte skall kunna administrera projektet utan endast kunna se projektet i sin helhet, och vilka olika uppgifter som finns i projektet, får då rättigheterna till dessa funktioner. Projektledaren eller administratören som skapar projektet får automatiskt alla rättigheter till projektet som skapades.

Användare, som arbetar med projektet har som huvuduppgift inom systemet att lägga till sina egna arbetsprestationer, som utförts under dagen. Arbetsprestationerna dokumenteras per uppgift med en kort beskrivning och antalet timmar.

För att dokumentera arbetsprestationer som utförts, måste en uppgift väljas från en lista med alla projekt och pågående uppgifter per projekt. När man valt uppgift, skriver man in sina utförda arbetstimmar och en beskrivning. All utförda arbetsprestationer grupperas efter datum och visas i en tabell. För att editera en dokumenterad arbetsprestation, väljer man en prestation från tabellen och skickas vidare till en editeringssida för den valda arbetsprestationen.

Projektledaren använder sig av arbetsprestationerna för att se arbetsmängd per användare och för att fakturera kunden. Arbetsprestationerna jämförs även mot den uppskattade arbetsprestationen per uppgift.

Användaren har också nytta av de dokumenterade arbetsprestationerna för att se sina egna arbetstimmar. Med den informationen kan den totala arbetsmängden jämföras med den arbetsprestation som användaren skall utföra under en period. Vanligtvis definieras en period som timmar per arbetsvecka. Med denna jämförelse kan användaren enkelt se över sina arbetstimmar. Se figur 4.



Work

Weekly Efforts

| Year | Week | Invoiceable | Uninvoiceable | Effort | Target | Difference |
|---|---|---|---|---|---|---|
| 2015 | 11 | 0:00 | 8:00 | 8:00 | 3:00 | 5:00 |
| 2015 | 10 | 0:00 | 29:00 | 29:00 | 15:00 | 14:00 |
| 2015 | 09 | 0:00 | 41:30 | 41:30 | 15:00 | 26:30 |
| 2015 | 08 | 0:00 | 23:00 | 23:00 | 15:00 | 8:00 |
| 2015 | 07 | 0:00 | 26:30 | 26:30 | 15:00 | 11:30 |
| 2015 | 06 | 0:00 | 23:20 | 23:20 | 15:00 | 8:20 |
| | | **0:00** | **151:20** | **151:20** | **78:00** | 73:20 |
| | | | | | **Balance:** | **148:20** |

*Figur 4. Arbetsprestationslista för en användare.*

# 3 Tekniker

Webbapplikationen skulle utvecklas med hjälp av ASP.NET MVC 5 Framework, och utvecklingsverktyget Microsoft Visual Studio 2013. Programmeringsspråket som användes var C Sharp för systemets logik och för att behandla och överföra informationen mellan användaren och databasen. Gränssnittet skapades med Html, CSS, Bootstrap Framework och extensionen DevExpress ASP.NET MVC.

## 3.1 ASP.NET MVC 5

ASP.NET MVC Framework är en teknik som används för att skapa dynamiska webbsidor, med designmönstret Model View Controller.

Tekniken bygger på Microsofts ASP.NET som är en del av Microsofts .NET ramverk. Microsoft .NET-ramverket är en samling av klasser som en utvecklare kan använda sig av för att få tillgång till bl.a. databaser och webbtjänster. [13]

Designmönstret Model View Controller är ett designmönster som delar upp programmet i tre separata delar.

Model är datalagret där all data samt datalogiken finns. Till exempel när man vill visa information om en person på en webbsida, hämtar man informationen från en databas, lagrar information som ett modellobjekt och skickar modellen till sidan. Modellen är inte låst till databasen, utan kan innehålla flera eller färre egenskaper än vad som finns i databastabellen.

När man skall editera modellens information med ett HTML-formulär kan man använda sig av ASP .NET MVC Model Binding. Model Binding överför data automatiskt från formuläret till en fördefinierad modell oberoende om en POST- eller GET-metod används. När man tar emot modellen i controllern, finns informationen från formuläret i modellen.

View är den del som handhar användargränssnittet och kopplar ihop Model med controllern. I View-delen finns allt som har med webb-gränssnittet att göra. View-delen kan skapas med hjälp av HTML, CSS och JavaScript. En webbsida skall helst bara innehålla visuella delar och komponenter och undvika avancerad logik och beräkningar.

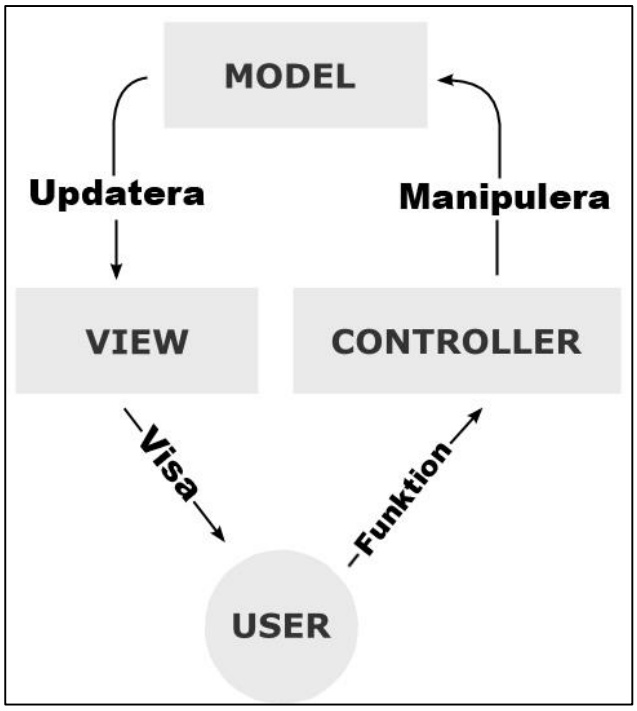
Att göra tunga beräkningar på en sida gör mer skada än nytta på grund av att allt detta sker på användarens dator och inte på en server, som oftast har högre kapacitet jämfört med en vanlig användares dator.

Controller är den del som tar hand om logiken och alla beräkningar, t.ex. för att hämta data och göra eventuella beräkningar och sedan skicka vidare informationen till en sida där användaren får se det manipulerade innehållet.

När man skall spara ändrade uppgifter till en databas, skickas de ändrade uppgifterna i en Model från View till Controller där lagringen till databasen sker.

När webbapplikationen gör en förfrågning till controllern, startas applikationens controllerdel på en server. Tack vare att controllern körs på en server kan man använda sig av programmeringsspråk som C Sharp, som också används inom .NET ramverket. [3, 15]

Figur 5 visar i vilken ordning informationen går i designmönstret Model View Controller.




*Figur 5. MVC-schema. [12]*

Med MVC kan man enkelt ändra hela programlogiken utan att behöva tänka på gränssnittet eller på den lagrade informationen. När logiken på webbsida skall uppdateras behöver man endast ändra på controllern. Uppdateringen av gränssnittet fungerar på likadant sätt, man behöver endast göra ändringar i de filer som berör gränssnittet.

Om man följer MVC-tekniken skall data lagras i en modell och alla tunga beräkningar och komplicerad logik sker i controllern, men om detta inte är en möjlighet, finns det andra metoder att använda sig av.

I MVC-tekniken finns det möjlighet att skicka data från en controller till en sida med hjälp av tre olika alternativ, vilka är ViewBag, ViewData och TempData. Dessa alternativ kan användas om användning av en modell inte är möjlig, eller om man endast vill visa en engångstext till användaren. Dessa alternativ är väldigt kortlivade och ”garbage collection” körs automatiskt då begäran har slutförs.

ViewBag är en dynamisk egenskap som använder sig av dynamiska funktioner från C Sharp version 4. Användningen av en ViewBag är endast tillgänglig mellan den aktuella begäran från en Controller till en View, vilket betyder att när den aktuella begäran är slutförd frigörs ViewBag. [23]

ViewData är en samling av objekt som lagras med hjälp av en teknik som baserar sig på nycklar och nyckelvärde. Användningen av ViewData är likadan som för en ViewBag, endast tillgänglig mellan den aktuella begäran från en Controller till en View. Precis som för ViewBag frigörs ViewData när den aktuella begäran är slutförd. [24]

TempData är ett liknande objekt som ViewData, men som även finns tillgänglig i den efterföljande begäran, som till exempel mellan två controllers. Detta fungerar för att TempData lagrar innehållet i sessionstillstånd. När den efterföljande begäran är slutförd frigörs TempData. [18]

Speciellt för MVC 5 jämfört med MVC 4 är att Bootstrap3 front-end framework är inkluderat i ett MVC5-projekt. Bootstrap front-end framework är ett ramverk för snabbare och enklare webbutveckling. Ramverket innehåller webbaserade metoder och stilmallar. [3, 4]

## 3.2 Structured Query Language

Structured Query Language (SQL) är ett frågespråk som kan användas för att t.ex. uppdatera eller hämta data från en databas. Exempel på ett vanligt kommando i SQL är select. Kommandot select används för att hämta information ur databasen. År 1987 antog Internationella standardiseringsorganisationen SQL som en standard. I dagsläget är SQL det dominerande frågespråket inom databashantering.

En typisk fråga till en databas kan vara ”select * from Customers”. Denna fråga betyder: välj alla kolumner ur tabellen Customers. Se figur 6. [17]

```
SELECT * FROM RV_Customers
```

| CustomerId | CompanyId | CustomerFirstName | CustomerLastName | CustomerAddress1 |
|---|---|---|---|---|
| 56DDFC7C-... | B18B564B... | Oskar | Svensson | Dikes Esplanaden 23 B 2 |
| 7EC3D6F9-... | B18B564B... | Alfred | Pettersson | Torparvägen 1 |
| D02E8DBE... | B18B564B... | Pelle | Andersson | Bilvägen 23 |

*Figur 6. SQL-förfrågning.*

## 3.3 Language-Integrated Query

Language-Integrated Query (LINQ) är en komponent i Microsoft .NET Framework som ger frågefunktion för .NET-språk. LINQ syntaxen är mycket lik syntaxen som används i SQL. Man använder sig av LINQ för att hämta eller manipulera dataobjekt. Dataobjekten kan finnas i listor, XML-dokument, klasser eller objekt från relationsdatabaser. I figur 7 visas ett exempel på ett LINQ-uttryck för att hämta information från ett objekt. [11]

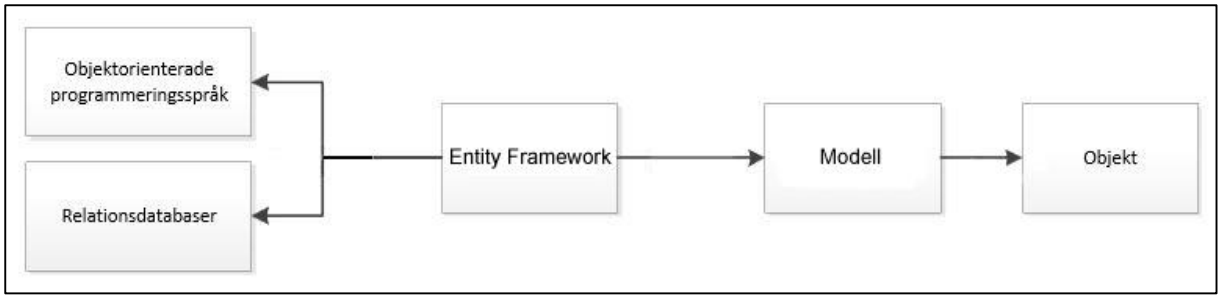
```
var UsersAccessToProject = (from user in db.RV_User
                            join projectAccess  in db.RV_ProjectAccess on user.UserId equals projectAccess.UserId
                            where projectAccess.ProjectId == model.ProjectId
                            select user).ToList();
```

*Figur 7. Exempel på LINQ syntax.*

## 3.4 Entity Framework

Entity Framework är ett ORM-ramverk ”Object-relational mapping”, som utvecklades av Microsoft för .NET Framework. ORM-ramverket möjliggör en koppling mellan t.ex. databaser och objektorienterade programspråk. Med ORM skapas ett virtuellt dataobjekt som kan användas inifrån programmeringsspråket för att manipulera data.

Entity Framework skapar en modell utgående från en databas, och modellen används då som ett virtuellt dataobjekt i programmeringsspråket.




*Figur 8. Entity Framework exempel. [21]*

Entity Framework kopplas mot en databas och hämtar databasens tabeller och kolumner. Databasen som hämtades används för att skapa en modell. Modellen som Entity Framework skapade är ett objekt med en samling av klasser och egenskaper som motsvarar relationsdatabasens tabeller och kolumner. Se figur 8.

Entity Framework har även en omvänd funktion kallad code-first, vilket innebär att man skapar alla sina klasser och baserat på dessa klasser genererar Entity Framework en databas. [7]

Tack vare Entity Framework sparar man mycket arbete. Utan Entity Framework skulle man behöva skriva hela SQL-förfrågningen manuellt, ta hand om felhanteringen och stega igenom hela SQL förfrågningens svar för att få tillgång till informationen. I figur 9 och 10 visas ett exempel på kod där Entity Framework används respektive inte används. [14]

Det finns två klasser i exemplet, en klass med instruktörer och en klass med kurser. I klassen instruktörer finns det en lista där kurserna för instruktören skall lagras. [14]

I figur 9 används Entity Framework.

```
public IEnumerable<Instructor> GetInstructors()
{
    List<Instructor> instructors;
    using (var db = new SchoolContext())
    {
        instructors = db.Instructors.Include("Courses").ToList();
    }
    return instructors;
}
```

*Figur 9. Med hjälp av Entity Framework exempel. [14]*

I figur 10 får man samma resultat som i figur 9 men med mycket mer kod, på grund av att man inte använder Enity Framework.

```
public IEnumerable<Instructor> GetInstructorsADONET()
{
    var connString = ConfigurationManager.ConnectionStrings["SchoolContext"].ConnectionString;
    List<Instructor> instructors = new List<Instructor>();
    using (var conn = new SqlConnection(connString))
    {
        conn.Open();

        var cmd = new SqlCommand("SELECT Person.PersonID, Person.LastName, Person.FirstName, Course.Title, Course.Credits " +
            "FROM Course INNER JOIN " +
                  "CourseInstructor ON Course.CourseID = CourseInstructor.CourseID RIGHT OUTER JOIN " +
                  "Person ON CourseInstructor.PersonID = Person.PersonID " +
                  "WHERE (Person.Discriminator = 'Instructor') " +
                  "ORDER BY Person.PersonID", conn);
        var reader = cmd.ExecuteReader();

        int currentPersonID = 0;
        Instructor currentInstructor = null;
        while (reader.Read())
        {
            var personID = Convert.ToInt32(reader["PersonID"]);
            if (personID != currentPersonID)
            {
                currentPersonID = personID;
                if (currentInstructor != null)
                {
                    instructors.Add(currentInstructor);
                }
                currentInstructor = new Instructor();
                currentInstructor.LastName = reader["LastName"].ToString();
                currentInstructor.FirstMidName = reader["FirstName"].ToString();
            }
            if (reader["Title"] != DBNull.Value)
            {
                var course = new Course();
                course.Title = reader["Title"].ToString();
                course.Credits = Convert.ToInt32(reader["Credits"]);
                currentInstructor.Courses.Add(course);
            }
        }
        if (currentInstructor != null)
        {
            instructors.Add(currentInstructor);
        }

        reader.Close();
        cmd.Dispose();
    }
    return instructors;
}
```

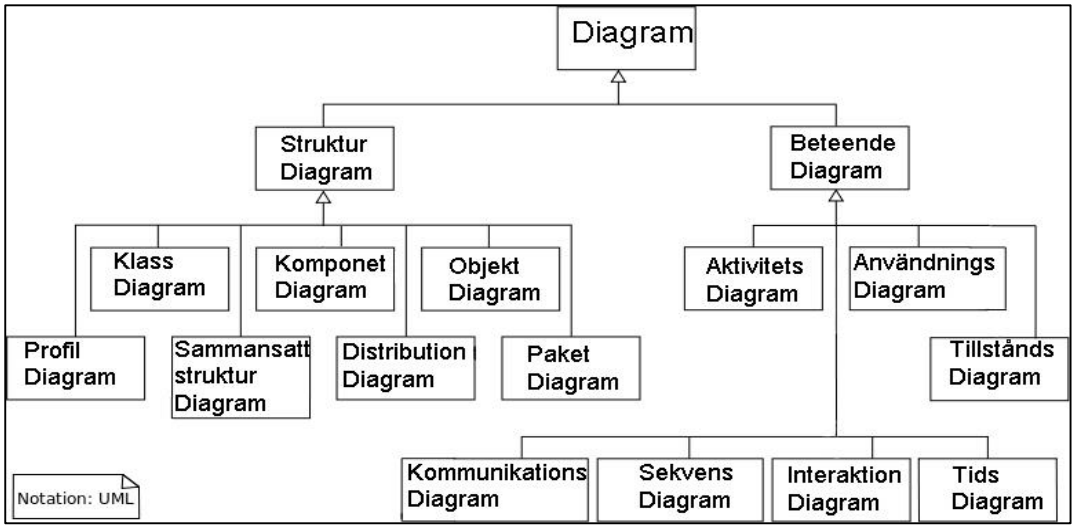
*Figur 10. Utan Entity Framework exempel. [14]*

## 3.5 Unified Modeling Language

För att skapa diagram över ett program kan man använda sig av Unified Modeling Language (UML). UML är ett språk för modellering, som används t.ex. inom objektorienterad programvarukonstruktion. För att skapa ett diagram över systemet kan man t.ex. använda sig av Microsoft Visual Studio. UML-diagram kan delas in i två större delar som är strukturdiagram och beteendediagram. [19]

Till strukturdiagrammen hör klass-, komponent-, sammansatt struktur-, distributions-, objekt-, paket- och profildiagram.

Till beteendediagrammen hör aktivitet-, kommunikations-, interaktions-, sekvens-, tillstånds-, tids- och användningsdiagram.




*Figur 11. UML-diagram. [19]*

De några av de vanligaste diagram som används är aktivitetsdiagram, användningsdiagram, klassdiagram och sekvensdiagram.

- **Aktivitetsdiagram** är grafiska representationer av arbetsflöden för aktiviteter och åtgärder med stöd för val, iteration och samtidighet.
- **Användningsdiagram** är en representation hur en användare kan använda systemet.
- **Klassdiagram** är diagram över alla klasser, arv, metoder osv.
- **Sekvensdiagram** är ett interaktionsdiagram som visar hur processerna fungerar med varandra och i vilken ordning.
  [19, 20]

## 3.6 Visual Studio Online

Visual Studio Online är ett online-utvecklingsverktyg, som är baserat på versionshanteringssystemet Team Foundation Server. Program som kan använda sig av Visual Studio Online är Microsoft Visual Studio, Eclipse, Xcode, och andra Git-klienter. Oberoende av hur många personer som arbetar med ett projekt, klarar versionshanteringen i Visual Studio Online av avancerade förgreningar och sammanslagningar.

Andra verktyg, som ingår i Visual Studio Online, är ett projekthanteringssystem med Scrum-metodiken, möjlighet att sammanställa din kod till ett program. Visual Studio Online är en så kallad molntjänst och behöver därför inte installeras eller konfigureras lokalt på en klient. Ett program eller en tjänst som körs på en server med internet-access, som användarna fjärr-använder, kallas för en molntjänst.

Versionshanteringen som är en funktion i Visual Studio Online, är en nödvändighet som behövs för både stora och små projekt. Versionshanteringen i Visual Studio Online laddar upp endast de filer som man har gjort modifieringar i. Alla de ändringar man sparar sammanställs till en ny version av projektet. Eftersom de ändringar man har gjort sparas till ett nytt projekt går det inte att ladda upp filer som innehåller fel.

Om något skulle gå fel med den lokala versionen man har på sin klient eller man vill återgå till en tidigare version, är det bara att hämta den specifika versionen man vill ha från Visual Studio Online.

Med versionshantering är det också enklare att arbeta med flera personer inom samma projekt. Ett exempel på versionshantering, kunde vara att fyra personer jobbar med samma projekt och alla har ansvar för olika delar av projektet, men delarna i projektet är beroende av varandra. När gruppen har arbetat med projektet under dagen, har det gjorts ändringar i flera filer och många nya filer har kommit till.

Nästa dag kan alla ladda ner en nya version till sin dator och få tillgång till alla de nya filerna och på så sätt kunna arbeta vidare utan att behöva vänta på att någon skall bli helt klar med sina delar.

Ett annat exempel är att man testar en annan lösning på ett problem, men som inte fungerar lika bra som den förra versionen. Då kan man enkelt ladda ner den föregående version och fortsätta därifrån utan att behöva skriva om allt på nytt.

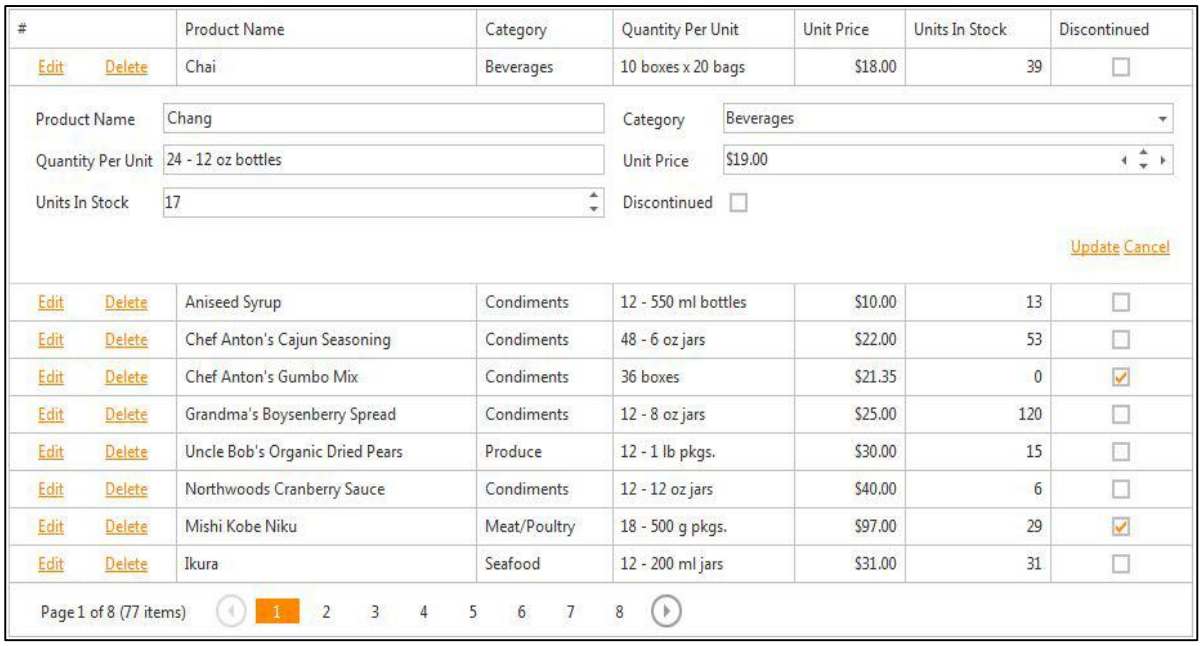
Med utvecklingsverktyget Microsoft Visual Studio är det väldigt enkelt att använda Visual Studio Online. Man skapar ett projekt i Visual Studio och kopplar detta projekt till Visual Studio Online. När man skall ladda upp sin version av programmet till molnet kan man jämföra sin aktuella version med den senaste som finns lagrat i molnet. Med denna funktion kan man enkelt se om man vill behålla ändringarna eller om man vill återställa till en föregående version som finns i molnet. Om man vill veta vad som har hänt med en specifik fil kan man hämta filens historik.

Om man arbetar från flera olika platser och många olika enheter, kan man ladda ner projektet till den nya enheten och fortsätta från den senaste versionen av programmet. Detta är en väldigt bra funktion som underlättar kodhanteringen mellan flera olika enheter och kan även användas som en backuplösning. [22]

## 3.7 DevExpress

DevExpress är ett företag som erbjuder tredjepartskomponenter för att skapa användargränssnitt inom huvudsakligen Microsoft-miljö. Verktygen som DevExpress levererar erbjuder många användbara funktioner, som inte finns i ASP.NET. En mycket användbar komponent, som DevExpress ASP .NET MVC erbjuder, är GridView. GridView är en grafisk komponent, som visar information i tabellform och har editeringsmöjligheter. Extensionen DevExpress stöder ASP.NET MVC Model Binding.

DevExpress komponenter kan exempelvis användas då man vill visa ett produktregister. Man behöver endast bidra med en produktmodell och koppla modellen med DevExpress GridView. När man har gjort kopplingen ställer man in vilka kolumner man vill visa.
Se figur 12.



| # | Product Name | Category | Quantity Per Unit | Unit Price | Units In Stock | Discontinued |
|---|---|---|---|---|---|---|
| Edit Delete | Chai | Beverages | 10 boxes x 20 bags | $18.00 | 39 | ☐ |
| Product Name: Chang | | Category: Beverages | | | | |
| Quantity Per Unit: 24 - 12 oz bottles | | Unit Price: $19.00 | | | | |
| Units In Stock: 17 | | Discontinued: ☐ | | | | |
| Update Cancel | | | | | | |
| Edit Delete | Aniseed Syrup | Condiments | 12 - 550 ml bottles | $10.00 | 13 | ☐ |
| Edit Delete | Chef Anton's Cajun Seasoning | Condiments | 48 - 6 oz jars | $22.00 | 53 | ☐ |
| Edit Delete | Chef Anton's Gumbo Mix | Condiments | 36 boxes | $21.35 | 0 | ☑ |
| Edit Delete | Grandma's Boysenberry Spread | Condiments | 12 - 8 oz jars | $25.00 | 120 | ☐ |
| Edit Delete | Uncle Bob's Organic Dried Pears | Produce | 12 - 1 lb pkgs. | $30.00 | 15 | ☐ |
| Edit Delete | Northwoods Cranberry Sauce | Condiments | 12 - 12 oz jars | $40.00 | 6 | ☐ |
| Edit Delete | Mishi Kobe Niku | Meat/Poultry | 18 - 500 g pkgs. | $97.00 | 29 | ☑ |
| Edit Delete | Ikura | Seafood | 12 - 200 ml jars | $31.00 | 31 | ☐ |

Page 1 of 8 (77 items) 1 2 3 4 5 6 7 8

*Figur 12. DevExpress grid view. [6]*

Varje kolumn i en GridView konfigureras skilt för sig, vilket innebär att alla kolumner kräver egna konfigureringar. DevExpress stöder model binding, som möjliggör att informationen skickas till controllern i en modell. [6]

# 4 Projekthantering

Projekthantering är en viktig process för både stora och små projekt. Projekthantering kan delas in i fem grupper: inledning, planering, verkställande, övervakning och styrning samt avslutning.

## 4.1 Scrum

Scrum är ett projekthanteringsramverk som används bland annat inom programutveckling. Ramverket Scrum är teambaserad och innehåller tre olika roller: produktägare, Scrum-master och utvecklingsteam.

Produktägare är den som bestämmer och ansvarar över vad som skall göras.

Scrum-master är den som hjälper utvecklingsteamet och organisationen med Scrum.

Utvecklingsteam är utvecklarna, som är självorganiserade och som bygger produkten ett steg i taget, enligt vad som har överenskommits skall bli gjort under en tidsperiod d.v.s. sprint. Dessa sprintar är en tidsperiod mellan en till fyra veckor, och under denna tidsperiod levererar utvecklingsteamet ett produktinkrement.

Scrum består av olika beståndsdelar. De huvudsakliga delarna är produktbacklogg, sprintbacklogg och produktinkrement.

Produktbacklogg är en lista på önskade nya funktioner till produkten och är ordnad i den ordning som funktionerna skall utvecklas. Alla arbeten som utvecklingsteamet utför är kopplad till en rad i produktbackloggen.

Sprintbacklogg listar de rader som valts från produktbackloggen i detalj och som skall finns med i nästa sprint. Alla punkter, som finns i sprintbackloggen, är valda så att arbetarna skall hinna utföra alla punkter före sprintens slutdatum.

Produktinkrement är produkten efter en avslutad sprint. Det skall vara en ny fungerande version med förbättrad funktionalitet samt uppfylla de överenskomna villkor som bestämdes vid föregående möte. [16]

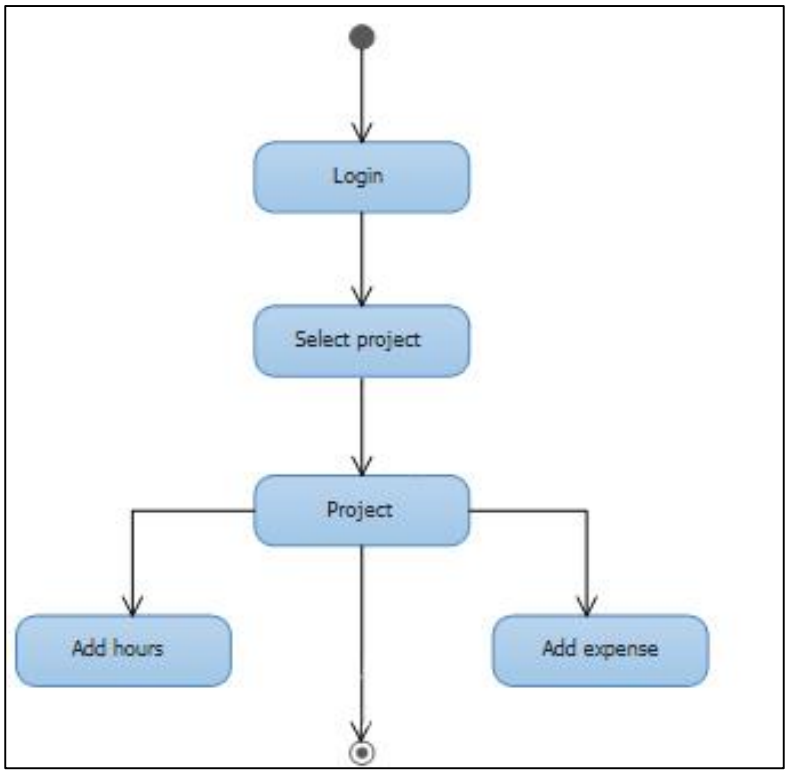
# 5 Utförande

Webbapplikationen vidareutvecklades vid Indevit Solutions kontor med deras resurser, såsom datorer och utvecklingsverktyg.

## 5.1 Planering

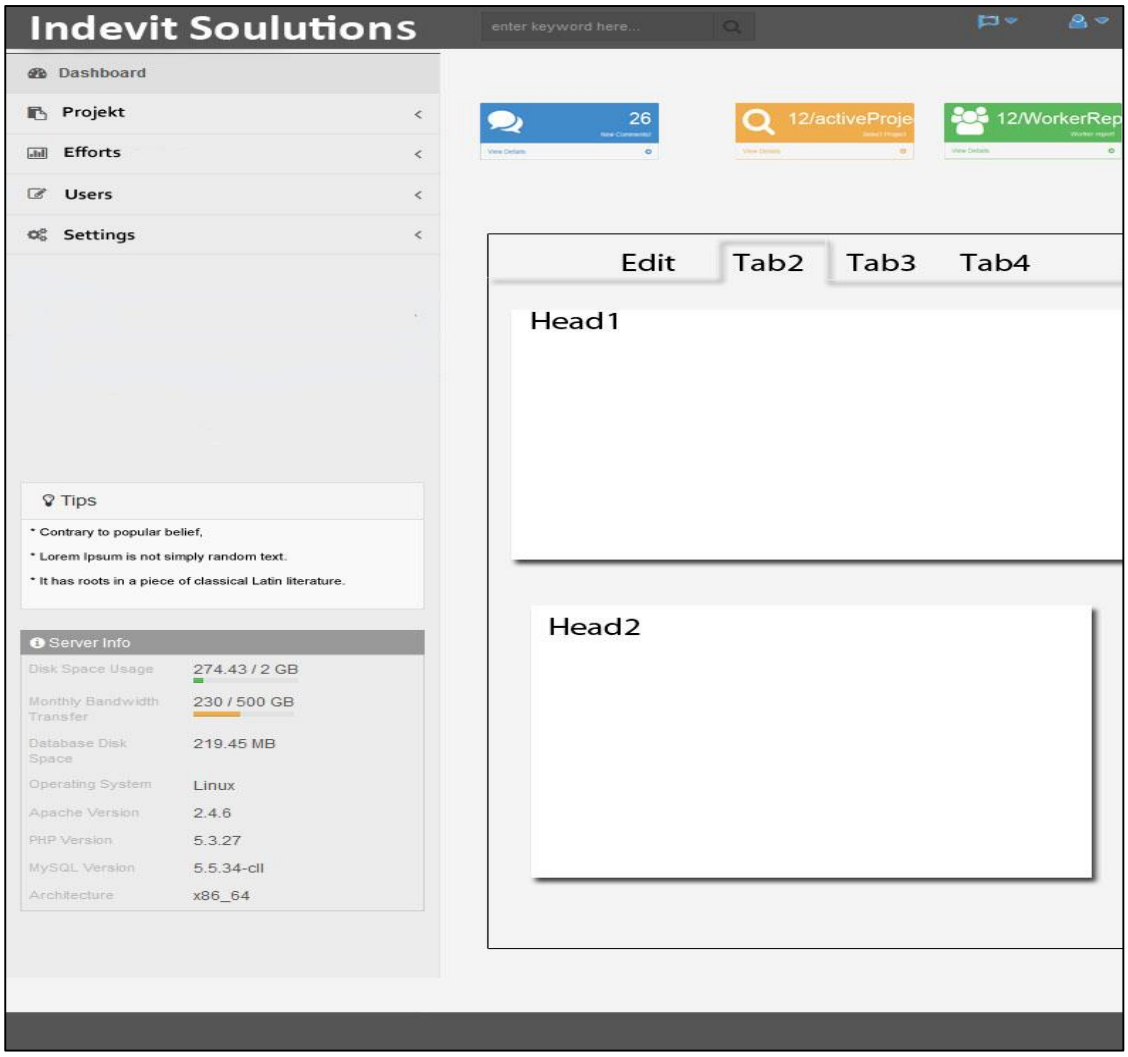
Hanteringen av detta projekt följde Scrum-metodiken. Varje månad anordnades ett sprintmöte, där en genomgång av föregående sprint gjordes. Efter genomgången planerades det vad som skulle ingå i nästa sprint och ett datum för nästa möte bestämdes.

Programfunktionaliteten för en användare planerades i form av ett flödesdiagram som beskriver vad användaren skall kunna göra. Se figur 13.




*Figur 13. Aktivitetsdiagram för användaren.*

Utvecklingen av användargränssnittet blir mycket enklare om en skiss över gränssnittet finns tillgängligt. I detta projekt gjordes en skiss över gränssnittet genom att hämta inspiration från webbsidor som använder ramverket Bootstrap [5]. Se figur 14.




*Figur 14. Skiss på användargränssnittet.*

## 5.2 Utveckling

Utvecklingen av webbapplikationen gjordes i Microsoft Visual Studio 2013 med ASP.NET MVC5. Extensionen DevExpress ASP.NET MVC användes för visning av data i användargränssnittet. Webbapplikationen visar samma sidor för alla användare oberoende av vilken roll användaren har, men vissa funktioner och visuella delar utelämnas om användaren inte har rättigheterna för dessa.

Rollen administrator har full kontroll över projektet och kan editera projektet, lägga till och ta bort användare, lägga till eller ta bort rättigheter för dessa användare, samt ha tillgång till alla andra funktioner som arbetare och kunderna har tillgång till.

Rollen arbetare har mindre rättigheter än administratorn. Arbetare kan se projektets information, men kan inte editera informationen. Den huvudsakliga funktionen för en arbetare är registrering av arbetsprestationer.

Rollen kund har endast läsrättigheter och kan endast se informationen om projektet.

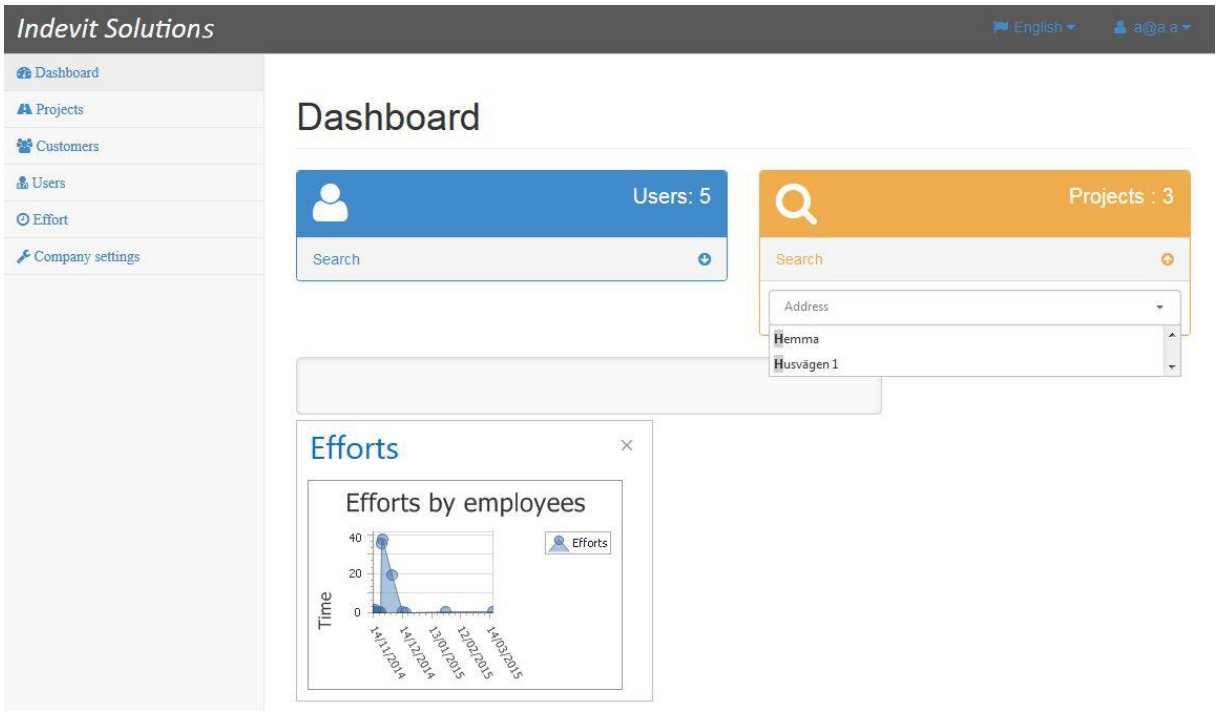
Hantering av användarautentisering sker i controllern och finns färdigt inbyggd i ASP .NET MVC5. För att kontrollera om användaren har rättighet till en funktion, används ett autentiseringsattribut framför funktionerna eller metoderna. Detta attribut kallar på en ASP .NET metod som avgör om användaren har den definierade rollen. I figur 17 visas hur dessa attribut används för att kontrollera om användaren har de rätta rollerna.

En tabell Roles skapas i databasen av ASP .NET när programmet körs första gången. Tabellerna som ASP .NET skapar innehåller ingenting. Innan man kan använda sig av autentiseringsattributen måste man definiera rollerna i databasen.

### 5.2.1 Översiktspanel

Första sidan man kommer till när man har loggat in till systemet, visas som en översiktspanel Information som visas på första sidan är gemensam för alla projekt som finns i projekthanteringssystemet. Översiktspanelen innehåller även genvägar till en specifik användare eller ett specifikt projekt.

Genvägarna är skapade med DevExpress ComboBox som är en listredigerare. ComboBox fungerar som en sökfunktion. Man skriver in text i textrutan och en lista med objekt renderas baserat på söktexten. Hämtningen av listorna sker asynkront via en callback-funktion, vilket innebär att listorna hämtas i bakgrunden och uppdaterar endast den aktuella genvägen. I figur 15 visas hur genvägarna ser ut.




*Figur 15. Översiktspanel med genvägar och statistik över prestationer.*

Controllern för översiktspanelen hämtar information från databasen och skapar listor till genvägarna och diagrammet. Listorna som används för genvägarna, är av typen SelectListItem. Denna typ används eftersom DevExpress ComboBox kan använda sig av SelectListItem för att visa en text och på samma gång finns ett unikt värde kopplat till texten.

Texten som visas i t.ex. projektgenvägen är adressen och värdet är projektets id. Listan skapades genom att hämta alla projekt som den aktuella användaren har tillgång till och som är aktiva. Projektadressen sparades som text och projektid som värde i listan. Hur listan skapades visas i figur 16.

```
public static IEnumerable<SelectListItem> GetAvailableProjects(RitvaDbEntities db, ApplicationUser applicationUser)
{
    var currentUser = GetUserId(db, applicationUser);
    var availableProjects = (from project in db.RV_Project
                             join projectAccess in db.RV_ProjectAccess on project.ProjectId equals projectAccess.ProjectId
                             where projectAccess.UserId == currentUser && project.Active == true
                             select project);

    IList<SelectListItem> projectsCollection = new List<SelectListItem>();
    if (availableProjects.Count() > 0)
    {
        foreach (var item in availableProjects)
        {
            projectsCollection.Add(new SelectListItem
            {
                Text = item.ProjectAddress.ToString(),
                Value = item.ProjectId.ToString()
            });
        }
        return projectsCollection;
    }
    else
    {
        return projectsCollection;
    }
}
```

*Figur 16. Listan för projektgenvägen.*

Data för diagrammet är en lista som innehåller en prestationsmodell. Denna lista skapades genom att först hämta alla prestationer som grupperas efter datum med hjälp av LINQ, och sedan skapa nya prestationsmodeller som sparas i listan. För att gör diagrammet tydligare räknas prestationstiden om till timmar. Denna kod finns i översiktspanelens controller och visas i figur 17.

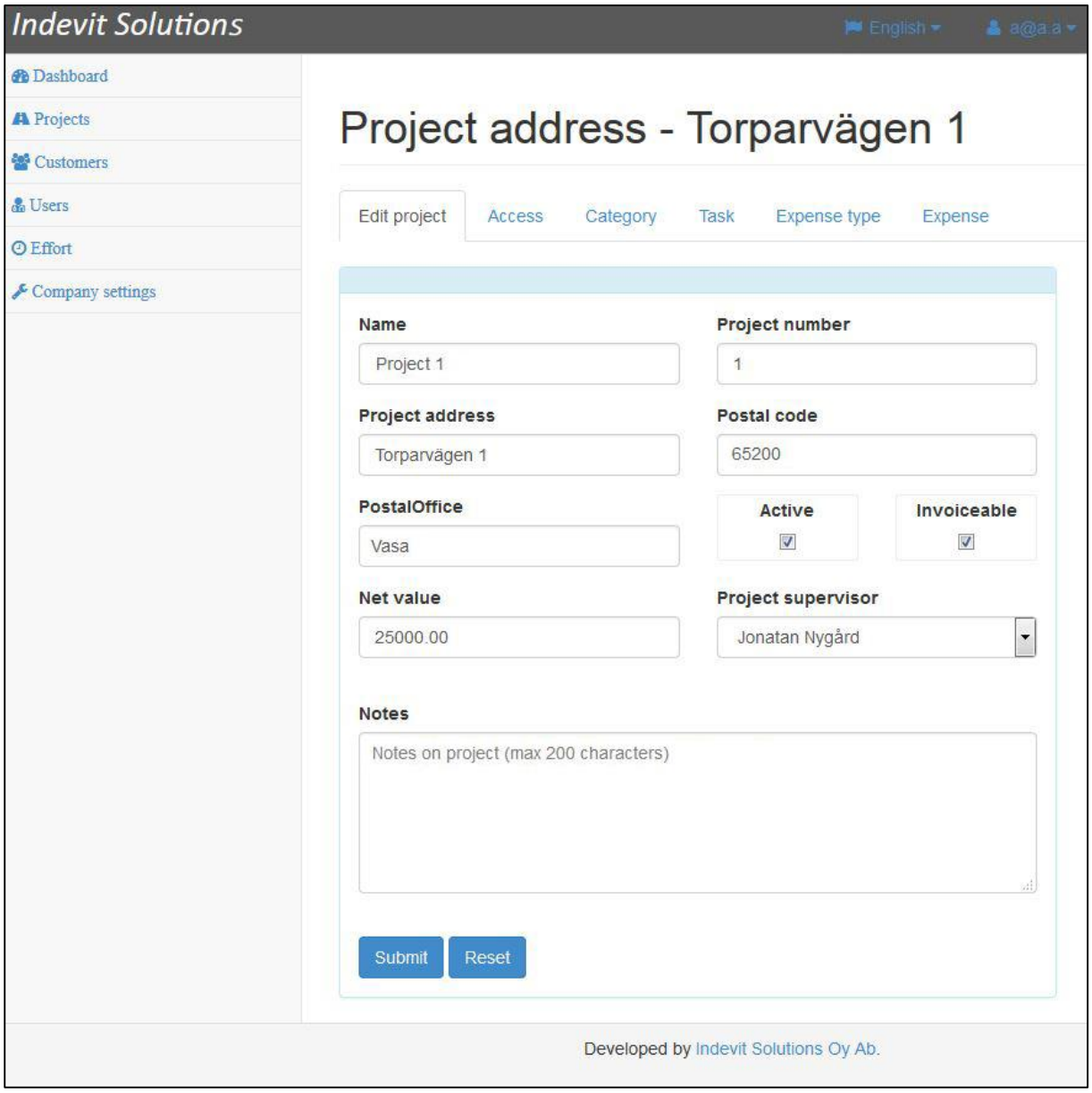
```
[Authorize(Roles = "Administrator, User")]
public ActionResult Index()...

[Authorize(Roles = "Administrator, User")]
public ActionResult About()...

[Authorize(Roles = "Administrator, User")]
public ActionResult Contact()...

[Authorize(Roles = "Administrator, User")]
public List<EffortList> GetEffortList(RitvaDbEntities db, ApplicationUser applicationUser)
{
    var effortEntities = from effort in db.RV_Effort.AsEnumerable()
                         group effort.Time by effort.EffortDate into effortDate
                         select new EffortList() { EffortDate = effortDate.Key, Time = effortDate.Sum() };

    List<EffortList> effortCollection = new List<EffortList>();
    if (effortEntities != null)
    {
        foreach(var item in effortEntities)
        {
            effortCollection.Add(new EffortList()
            {
                EffortDate = item.EffortDate,
                TotalTimeDecimal = (Math.Truncate(((double)item.Time / 60)*100)/100)
            });
        }
    }
    return effortCollection;
}
```

*Figur 17. Översiktspanelens controller.*

### 5.2.2 Projekteditering

Gränssnittet för hanteringen av ett projekt gjordes med flikar. Flikhanteringen gjordes med hjälp av flikfunktionen i jQuery-UI [10]. Sidorna i de olika flikarna skulle renderas asynkront. För att få denna funktionalitet skapades sidorna som partial view.

Partial View (partiell vy) betyder att sidor renderas inne i en annan sida. När man laddar om eller renderar en partiell vy berörs inte huvudsidan utan endast den partiella vyn, vilket uppfattas som att man inte laddar om sidan alls. Se figur 18.

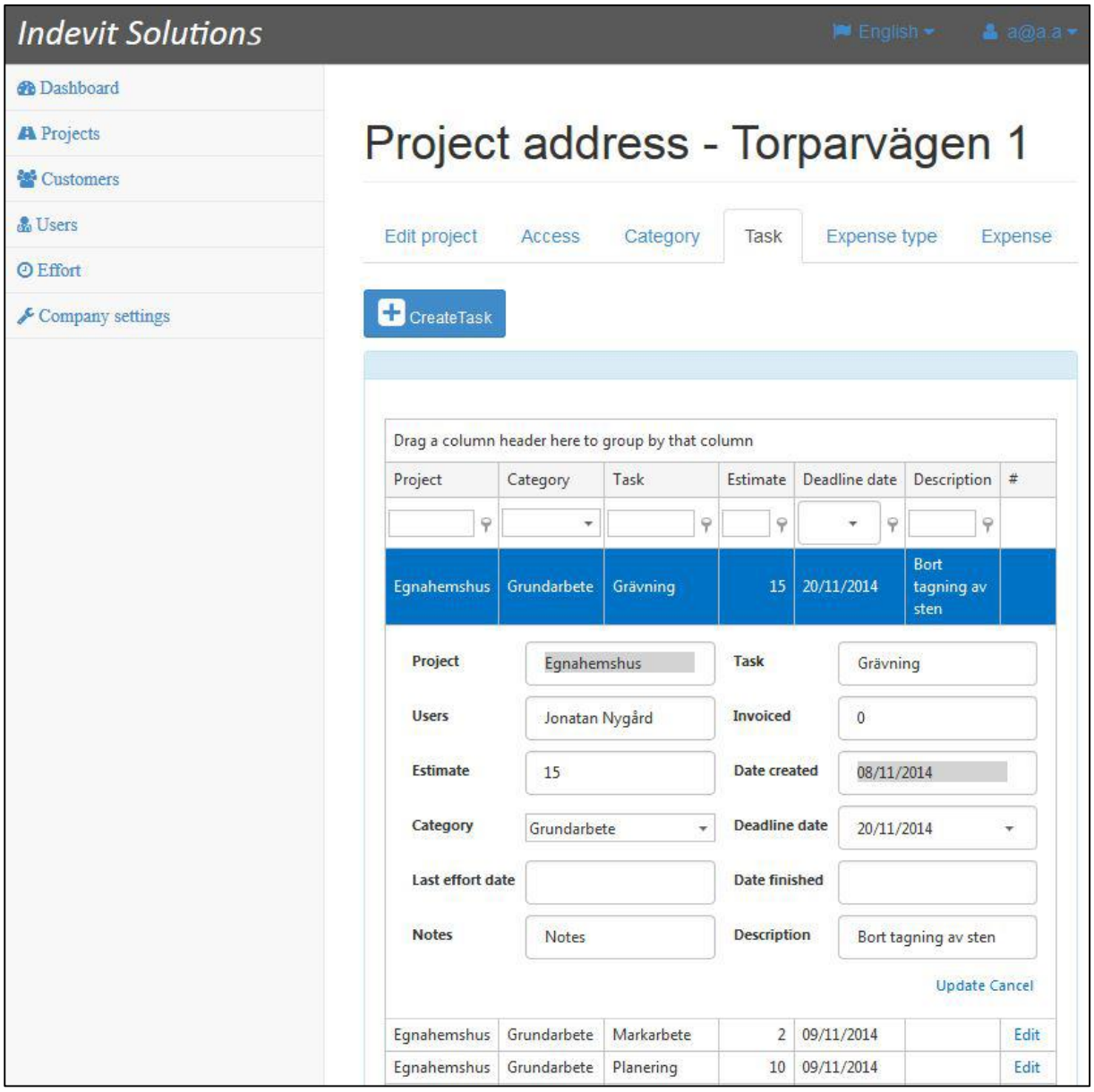



*Figur 18. Hantering av projekt i flikuppställning.*

Visningen av kategorierna, uppgifterna, utgiftstypen och utgifter görs via DevExpress GridView, som är ett rutnät med data. För att enkelt hitta rätt data ur en GridView finns en sorterings- och sökningsfunktion inbyggd i själva GridView. Sortering av data i en GridView görs från kolumnerna och användaren själv väljer hur informationen skall sorteras.

Sökning av data görs via ett textfält och finns i varje kolumn. Sökfältet har inställningar om hur funktionen skall behandla söktexten. Användaren själv kan använda sig av dessa inställningar.

För både kategorierna och uppgifterna är sökningen inställd så data i GridView kan innehålla söktexten. Editeringen av en kategori eller uppgift sker direkt i DevExpress GridView. När man väljer att editera en rad renderas ett dynamiskt formulär med editeringsmöjligheter. Se figur 19.




*Figur 19. GridView för uppgifter.*

För att spara alla ändringar, uppdatera fliken eller att ta bort en rad användes en callback-metod som finns i GridView. Callback-metoden kallar på controllern asynkront, vilket innebär att när controllern är klar med uppgiften, uppdateras GridView med en ny modell från controllern utan att sidan laddas om.

För att skapa en kategori, arbetsuppgift, utgiftstyp eller utgift användes Ajax-form som är nästan samma sak som ett vanligt formulär, men det fungerar asynkront precis som callback-metoden i GridView. Se figur 20.

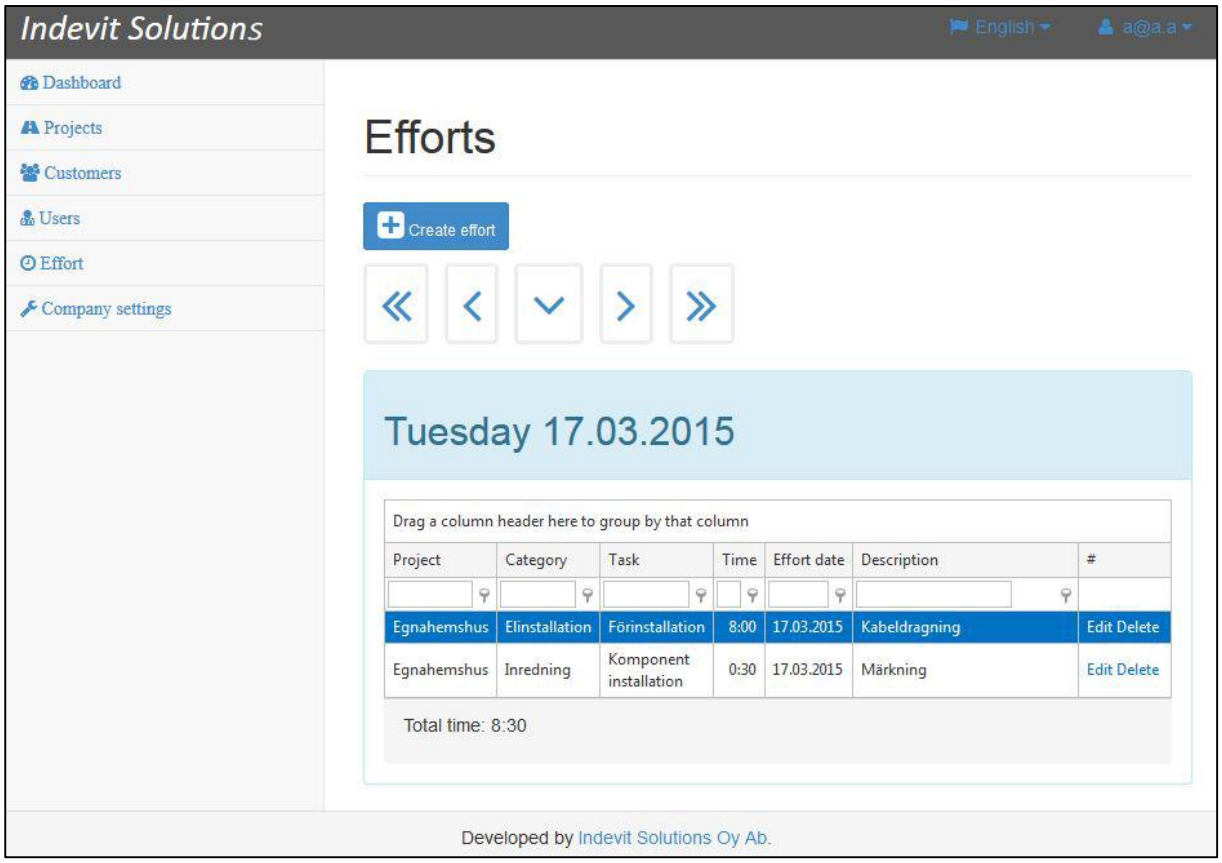
```
<div id="new_effort_content" style="display:none">
    @using (Ajax.BeginForm("CreateEffort", "Effort",
                           new AjaxOptions
                           {
                               HttpMethod = "POST",
                               OnBegin = "loadingPanelEffort.Show();",
                               OnComplete = "loadingPanelEffort.Hide(); gridViewEffort.Refresh();",
                               UpdateTargetId = "_CreateEffortPartial",
                               InsertionMode = InsertionMode.Replace,
                               OnFailure = "ReloadFormEffort(xhr)",
                               OnSuccess = "$('#new_effort_content').slideToggle(); clear_effortform_elements(); "
                           },
                           new
                           {
                               id = "validationFormEffort",
                               @class = "edit_form"
                           }))
    {
        <div id="contentCreateCategory">
            @Html.Partial("_CreateEffortPartial", ViewBag.EffortModel as Ritva.Models.EffortModel)
        </div>
    }
</div>
```

*Figur 20. Ajax formulär.*

### 5.2.3 Arbetsprestationer

Användaren som är involverad i ett projekt kan skriva in sin arbetsprestation per dag och per uppgift. Arbetsprestationen är kopplad till en användare och en uppgift, vilket betyder att en prestation måste ha en användare och en uppgift. Arbetsprestationssidan är beroende av datum. När man väljer ett nytt datum, laddas sidan om och visar endast prestationer för det valda datumet. När man lägger till en ny prestation används det valda datumet.

Visningen och editeringen av arbetsprestationen görs via DevExpress GridView likadant som för kategorierna och uppgifterna. Se figur 21.




*Figur 21. Sida för arbetsinsatser.*

Sidan för arbetsprestationerna är asynkron när man skapar och editerar en prestation eller när tar bort en prestation, men när man byter datum laddas hela prestationssidan om. Detta på grund av att navigeringsknapparna för datumen är länkar och skickar med datumet som parameter till controllern. I controllern skapas en modell för det specifika datumet.

### 5.2.4 Rapport

Rapporten skapades med hjälp av DevExpress. Utseendet för rapporten konstruerades med DevExpress rapportdesigner som fungerar med drag and drop, vilket innebär att man väljer t.ex. en textruta och placerar den på rapporten där man vill ha den. Rapporten byggdes upp med ”Report Bands” som gör att man delar in rapporten i olika grupper såsom t.ex. sidhuvud. ”Report Band” fungerar väldigt bra, för man kan länka ett band till en lista så rapporten renderar listans innehåll.

Data som visas på rapporten är en listning om alla användare som har arbetat med projektet en viss månad med den information som behövs för rapporteringen till skatteverket.

Information som hämtas till rapporten sker i controller via en callback. För att få data ur databasen för ett specifikt projekt och en specifik månad, skapades en SQL select-sats.

En koppling mellan tabellerna gjordes mellan användare, prestation, uppgift och kategori, denna koppling behövs för att projekt-id finns i kategori. Med projekt-id och månad får man hämtat alla användare som har prestationer till projektet under den specifika månaden.

## 5.3 Resultat

Resultatet av examensarbetet blev en webbapplikation med en grundfunktionalitet för att hantera projekt. Användargränssnittet för webbapplikationen är strukturerat med flikar, använder responsiv webbdesign och har asynkron funktionalitet. Rapporten för byggnadsindustrin blev en enkel listning av information om alla användare som har jobbat på projektet under ett visst tidsintervall. Webbapplikationen är inte helt färdigställd, men kommer att färdigställas inom en snar framtid.

## 5.4 Diskussion

Med detta examensarbete fick jag en bra insyn i hur programutvecklingsprojekt går till. Under utvecklingen av webbapplikationen lärde jag mig att oavsett hur mycket tid man lägger på en funktion kan man alltid förbättra den. Jag fick arbeta självständigt och i efterhand kan jag konstatera att jag lade ner för mycket tid på enskilda funktioner i början av examensarbetet för att kunna bli klar inom utsatt tid. Jag lärde mig även att återanvända kod och att använda extensioner såsom DevExpress.

Detta examensarbete blev gjort som en webbapplikation och när det handlar om användargränssnitt och webbteknologi är det svårt att förutspå hur allt kommer att fungera och se ut i förväg. Därför måste man skapa en prototyp av användargränssnittet för att kunna avgöra om det blev bra eller inte. Om prototypen av användargränssnittet inte blev som man förutspådde är det bara att börja om.

Tidsplaneringen i ett programutvecklingsprojekt är en svår uppgift att genomföra. I tidsplaneringen behöver man uppskatta tid för hela projektet, även för testning av produkten och för att lösa oväntade problem som kan uppstå.

Oväntade problem uppstår nästan alltid i ett projekt och det är svårt att beakta hur mycket tid det krävs för att lösa dem. I detta projekt spenderades det för mycket tid på fel ställen, såsom att förbättra en fungerande funktion.

Med hjälp och handledning av Christoffer Smeds och Peter Hertsbacka från Indevit Solutions, lärde jag mig mycket om ASP .NET MVC och fick chansen att lära mig mer och vidareutvecklas som programvaruutvecklare.

# 6 Källförteckning


[1] Anmälningar om byggande Företags- och samfundskunder
www.skatt.fi/sv-FI/Foretags_och_samfundskunder/Anmalningar_om_byggande
[Hämtat 18.03.2015]

[2] Anmälningar om byggande personkunder
www.skatt.fi/sv-FI/Personkunder/Anmalningar_om_byggande
[Hämtat 18.03.2015]

[3] ASP .NET MVC
*www.asp.net/mvc/overview/older-versions-1/overview/asp-net-mvc-overview*
[Hämtat 15.01.2015]

[4] Bootstrap ramverk hämtnings sida
*www.getbootstrap.com*
[Hämtat 17.03.2015]

[5] Bootstrap användargränssits SB-Admin2 tema
*www.startbootstrap.com/template-overviews/sb-admin-2*
[Hämtat 18.03.2015]

[6] DevExpress ASP .NET MVC demos
*https://demos.devexpress.com/MVC*
[Hämtat 15.01.2015]

[7] Entity Framework Code-First
*www.entityframeworktutorial.net/code-first/what-is-code-first.aspx*
[Hämtat 11.04.2015]

[8] Ilmoitin Hjälpverktyg för skapande av filer
*www.ilmoitin.fi/webtamo/sivut/ExcelPohjat?0&kieli=sv*
[Hämtat 01.03.2015]

[9] Indevit Solutions
*www.indevit.fi*
[Hämtat 15.01.2015]

[10] jQuery-UI tabs
*www.jqueryui.com/tabs*
[Hämtat 18.03.2015]

[11] Language Integrated Query
*www.en.wikipedia.org/wiki/Language_Integrated_Query*
[Hämtat 10.03.2015]

[12] Model-View-Controller
*www.en.wikipedia.org/wiki/Model-view-controller*
[Hämtat 15.01.2015]

[13] .NET ramverk
*www.msdn.microsoft.com/en-us/library/zw4w595w%28v=vs.110%29.aspx*
[Hämtat 14.01.2015]

[14] Object-Relational Mapping
*www.msdn.microsoft.com/en-us/library/ms178359.aspx#orm*
[Hämtat 17.02.2015]

[15] Galloway, Jon. Wilson, Brad. K. Scott, Allen. Matson, David. (2014).
*Proffesional ASP.NET MVC 5.* Indianapolis: John Wiley & Sons, Inc.

[16] Scrum en beskrivning
Översatt av Åhlander, A. & Fagrell, P. (2013)
*www.scrumalliance.org/scrum/media/ScrumAllianceMedia/Files%20and%20PDFs/Why%20Scrum/Core%20Scrum%20Translations/Core-Scrum-Swedish.pdf*
[Hämtat 15.01.2015]

[17] Structured Query Language
*www.sv.wikipedia.org/wiki/Structured_Query_Language*
*[Hämtat 10.03.2015]*

[18] TempData
*www.webdevelopmenthelp.net/2014/06/using-tempdata-asp-net-mvc.html*
[Hämtat 09.04.2015]

[19] Unified Modeling Language
*www.en.wikipedia.org/wiki/Unified_Modeling_Language*
[Hämtat 15.01.2015]

[20] Unified Modeling Language diagrams
*www.uml-diagrams.org*
[Hämtat 11.04.2015]

[21] Utvärdering av Entity Framework Object-Relational Mapping
*www.diva-portal.org/smash/get/diva2:518944/FULLTEXT01.pdf*
Hall, A. & Hindrikes, D., 2010. Entity Framework 4.0, en utvärdering av ett ORM-ramverk. Dalarna: Examensarbete, Grundnivå 2 Informatik. Högskolan Dalarna, Akademin industri och samhälle.
[Hämtat 17.02.2015]

[22] Visual Studio Online
*www.visualstudio.com/en-us/products/what-is-visual-studio-online-vs.aspx*
[Hämtat 15.03.2015]

[23] ViewBag
*www.msdn.microsoft.com/en-us/library/system.web.mvc.controllerbase.viewbag(v=vs.118).aspx*
[Hämtat 09.04.2015]

[24] ViewData
*www.msdn.microsoft.com/en-us/library/system.web.mvc.viewpage.viewdata(v=vs.118).aspx*
[Hämtat 09.04.2015]