Customer-Relationship Management System til Teknologisk Institut

Maria Miland Elmvang, s991112

31 marts, 2005

Polyteknisk eksamensprojekt

Institut for Matematisk Modellering

Danmarks Tekniske Universitet

Vejledere: Jens Thyge Kristensen & Hans Bruun

Indhold

Tabeller

Figurer

1 Abstract

In order for a company to have as effective an interaction with their customers as possible, it is important that information about their customers are saved and stored in an easily accessable manner. One way to do this is through the Customer-Relationship Management(CRM) system. The data are stored in a database and accessed through e.g. webpages.

The Danish Institute of Technology(DTI) were in possession of such a database used for storing information about customers. Among other things this information included name and address of customers, which projects DTI had previously been involved with in their dealings with the customers, and which special offers or rebates the customer was entitled to. The data were used both in order to give the customer a better experience when contacting DTI, because DTI would already be aware of who they were talking to, and in order for DTI to have a clear view of which customers they currently had. The program used to access the data and maintain it (affectionately nicknamed 'Benny') was unfortunately not as effective or user-friendly as one could have wished. Therefore I was asked to write a new version of the program, through which maintenance of the data would be an easier task.

I knew very little about CRM systems before starting the assignment and therefore had to research the topic before being able to start the actual implementation.

As the program I was to write was supposed to be an alternative to the program already in existence I put a lot of focus on user-participation to make sure that the program I ended up with, actually fulfilled the requirements the end users had to the program. Through interviews I determined that the most important demands to the program were as follows:

- easy and fast navigation
- intuitive search
- the ability to see forecasts for projects
- intuitive 'update' and 'edit' functions

Based on these and other demands I determined a requirement specification the program should adhere to. From this I was able to make various use cases and thus design the program's functionality.

My contact at DTI was well aware that I had a limited amount of time available for this project, and therefore asked that the program easily could be developed further by other programmers. He therefore asked me to create ASP.NET pages using C# so other pages could easily be added at a later time.

I ended up with a total of only five webpages: The front page, which is a combined search page, where the most common search fields for 'companies', 'contacts' and 'projects' could be used; three specific pages for 'companies', 'contacts' and projects', where one can make an advanced search, add new elements to the database or update ones already there; and finally a page for viewing the forecasts of various employees at DTI.

It is my impression that while my program has fewer functionalities than the original program, I have fulfilled the demands set to me by the end users in that

where the two programs *do* share functionality, the program I have written is faster to use. I am confident that the end users will share this opinion.

2 Forord

Når man som studerende kommer ud til en så stor virksomhed som Danmarks Teknologiske Institut virker det hele meget overvældende, men det viste sig hurtigt, at alle jeg stødte på, var utrolig interesserede og hjælpsomme. Hvis jeg savnede noget, var der altid flere, der stod klar til at komme mig til hjælp. Lige fra teknisk hjælp med at få fat i en computer og installeret de rigtige programmer til at tilbyde mig pladser på relevanter kurser afholdt på DTI og give mig af deres tid til lange samtaler og evalueringer af projektet. Jeg kunne på ingen måde have lavet projektet uden deres bistand og moralske støtte. Jeg vil især gerne sige tak til:

Claus Tøndering - Seniorkonsulent ved DTI.
Gorm Bjerre - Cand. oecon., vedligeholder af 'Benny'.
Susanne Christoph - Centerchef ved DTI.
Margit Limkilde Bloch - Sælger ved DTI.
Ole Thingsted - Konsulent ved DTI.
Henrik Poulsen - Account manager ved DTI.
Jette B. Smith - Seniorkonsulent ved DTI. Vejleder om ASP.NET.

Jeg håber, at resultatet af projektet vil kunne finde anvendelse og gøre, at det har været tiden værd, for de personer der har hjulpet mig.

Maria Miland Elmvang, DTU, 2005

3 Indledning

Da jeg skulle i gang med at vælge et eksamensprojekt, begyndte jeg min søgen med at tage kontakt til Danmarks Teknologiske Institut(DTI) i håbet om, at de havde et projekt, jeg kunne deltage i. Det viste sig også at være tilfældet, da de sad i den situation, at søgeprogrammet til deres kundedatabase, ikke fungerede optimalt, og de derfor var interesserede i, at det blev forbedret.

Min kontaktperson på DTI forklarede, at de var interesserede i et 'CRM-light'-system, der kunne erstatte, eller i hvert fald virke som alternativ til søgeprogrammet, 'Benny', så udover et mere brugbart søge- og redigeringsprogram ønskede de også, at programmet blev integreret med andre programmer som f.eks. Internet Explorer og Outlook.

Jeg vil i dette projekt undersøge, både hvad begrebet "Customer Relationship Management"(CRM) dækker over og behovet for at have et hjemmelavet CRM-program i stedet for blot at anskaffe et af de mange programmer, der er til salg. Målet med projektet er at få en god forståelse af CRM-konceptet og at udvikle et program, der kan lette DTI's sælgers daglige arbejde.

Under planlægningen af projektet har jeg stræbt efter at holde DTUs retningslinier for et eksamensprojekt for øje og sørge for, at det indeholder flere forskellige elementer af en civilingeniørs arbejde. Dette er i høj grad kommet af sig selv, da jeg en stor del af vejen selv skulle definere og planlægge projektet. Blandt andet var det nødvendigt selv at skrive en problemformulering. I samråd med Gorm Bjerre på DTI har det været muligt at afgrænse opgaven til en passende størrelse.

Da programmets slutbrugere først og fremmest er sælgere, kan jeg ikke antage, at de også er computereksperter. Derfor har det gennem hele forløbet været vigtigt, at brugervenligheden og produktets generelle anvendelighed svarede til målgruppens forventninger. Dette har jeg gjort ved meget præcist at definere målgruppen og fastlægge dennes behov. Jeg har gennem hele forløbet fået feedback fra både slutbrugere og udvikleren af 'Benny' og har så forsøgt at indfri deres forventninger og ønsker, når det var muligt.

3.1 Motivation

Det var meget vigtigt for mig, at jeg fandt et eksamensprojekt, som havde praktisk værdi, så det ville kunne bruges til noget, når jeg blev færdig, i stedet for at lave et 'tænkt' projekt som bagefter blot ville stå og samle støv på hylden. Jeg vidste med mig selv, at det ville betyde meget for min motivation, at der var nogen, der ville få gavn af mit projekt.

Jeg var derfor meget glad for at finde et projekt, hvor resultatet ikke bare kan bruges til noget, men det vil udfylde et tydeligt hul. Jeg har modtaget megen positiv feedback selv inden jeg fik lavet noget, jeg kunne vise frem. De kommende brugere, jeg har snakket med, er meget hjælpsomme på alle måder, og de giver tydeligt udtryk for, at det vil lette deres arbejde væsentligt, hvis jeg, når projektperioden er ovre, kan præsentere dem for et færdigt produkt.

De fleste projekter, jeg hidtil har lavet ved DTU, har været i grupper på mindst to. Jeg var interesseret i at prøve kræfter med at lave et projekt alene, da jeg gerne

ville have noget erfaring med at arbejde alene.

Endnu en motivation til projektet var, at jeg gerne ville afprøve mange forskellige grene af det, jeg har lært på DTU. At opbygge et software-produkt kræver software-udviklingsteknikker. Men det er ikke blot gjort med at implementere et program, der virker og dokumentere det. Da programmet skal kunne udfylde et reelt behov, er det vigtigt, at brugergrænsefladen er intuitiv og brugervenlig. Dette kan jeg opnå ved at benytte teknikker fra interaktionsdesign (ID) og menneske-computer interaktion (Human-Computer Interaction, HCI).

3.2 Læsevejledning

Rapporten er delt op i 4 dele: afsnit 4 – afsnit 8 beskriver problemformuleringen og baggrunden for opgaven; i afsnit 9 kan man læse om det endelige program, projektet resulterede i; afsnit 10 – afsnit 12 omhandler arbejdsprocessen, analyse og design; endelig kan man i afsnit 13 – afsnit 15 læse de endelige tests og kommentarer til programmet.

Kommende brugere kan med fordel læse: afsnit 4 – afsnit 9. Derudover kan en brugervejledning findes i appendiks F.

Software-udviklere og andre teknisk interesserede vil derudover også have glæde af at læse: afsnit 10 – afsnit 13.

Personer som vil videreudvikle programmet kan anbefales at læse hele rapporten, men især afsnit 4, afsnit 9 og afsnit 15.

Koden til programmet vedlægges som bilag.

4 Problemformulering

Det første, jeg skulle gøre i projektforløbet, var at formulere en brugbar problemformulering, som jeg kunne arbejde efter. Dette blev gjort i samråd med mine vejledere både på DTU og DTI, således at problemformuleringen var passende til et polyteknisk eksamensprojekt og stemte overens med DTI's forventninger til mig.

4.1 Problemformulering

Det følgende er den problemformulering mit eksamensprojekt er baseret på.

Danmarks Teknologiske Institut (DTI) benytter oplysninger om deres kunder (typisk andre virksomheder) til at finde ud af, hvilke af de kurser og produkter som DTI tilbyder, der er relevante for den givne kunde. For at holde bedre styr på disse oplysninger, savner DTI et Customer-Relationship-Management (CRM) system. CRM bruges af mange virksomheder til at 'kende deres kunder'. Det hjælper virksomheder til at maksimere den værdi, de får ud af deres interaktion med kunderne og holde de fundne informationer samlet og let tilgængelige.

Selvom DTI har de informationer, de har brug for om deres kunder, er de på nuværende tidspunkt ikke struktureret optimalt. Derfor efterlyser de et søgesystem, der hurtigt og let kan give en profil af en virksomhed. Det skal være browserbaseret, således at man hurtigt og enkelt får nogle ganske få skærmbilleder med de relevante informationer om virksomheden. Disse informationer kan inkludere almene facts om virksomheden, deres interaktion med DTI, kontaktperson både hos virksomheden og på DTI o.l. Desuden skal det helst være muligt at sende email fra systemet og at opdatere systemet direkte.

Projektet er baseret på analyse-design-implementation. Størstedelen af projektet vil gå ud på at undersøge, hvordan systemet bedst muligt kan designes, dels ud fra inspirationer fra allerede kendte CRM-systemer, dels ud fra interviews med kommende brugere på DTI. Derefter skal systemet designes og slutteligt implementeres.

4.2 Afgrænsning af opgaven

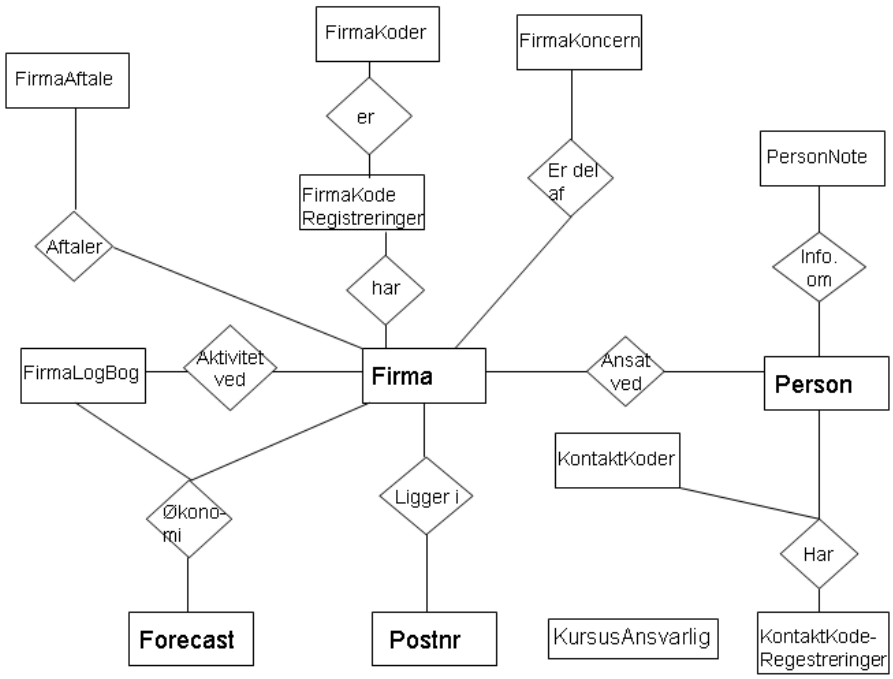
DTI manglede et program til at håndtere de informationer, de har i deres kundedatabase. Programmet skulle først og fremmest kunne søge gennem oplysningerne, men for at gøre det mærkbart bedre end det eksisterende program, skulle man også helst kunne redigere i oplysningerne og oprette nye oplysninger i databasen.

Databasen indeholder mange forskellige oplysninger udover kundeoplysningerne. Det er også der, man kan finde informationer om de kurser DTI tilbyder, DTI's ansatte m.v. I sin helhed er databasen så omfattende, at det ikke ville være realistisk at lave et program, der håndterede *alle* oplysningerne fra den på den - relativt - korte tid, jeg havde til min rådighed. I samråd med min vejleder på DTI, blev vi enige om, at jeg udelukkende skulle fokusere på de detaljer, DTI's sælgere bruger, når de kontakter en (potentiel) kunde, da det var på dette område, det eksisterende program lod mest tilbage at ønske. Det var især svært, at finde informationer om forløbet inden kunden underskriver en kontrakt, og dermed bliver mere end blot en potentiel kunde. For sælgere ville det være en fordel hurtigt at kunne se, hvilke

markedsføringsaktiviteter kunden er blevet præsenteret for, men bruger man det eksisterende system, kan man godt risikere, at det tager uforholdsmæssigt lang tid at få oplysningerne samlet. Det ville være en fordel at kunne lave *én* søgning på et firma og derved få en enkelt side som resultat med *alle* oplysningerne om det givne firma. Jo flere oplysninger, der kunne komme med på ét skærmbillede, desto bedre.

Det var som sagt også ønskbart, at brugerne kunne ændre i nogle af oplysningerne, når det blev nødvendigt, og at ændringerne kunne foregå på en let og intuitiv måde. I det eksisterende program er det kun nogle af oplysningerne, der kan opdateres direkte, andre kræver, at man benytter et helt nyt program. Det er ikke holdbart i længden, da det får mange til at udsætte - og måske helt glemme - opdateringerne. Databasen er kun brugbar, så længe man kan regne med, at den er aktuel.

De oplysninger, sælgerne har brug for, er detaljer om firmaer (dvs. kunder), kontaktpersoner og projekter hos kunden. Disse oplysninger er fordelt i databasen i de forskellige databasetabeller, der er vist i E-R diagrammet i figur 1. De mest generelle oplysninger kan findes i **Firma**, **Person** hhv. **FirmaLogBog**, men for at få alle detaljerne med er det også nødvendigt at inkludere de øvrige tabeller vist i figuren. Jeg har valgt ikke at inkludere attributter i diagrammet, da det ville gøre det unødvendigt uoverskueligt. Attributterne kan findes i databaseskemaet, figur 15 i afsnit 12. De fleste af tabellerne eksisterede i databasen i forvejen, men for at efterkomme brugernes behov var det nødvendigt at tilføje fire nye databasetabeller: `FirmaAftale`, `FirmaKoncern`, `Forecast` og `PersonNote`. Dette er beskrevet mere detaljeret i afsnit 12. `KursusAnsvarlig` er medtaget til brug ved login.



Figur 1: E/R-diagram over databasen.

4.3 Risikoanalyse

Når man går i gang med et længerevarende projekt, er det en udfordring hele tiden at have en tidsplan for øje og have overblik over, hvor man er, i forhold til hvor man burde være. For at være forberedt på de mest tænkelige problemer dette kan forårsage, har jeg gjort mig tanker om, hvilke elementer der kunne være med til at forsinke - eller i værste fald stoppe - projektet, og hvad der kan gøres for at undgå disse situationer.

Jeg er kommet frem til følgende risici:

1. Mangelfuld planlægning så projektet ikke overholder deadline.
2. Tekniske vanskeligheder med C# eller ASP.NET.
3. Tekniske vanskeligheder med computerudstyr.

Selvfølgelig er det ikke sikkert, at situationerne kan undgås fuldstændig, men fordelen ved at have identificeret de mest tænkelige risici på forhånd er, at jeg kan tage de nødvendige forholdsregler, så projektet kan gennemføres selv hvis en af disse situationer skulle opstå. Derfor har jeg følgende risikoimplementation for de tre ovennævnte punkter:

1. Dette er nok den største risiko, man løber ved at være alene om et længerevarende projekt. Det er vigtigt, at projektforløbet planlægges omhyggeligt, og at man arbejder i iterationer, således at der hele tiden eksisterer en fungerende version af programmet. Desuden skal rapporten skrives løbende under hele forløbet, så man hele tiden har dokumentation for de ting, man har lavet.
2. Både C# og ASP.NET er nye teknologier for mig. Da jeg imidlertid har programmeret meget i andre objekt-orienterede sprog før, bør C# ikke volde de store problemer for mig. Nogle af mine 'kollegaer' på DTI har megen erfaring med ASP.NET og vil ikke have noget imod at hjælpe mig med de problemer jeg måtte støde på. Det er også vigtigt, at jeg sætter mig ind i dokumentationen for programmeringssproget og muligheden for at stille spørgsmål på diskussionsgrupper på Internettet. Når jeg støder på et teknisk problem, er det vigtigt, at jeg hurtigt finder ud af, om det blot er et spørgsmål om at finde ud af, *hvordan* man løser det, eller om det er en funktionalitet, det ikke er muligt at benytte i det pågældende sprog. Dette kan jeg afklare ved at spørge mere erfarne brugere af sproget, enten på DTI eller over Internettet.
3. I forbindelse med projektet arbejder jeg det meste af tiden på DTI, hvor jeg har fået stillet en computer til rådighed. Gennem deres computer-support afdeling er der adgang til langt de fleste Microsoft produkter, så der vil derfor ikke være problemer med at få installeret de programmer, jeg får brug for under projektforløbet. Derudover er IT-supporterne meget kvalificerede og hjælpsomme og kan hjælpe mig med de tekniske problemer, jeg måtte løbe ind i.

5 Customer-Relationship Management

Da jeg først blev præsenteret for dette projekt, fik jeg at vide, at det var et 'Customer-Relationship Management' system, jeg skulle lave. På det tidspunkt havde jeg ganske vist hørt udtrykket før, men jeg havde ingen anelse om, hvad der egentlig gemte sig bag ordene. Min første opgave blev derfor at finde ud af, dels hvad der forstås ved udtrykket generelt, og dels hvad DTI forstod ved CRM. Derudover skulle jeg finde ud af, hvorfor de ikke bare benyttede sig af nogle af de CRM systemer, der er til salg rundt omkring på Internettet. Det viste sig desværre at være lettere sagt end gjort. Søger man på "Customer-Relationship Management" eller "CRM" på Google[1], får man ganske vist mange svar, men de fleste er blot reklamer, ikke sider der giver en forklaring på teorien bag det. Jeg har derfor i høj grad måttet benytte kvalificerede gæt baseret dels på, hvad jeg kunne få ud af hjemmesiderne, og dels på, hvad mine kontaktpersoner på DTI og DTU har fortalt mig.

5.1 Hvad er et CRM-system?

Generelt beskrevet er CRM en hjælp til at lave en forretnings-strategi, der sætter kunden i fokus og typisk berører hele virksomheden på det strategiske, organisatoriske og IT-systemmæssige niveau. CRM er designet til at optimere profit, udbud og kundernes tilfredshed.

De væsentligste grunde til at fokusere på CRM er, at det skaber fordele både for kunden og for virksomheden. Kunden sparer tid, fordi virksomheden på forhånd er klar over, hvilke relationer de har med kunden, hvilke tilbud der er relevante, og hvilke der blot ville være spild af tid. Kunden får tilpasset sine ydelser bedre og får dermed en oplevelse, der er præget af seriøsitet og professionalisme. Virksomheden kan indsamle, fastholde og anvende systematisk viden om kunder og får på denne måde meget tilfredse kunder, hvilket forhåbentlig vil øge virksomhedens indtægter. Samtidig bliver det lettere at holde sig opdateret om kundernes interaktion med virksomheden, hvilket også øger medarbejdernes tilfredshed.

Antallet af de virksomheder, der benytter et CRM-system til kundeaktiviteter og salgskontakter, er vokset med ca. 10% på det sidste år. Udbredelsen af CRM-systemer kan ses i figur 2, som er taget fra [8].

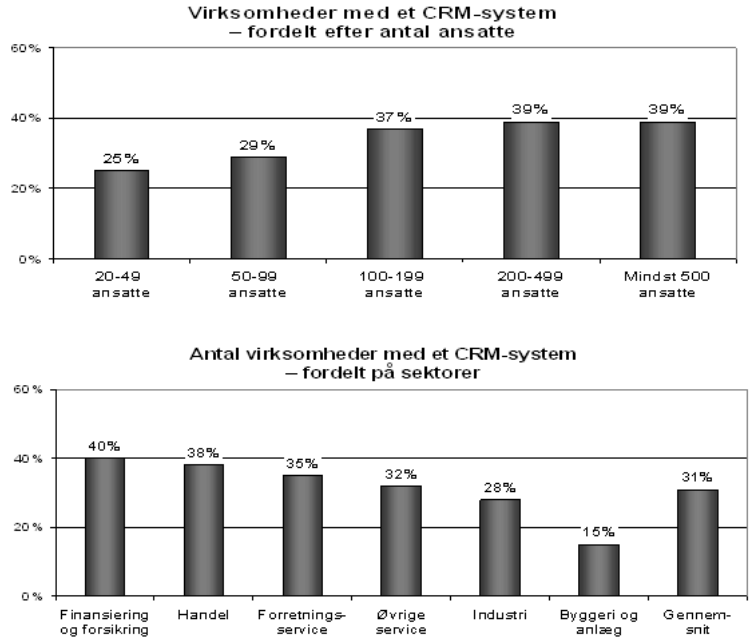
5.2 Hvad benyttes CRM-systemer til?

CRM-systemerne anvendes bredt i virksomhedernes organisation. De største brugere er salgs- og marketingafdelingerne, eftersom det er i de afdelinger, at det er vigtigst at have detaljeret viden om kunderne.

Mere konkret anvendes CRM-systemerne hovedsageligt til følgende aktiviteter, som er en blanding af registrering af data og elementer, der gør det muligt at udtænke strategier:

- Registrering af kundeinformationer.
- Salg til eksisterende kunder.

[1] http://www.google.com



Figur 2: Udbredelse af CRM-systemer i sektorer og efter virksomhedsstørrelse

- Vidensdeling mellem kollegaer.
- Opdeling af kunder i grupper.
- Styring af kampagner.
- Opdyrkning af salg til nye kunder.

Ved at bruge et CRM-system til disse aktiviteter får en virksomhed bedre overblik over deres kunder og hvilken interaktion, der har været mellem virksomheden og kunderne. Dette gør det lettere for virksomheden at gennemskue, hvad deres CRM-strategi bør være. En CRM-strategi bør ikke forveksles med et CRM-system. Selv virksomheder uden et CRM-system kan sagtens have en ganske udmærket CRM-strategi. Strategien omhandler virksomhedens planer og målsætninger for servicering og kontakt til deres kunder. Et CRM-system gør det simplere at realisere denne strategi, men det er ikke bydende nødvendigt at have et.

Et CRM-system er således et IT-understøttet værktøj til at realisere virksomhedens CRM-strategi. Hverken mere eller mindre.

Udover den allerede nævnte form for CRM-systemer lokalt i virksomheden er der også nogle virksomheder, der anvender e-CRM (electronic Customer Relationship Management) via deres hjemmeside, dvs. at kunderne betjenes på eget initiativ. I denne tidsalder hvor det efterhånden er mindretallet, der ikke har adgang til computer med Internet, er der et stort potentiale for udvikling af e-CRM.

Selvom DTI er interesserede i alle disse aspekter af benyttelse af et CRM-system, er det især den mere proaktive anvendelse af CRM-systemet, f.eks. i forhold til kampagnestyring og opdyrkning af nye kunder, der interesserer dem. De efterspørger

ét samlet sted til at opbevare alle informationer om deres kunder og kundernes interesser, dels for at lette sælgernes arbejde, og dels for at vide præcis hvilke kunder - reelle eller potentielle - de skal henvende sig til, når nye tilbud kommer frem.

5.3 Hvorfor køber DTI ikke bare et CRM-system?

Før man udvikler eller anskaffer sig et CRM-system, er der forskellige valg, der skal træffes. Først og fremmest skal man beslutte, hvem skal udarbejde CRM-systemet? Er det noget virksomheden selv kan gøre, eller bør man bruge en leverandør udefra?

Mange firmaer tilbyder at sælge CRM-systemer til deres kunder. Man kunne mene, at det ville være oplagt, at en så stor virksomhed som DTI bare køber et system i stedet for at sætte en studerende til at lave et for dem. Umiddelbart er der to grunde til, at de ikke har gjort det.

For det første er DTI's kundedatabase opstået ved 'knopskydning', som de selv kalder det. I begyndelsen havde de blot brug for et sted, hvor de kunne samle oplysningerne om de forskellige kurser, som DTI tilbyder, så de var nemme at finde. Efterhånden som flere og flere firmaer benyttede sig af disse kurser, var det nødvendigt at samle oplysninger om firmaerne, så det hurtigt var til at se, om ansatte fra et givent firma havde været til andre kurser på DTI, om de var lovet rabat osv. Det virkede logisk at lægge dem i det samme databasesystem, da der ikke var grund til at lave en helt ny database bare for det. Langsomt voksede og voksede databasen, indtil den fik den størrelse, den har i dag. Den oprindelige database var alt for lille til, at det kunne betale sig at anskaffe sig et CRM-system lavet udefra, og da den voksede sig så stor, at det måske kunne være en fordel, var det blevet for specifikt og for indarbejdet til, at det kunne betale sig. Ganske vist er CRM-systemer lavet til at kunne trække informationer fra allerede eksisterende databaser (ellers ville det da også kun være nystartede virksomheder, der ville være interesserede i at anskaffe dem), men når DTI allerede har et program der fungerer - omend ikke optimalt - hvorfor så betale for en forbedring?

For det andet ville det være at ganske unødvendigt at anskaffe et præ-fabrikeret CRM-system på nuværende tidspunkt. Det meste af DTI's program virker ganske udmærket, og i hvert fald i første omgang er det kun en lille del af det, der skal forbedres. Der er i øjeblikket ingen grund til at anskaffe et helt nyt program til det i stedet for at videreudvikle det gamle. Især da det så ikke er sikkert, at man kan bibeholde alle de gode sider ved programmet.

Det kan dog være, DTI vil se anderledes på det om nogle år. I 2003 viste undersøgelsen beskrevet i [8], at cirka halvdelen af de virksomheder, der har et CRM-system, selv havde udviklet systemet. Dette tal falder dog, efterhånden som leverandører af CRM-systemer kan gøre deres systemer billigere og mere effektive. Egenudviklingen halter på mange punkter bagefter standardprogrammernes faciliteter, især når det drejer sig om integration mellem CRM-systemet og andre programmer som f.eks. mail-program. Dette er da også et af problemerne med DTI's nuværende program - et problem som de håber, jeg kan finde en brugbar løsning på.

6 Domænebeskrivelse

I dette afsnit beskriver jeg det domæne, projektet er udviklet til. Formålet med en domænebeskrivelse er at sikre, at udviklerne har en grundlæggende forståelse af domænet, som slutbrugerne kan godkende, inden projektet for alvor påbegyndes. Derudover kan en analyse af domænet bekræfte, om der i det hele taget er grundlag for at fortsætte med projektet.

Det følgende er min opfattelse af det miljø, programmet skal bruges i.

6.1 Danmarks Teknologiske Institut - Informatik

DTI er et ganske stort firma med ca. 10 divisioner og over 700 ansatte. DTI tilbyder kurser, konsulentbistand og mange forskellige produkter, alt efter hvilken division, man henvender sig til. Jeg har udelukkende haft kontakt til divisionen for Informatik. Denne division er delt op i to centre: "Center for udvikling" og "Center for undervisning". Som man kan læse af navnene, står centrene for udvikling af nye produkter hhv. diverse kurser. Der er en flydende afgrænsning mellem disse to centre, og de har da også fælles sælgerstab.

I Informatik er der medarbejdere, såkaldte *phonere*, hvis opgave det er at ringe ud til virksomheder enten for at skabe interesse, booke møder eller lignende. Det er som oftest disse phonere, der skaber kontakten mellem potentielle kunder og sælgere. Phonere kan også sælge produkter direkte til kunden, men det sker sjældent, da kunden ofte har brug for flere informationer om produktet, end phoneren kan give.

De forskellige divisioner på DTI tager sig ikke selv af faktureringen af kunder. Det håndteres af et fælles *call center*, som bruges af alle divisionerne. Det er også dem, der tager sig af indgående opkald fra kunder og stiller dem videre til den relevante division.

Selv om enkeltpersoner også kan henvende sig til DTI, er deres kunder fortrinsvis andre firmaer.

6.2 DTI's database

Som tidligere nævnt er DTI's database opstået via knopskydning. Oprindeligt indeholdt den kun få tabeller med informationer om de kurser, DTI tilbyder. Det blev så nødvendigt også at have informationer om de personer, der holdt kurserne, kursusdeltagerne, rabatter, specialordninger osv. På nuværende tidspunkt består databasen af næsten 160 tabeller med informationer om alt lige fra DTI's ansatte og deres ansvarsområder og kompetencer, over informationer om DTI's kunder og kurser til interne marketing- og evalueringsskemaer.

'Benny' er Informatiks intranet og kælenavnet på søgeprogrammet til den database, hvor oplysninger om kunder, kurser, kontaktpersoner, medarbejdere, økonomi m.v. er gemt. Databasen er først og fremmest beregnet til brug af informatikdivisionens sælgere, og er derfor kun fuldt opdateret der.

Det er brugernes eget ansvar at opdatere og vedligeholde informationerne i databasen, hvilket desværre ikke altid er lige effektivt. Dels fordi det ikke er blevet en fast rutine for dem, dels fordi opdateringsprogrammet ikke tjekker for inkonsistens eller gentagelser. Derfor sker det ofte, at den samme virksomhed er registeret flere

gange, hvis der er forskellige stavemåder af deres navn. F.eks. står A.P. Møller registreret som A.P. Møller, A.P. Møller A/S og A.P. Møller Maersk A/S. DTU står som DTU, Danmarks Tekniske Universitet og DTU - [alle mulige forskellige institutter] osv. Dette kan gøre det besværligt at finde lige præcis den virksomhed, man er interesseret i, hvis man ikke har andre oplysninger om den end navnet.

Hvis man ved, *hvordan* man skal søge, er det muligt at finde de oplysninger man har brug for, gennem 'Benny', men det tager tid. Selve søgehastigheden er hurtig nok, men man kan risikere at skulle ind på mange forskellige sider, for at finde alle de oplysninger man har brug for. Det er heller ikke altid logisk, hvor man skal finde de forskellige oplysninger. Hvis man ikke er en erfaren bruger, kan man næsten lige så godt give op på forhånd.

6.3 At få en ny kunde

Selv om det til tider hænder, at firmaer melder sig selv som potentielle kunder ved enten at ringe ind med spørgsmål eller for at søge tilbud hos DTI, sker det oftere, at nye kunder findes ved, at en phoner ringer ud for at høre, om et firma er interesseret i dette eller hint, og hvis det er tilfældet, sørger de for at arrangere et møde mellem sælger og en kontaktperson ved det givne firma.

Hvis kunden efter dette møde stadig er interesseret i at høre mere om DTI's tilbud (og det virker som en reel interesse og ikke bare høflighedsforespørgslser), opretter sælgeren sagen i DTI's database med informationer om firmaet, kontaktperson ved firmaet, en beskrivelse af tilbuddet, dato for næste aktivitet, tilbuddets størrelse og hvilke oplysninger, sælgeren ellers måtte have på dette tidspunkt. De individuelle sider med oplysninger om firma, kontaktperson og tilbuddet kaldes *stamkort*.

Når sælgeren har fået skaffet den nødvendige dokumentation til at fremstille et tilbud, mødes sælger og kontaktpersonen for at blive enige om tilbuddets størrelse.

Hvis kunden siger ja til tilbuddet, bliver en kontrakt udfærdiget. Herefter har sælgeren ikke mere kontakt til kunden. Rabataftaler mm. er blevet indskrevet på stamkortet og ligger derfor i databasen, så DTI's call center ved, hvad der er aftalt og kan håndtere faktureringen.

Herefter er firmaet i DTI's kundedatabase og får automatisk tilsendt kataloger og relevante tilbud.

Sælgeren bliver først involveret igen, næste gang et salg skal gennemføres.

Et sådant salgsforløb - fra kontakt til enten accept eller afslag af et tilbud - kaldes i daglig tale et *projekt*. De enkelte *aktiviteter* i et projekt kan være elementer som e-mail, telefonmøder mv.

6.4 Forecast

Når et nyt projekt bliver oprettet, skal der laves et *forecast* til projektet. Et forecast er en slags prognose over hvilket beløb, sælgeren personligt formoder at få for projektet. Dette ses ud fra tilbuddets størrelse, og hvor stor sansynlighed der er for, at en kontrakt bliver skrevet under. Disse forecasts bliver lavet således, at sælgernes arbejde kan blive analyseret, og instituttet har et gæt om, hvor mange penge de vil tjene en given måned.

Et forecast består af 3 dele: et *worst case estimate* (hvad tror sælgeren, DTI mindst får ud af projektet), et *best case estimate* (hvad tror sælgeren, DTI højst får ud af projektet, hvis alt går glat) og et *forecast* (ordresandsynligheden ganget med tilbuddets størrelse). Best case og forecast skal selvfølgelig helst være det samme.

Ordresandsynligheden bliver målt i procenter. Ved 0% er projektet enten lige startet op, eller sagen er tabt. Ved 100% er kontrakten skrevet under. Procenten stiger afhængig af, hvor langt henne i forløbet man er, og hvor sikkert man regner med det er, at kontrakten bliver skrevet under.

7 Beskrivelse af målgruppen

Når man sætter sig for at lave et produkt, der skal have praktisk anvendelse, er det vigtigt at definere præcist hvem, der skal bruge det, og hvad deres forudsætninger for at bruge produktet er. Ellers kan man nemt ende med at lave noget, der måske er praktisk nok i sig selv, men som viser sig at være ubrugeligt, fordi det er helt forkert for målgruppen.

7.1 Primære målgrupper

Programmet har umiddelbart kun to primære målgrupper:

1. Sælgere ved DTI-Informatik.
2. Analytikere.

Det er dog allerede nu tanken, at 'Benny' efterhånden skal udvides så alle afdelinger på DTI kan benytte oplysningerne. Når det sker, skal dette CRM-system med ganske få ændringer også kunne bruges af sælgere i andre afdelinger af DTI.

7.1.1 Nærmere beskrivelse af målgrupperne

Sælgere i Informatik-afdelingen ved DTI består af alle de personer, der har med salg at gøre. Analytikere benytter sælgernes salgsreportager til at bedømme dels sælgernes arbejde og dels kundernes behov og ønsker.

I det følgende er både sælgere og analytikere opfattet som én gruppe og omtalt som 'salgspersoner'.

Mit indtryk af de salgspersoner, jeg har talt med, er, at de lider under den noget firkantede måde 'Benny' fungerer på nu (som beskrevet i afsnit 6.2, og at de er meget villige til at hjælpe mig med deres idéer og ønsker til forbedringer af programmet.

For salgspersoner er det vigtigt, at det er let at komme til de oplysninger, DTI har om et givent firma, således at de kan være forberedte, når de skal mødes med en kunde fra firmaet og dermed ikke spilder tid med irrelevante tilbud og informationer. De er afhængige af at give kunden et godt indtryk af DTI, så vedkommende vender tilbage til DTI, hvis de senere skulle få brug for andre af DTI's tilbud.

Computererfaringen og -disciplinen er meget varierende mellem salgspersoner. De bruger alle computere til deres arbejde, men nogle benytter kun de programmer og metoder, de kender, hvor andre er mere villige til at eksperimentere lidt. Tilsvarende er det meget forskelligt, hvor pligtopfyldende de er, når oplysninger skal indtastes eller opdateres.

7.2 Sekundære målgrupper

Udover de primære målgrupper findes der også sekundære målgrupper. Sekundære målgrupper er personer, som også vil komme til at benytte programmet, men hvis ønsker og behov, der ikke er blevet taget lige så meget hensyn til under design og implementering. Her kan nævnes:

1. DTI's call-center.

2. Phonere ved DTI's Center for Informatik.

For personerne i begge disse målgrupper gælder det, at de kun har brug for en ganske begrænset del af oplysningerne om kunden og kun på en 'read-only' basis, da de ikke har behov for selv at redigere i oplysningerne.

Som det fremgik af afsnit 6, skal phonere bruge telefonnumre til mulige interesserede. Call-center skal kende til oplysninger om fakturering - f.eks. hvor stort tilbudet er, og om kunden har specielle rabatter eller aftaler med DTI.

8 Fastlæggelse af behov og krav

Inden man går igang med at lave et nyt produkt, er det vigtigt at slå fast præcist hvilken niche, man prøver at udfylde. Det er vigtigt at finde ud af, hvad der i forvejen findes indenfor området, så man ikke bruger tid på at implementere noget, der allerede eksisterer. Det er også vigtigt at se på, hvilke krav målgruppen stiller, så det produkt, man ender med, rent faktisk også stemmer overens med det, slutbrugerne forventer og har brug for.

8.1 Salgspersonernes behov

De salgspersoner, jeg snakkede med, var alle meget enige om, hvilke behov der skulle dækkes, for at de fik nogen glæde ud af et CRM-system.

Først og fremmest er det vigtigt, at det er muligt at finde alle de oplysninger DTI har om en specifik kunde. DTI er interesserede i, at det skal være en god oplevelse for deres kunder at handle med DTI, og derfor skal sælgerne kunne gøre et godt indtryk. Dette inkluderer at vide så meget om kunden og de projekter, kunden er involveret i, som muligt.

Det betyder også, at det skal være let at vedligeholde oplysninger, for at man kan være sikker på, at alt er up-to-date. Dette gælder både informationer om firmaet (kunden), DTI's kontaktpersoner i firmaet og de igangværende projekter. Især omkring projekterne er det vigtigt, at man kan ændre oplysningerne undervejs. I øjeblikket kan man ikke ændre størrelsen på et givet tilbud. I stedet er man nødt til at lukke projektet og så åbne det igen under et andet navn. Det er ikke særlig hensigtsmæssigt. De salgspersoner, jeg snakkede med, syntes det ville være rart, hvis man i stedet kunne ændre oplysningerne og så føre logbog over ændringerne.

Men det er ikke nok bare, at man kan finde alle oplysningerne. Det skal også være muligt at finde dem hurtigt. Hvis man sidder og taler i telefon med en kunde, nytter det ikke noget, at det tager to timer at rode alle internetsiderne igennem, før man finder de oplysninger, man har brug for. Derfor er det vigtigt, at det kun kræver én eller højst to søgninger, før man har, hvad man har brug for.

Når der skal laves rapport over DTI's forventede indtægter, bliver dette gjort på grundlag af de igangværende projekter. Derfor skal det være muligt hurtigt at få et overblik over, hvilke projekter sælgere er involveret i, projekternes forecast og deres status - dvs. om de er aktive eller passive.

8.2 Kravspecifikation

Gennem interviews og gennemgang af 'Benny' og andre CRM-systemer er jeg nået frem til nedenstående krav til programmet. Kravene er blevet mere detaljerede undervejs i processen, efterhånden som jeg fik en klarere idé om, hvad det egentlig var, DTI ønskede. Denne liste er det endelige resultat, som programmet bliver testet funktionelt imod.

8.2.1 Funktionelle krav

Disse krav specificerer, hvad programmet skal kunne.

K 1.1 **Det skal være muligt at søge på et specifikt firma i databasen udfra oplysninger som f.eks. navn, adresse, og/eller telefonnr, og derved få alle de oplysninger DTI har om firmaet. Resultatet af søgningen skal gives på en let overskuelig måde.**
Begrundelse: Det kan være nødvendigt at finde oplysninger om et firma, der ikke har noget med en kontaktperson eller et projekt at gøre.

K 1.2 **Det skal være muligt at søge på en specifik kontaktperson i databasen ud fra oplysninger som f.eks. navn, telefonnummer og/eller titel, og derved få alle de oplysninger DTI har om personen. Resultatet af søgningen skal vises på en let overskuelig måde.**
Begrundelse: Det kan være nødvendigt at søge direkte på kontaktpersonen, for at gøre det både lettere og hurtigere, end hvis man først skal gennem firmastamkortet.

K 1.3 **Det skal være muligt at søge på et specifikt projekt i databasen udfra oplysninger som f.eks. navn, salgsansvarlig og/eller sum, og derved få alle de oplysninger DTI har om projektet. Resultatet af søgningen skal gives på en let overskuelig måde.**
Begrundelse: Det kan være nødvendigt at søge direkte på projektet for at gøre det både lettere og hurtigere, end hvis man først skal gennem firmastamkortet.

K 1.4 **Det skal være muligt at lave en kombineret søgning af firma, kontaktperson og projekt.**
Begrundelse: For at lette søgningen af projekter eller kontaktpersoner kan det være nødvendigt at søge på et firma samtidig, så mængen af resultater af søgningen bliver mere overskuelig.

K 1.5 **Det skal være muligt at oprette et nyt firma i systemet.**
Begrundelse: DTI antages hele tiden at få nye kunder. Hvis systemet ikke kan opdateres, bliver det meget hurtigt ubrugeligt.

K 1.6 **Det skal være muligt at oprette en ny kontaktperson i systemet.**
Begrundelse: Dels antages DTI hele tiden at få nye kunder, dels kan det blive nødvendigt at have flere kontaktpersoner hos samme kunde. Hvis systemet ikke kan opdateres, bliver det meget hurtigt ubrugeligt.

K 1.7 **Det skal være muligt at oprette et nyt projekt i systemet.**
Begrundelse: DTI antages hele tiden at få nye aftaler med kunder. Hvis systemet ikke kan opdateres, bliver det meget hurtigt ubrugeligt.

K 1.8 **Det skal være muligt at redigere oplysningerne i systemet.**
Begrundelse: Man kan ikke antage, at sælgeren dels har alle oplysningerne fra begyndelsen og dels indtaster alt rigtigt hver gang. Derfor skal det være muligt at redigere oplysningerne, så man ikke er nødt til at starte helt forfra, hver gang man opdager en fejl eller skal opdatere nogle oplysninger.

K 1.9 **Det skal være muligt at finde alle de forecasts, en sælger har oplyst.**
Begrundelse: Til DTI's interne rapporter er det nødvendigt at vide, hvor mange projekter en given sælger er igang med, og hvad deres forecasts er.

K 1.10 **Programmet skal være integreret med et email-program (f.eks. Outlook).**
Begrundelse: Eftersom firmaers og kontaktpersoners email kan findes i programmet, vil det lette arbejdet væsentligt, hvis man kan sende email direkte fra programmet i stedet for først at være nødt til at åbne et nyt program.

8.2.2 Data krav

Disse krav specificerer de krævede data, d.v.s. type, størrelse, nøjagtighed osv.

K 2.1 **Input til databasen skal være af den korrekte type.**
Begrundelse: For at undgå fejl når man skriver til eller læser fra databasen, er det vigtigt, at de forskellige informationer er af den type databasen forventer. De forskellige typer kan ses i tabel 2 i appendiks D.

8.2.3 Brugeroplevelseskrav

Disse krav omhandler den oplevelse brugeren får under brug af programmet.

K 3.1 **Programmet skal være enkelt og intuitivt at bruge.**
Begrundelse: Det er meningen at programmet skal lette brugerens arbejde, ikke gøre det mere besværligt.

K 3.2 **Hvor der kan være tvivlspørgsmål skal programmet give konstruktive oplysninger eller fejlbeskeder.**
Begrundelse: Se ovenfor.

K 3.3 **Det skal være let og hurtigt at opdatere informationerne.**
Begrundelse: Dette vil gøre det lettere at holde databasen up-to-date og dermed brugbar.

8.2.4 Brugerkrav

Disse krav omhandler hvordan programmet skal tilpasses målgrupperne.

K 4.1 **Programmets grænseflade skal være opdelt i få skærmbilleder.**
Begrundelse: Ud fra samtaler med slutbrugerne er det blevet klart, at de er mest interesserede i få skærmbilleder med mange oplysninger, end mange skærmbilleder med få oplysninger, da hurtig navigation betyder mere for dem end simple skærmbilleder.

K 4.2 **Navigationen skal være simpel.**
Begrundelse: Programmet kan antages ofte at skulle bruges, mens man sidder og snakker med en kunde, derfor er det nødvendigt, at man hurtigt kan finde frem til de relevante oplysninger.

8.2.5 Brugervenlighedskrav

Disse krav specificerer brugervenlighedsmålene og de tilhørende målemetoder.

K 5.1 **Sælgerne skal efter ca. 20 minutters arbejde med programmet kende det så godt, at de fremover kan bruge det hurtigt og effektivt.**
Begrundelse: Hvis det kræver længere tid at sætte sig ind i, kunne man forestille sig, at sælgerne ville foretrække at blive ved noget, de kendte.

9 Resultat

I det følgende beskrives det endelige produkt. Samtidig beskrives de overvejelser og diskussioner, jeg har haft undervejs i processen. Dette afsnit kan bruges til at få et overblik over det endelige produkt og forstå hvorfor det ser ud, som det gør.

Afsnittet er placeret her, før jeg begynder at gå i detaljer med analyse og design, fordi jeg mener, det er vigtigt, at læseren har et indtryk af det færdige program, inden de mere tekniske detaljer bliver beskrevet.

Programmet er bygget op af fem sider eller webforms (i resten af rapporten vil begge ord blive brugt) skrevet ved brug af ASP.NET. Disse sider består af en beskrivelse af sidens udseende ('side.aspx') og den bagvedliggende kode i C# ('side.aspx.cs').

9.1 Distribution og installation

Da programmet skal være let tilgængeligt for alle på DTI's Center for Informatik, vil det være oplagt at lægge det på den server, hvor 'Benny' og den oprindelige database er placeret. Her vil det også være muligt at finde en brugervejledning til programmet. Jeg har tilføjet nogle få tabeller til databasen, som også skal inkluderes i den oprindelige database.

Programmet kræver ingen installation og er derfor klar til brug, når det er blevet lagt på serveren. For at gøre det lettere at finde, kunne man eventuelt tilføje et link til det fra 'Benny', men det er op til DTI's webmaster, om han vil gøre det. Alternativet er blot, at man skal kende den direkte adresse.

9.2 Programmet

Det var vigtigt for mig, at programmet blev så brugervenligt som muligt, så slutbrugerne kunne anvende det med ingen eller kun ganske kort instruktion. Det skal ikke være nødvendigt at kunne huske fremgangsmåden fra gang til gang, da det bør være muligt at tænke sig frem til, hvilke funktioner man skal benytte. For at sikre denne brugervenlighed har jeg valgt at anvende Gestaltlovene [4, s.108], som er en standard-metode inden for interaktionsdesign.

Gestaltlovene beskæftiger sig med, hvordan vi opfatter helheder i billeder. På en brugergrænseflade er det vigtigt, at elementer på siden hænger sammen både visuelt og logisk, således at tekst og knapper, der hører sammen, også sidder sammen. De vigtigste love er loven om *nærhed*, loven om *lukkethed*, loven om *lighed* og loven om *forbundethed*. Jeg har forsøgt at overholde disse love ved at sørge for, at siderne ligner hinanden, ved at placere beslægtede elementer sammen og ved at benytte rammer omkring relaterede elementer og således adskille forskellige elementer fra hinanden. Det sidste var især vigtigt på forsiden, som ellers kunne være blevet ret uoverskuelig, se evt. figur 9 i afsnit 10.

Brugervenlighed er dog mere end blot udseendet. Også i funktionaliteten har jeg forsøgt at gøre programmet så let at bruge som muligt. Dette indebærer bl.a. at lave konstruktive fejlmeddelelser, hvis brugeren alligevel skulle komme til at gøre noget forkert, og at søgefunktionen er fleksibel. Først og fremmest er søgningen ikke 'case sensitive', og det er derfor ligemeget, om man bruger små eller store bogstaver. Desuden er søgefunktionen konstrueret, så den tager højde for, at nogle navne kan

staves både med 'aa' og med 'å' og søger derfor efter begge dele. Slutteligt er det muligt kun at søge på en del af et navn, da der automatisk tilføjes wildcards før og efter de fleste søgeord, så en søgning på en vilkårlig delmængde af f.eks. 'Maria Miland Elmvang' (hvor bogstaver og mellemrum stadig er i samme rækkefølge) vil give det rigtige resultat.

`Login.aspx`

Da programmet tillader brugeren at ændre i DTI's database, er det vigtigt at sørge for, man har styr på, hvem der benytter programmet. Login-siden sammenligner brugernavn og password og sikrer, at brugeren er blandt de ansatte ved DTI's Center for Informatik.

Hvis brugernavn og password accpteres, bliver man sendt videre til forsiden. Ellers får man at vide, at man ikke har videre adgang til systemet.

Når man er blevet logget på, bliver ens brugernavn gemt i en "Session" og overført til alle de andre sider. Det bruges når man opretter eller redigerer oplysninger i databasen, hvor et tidsstempel med navn bliver gemt i databasen. En "Session" er kun gyldig i et begrænset tidsrum (bestemt af serveren - det varer typisk 20 minutter). Når ens session er udløbet, skal man logge på igen, før man må gå videre. Dette er gjort for at undgå, at uvedkommende kan bruge programmet, hvis man har glemt at lukke det, når man går fra computeren.

`Forside.aspx`

Forsiden er tænkt som en genvej til de forskellige søgesider. Som på søgesiderne kan man søge adskilt på firma, personer og projekter, men det er også muligt at lave kombinerede søgninger og for et givet firma se, hvilke kontaktpersoner DTI har i det firma, eller hvilke projekter, der er registreret for firmaet, alt afhængig af hvilken søgeknap, man trykker på. Søgefelterne er indrammet, jfr. loven om lukkethed.

Når man har foretaget en søgning, bliver resultaterne vist i en liste opdelt i firmaer, personer og projekter. Fra denne liste kan man vælge det resultat man søgte efter, ved at trykke på *Vælg denne*. Uddybende oplysninger om firmaet, personen eller projektet bliver så vist på den relevante side.

Se evt. appendiks B for eksempler på de forskellige tabeller med resultater.

`Forside.aspx` læser kun fra databasetabellerne. Man kan hverken bruge den til at oprette eller redigere i dem.

`Firma.aspx`, `Person.aspx` og `Projekt.aspx`

Da disse tre sider følger den samme overordnede struktur, bliver de beskrevet under ét. For nemheds skyld refererer jeg til `Firma.aspx`, men med mindre andet er sagt, gælder beskrivelsen også for `Person.aspx` og `Projekt.aspx`.

`Firma.aspx` opfylder slutbrugerens ønske om så få sider som muligt, da den indeholder tre forskellige funktionaliteter. I stedet for at have tre forskellige sider til at søge efter, oprette og redigere firmaer er dette slået sammen på én side, hvorfra alle tre ting er muligt. Dette kan måske give lidt forvirring ved, at ikke alle felter er

søgefelter, at man ikke kan rette alle oplysninger, og at nogle felter skal efterlades tomme, når man opretter et firma. Efter at have talt med slutbrugerne, mener de dog, at den overskuelighed, man får ved kun at have én side, vil opveje de mulige ulemper. For de fleste vil det være klart at se, hvilke felter man skal udfylde, og skulle man alligevel komme til at begå en fejl, giver systemet konstruktive fejlmeddelelser, så den hurtigt kan rettes.

På både `Firma.aspx` og `Person.aspx` bliver firmaets/kontaktpersonens eventuelle projekter vist. Det er således muligt at vælge et givet projekt fra denne liste og få vist projekt-stamkortet med de relevante oplysninger.

For at sikre brugeren imod gentagelser i databasen når nye elementer oprettes, laver programmet et tjek af databasen, inden elementet oprettes. For firmaer og personer bliver dette tjek *ikke* lavet på baggrund af navnet, da man i så fald ikke ville fange forskellige stavemåder af samme navn - som f.eks. A.P. Møller/AP. Møller. - eller eventuel brug af mellemnavne. I stedet bliver firmaer sammenlignet vha. adresse og telefonnummer og personer vha. firma og email-adresse. Jeg er godt klar over, at dette betyder, man kan risikere gentagelser, hvis en person flytter fra et firma til et andet, men regner med, at det sker tilpas sjældent til ikke at være et problem. Desuden ville det være vanskeligt at have noget sammenligningsgrundlag overhovedet ved flytning, da man i så fald ikke kan regne med at andre informationer end netop navnet bibeholdes.

Ved projekter har jeg valgt at bruge projektnavnet og firmaet som sammenligningsgrundlag, da det kun er den aktuelle salgsansvarlige, der ville oprette projektet, og der er derfor ikke de store farer for gentagelser. På denne måde undgår man også, at der ved et firma kan være to forskellige projekter med samme navn.

`Forecast.aspx`

`Forecast.aspx`'s funktionalitet er helt anderledes end de fire andre sider og en helt ny funktionalitet i forhold til 'Benny', som ikke beskæftiger sig med projekters forecast overhovedet. I stedet for at holde styr på DTI's eksterne oplysninger, omhandler denne side DTI's interne oplysninger. Dvs. oplysninger om de forskellige sælgeres arbejde og deres forventede forecasts. Sælgerne kan benytte denne side til at sikre sig, at de har fået opdateret alle deres projekter. Siden er dog mest tænkt som hjælp til salgsanalytikernes arbejde, da de kan bruge den til hurtigt at få et overblik over, hvilke projekter DTI er involveret i og deres ordresansynlighed. Dermed kan de lave en rapport over sælgernes arbejde, og hvor mange penge DTI kan forvente at få fra kunder i en given måned.

Det er på denne side muligt at søge på en given sælger og få en liste over de projekter sælgeren er involveret i, tilbudets størrelse, oplysninger om forecast mv. Ved hvert enkelt projekt kan man trykke på *Vælg denne* for at blive dirigeret til `Projekt.aspx` og få vist oplysningerne om projektet.

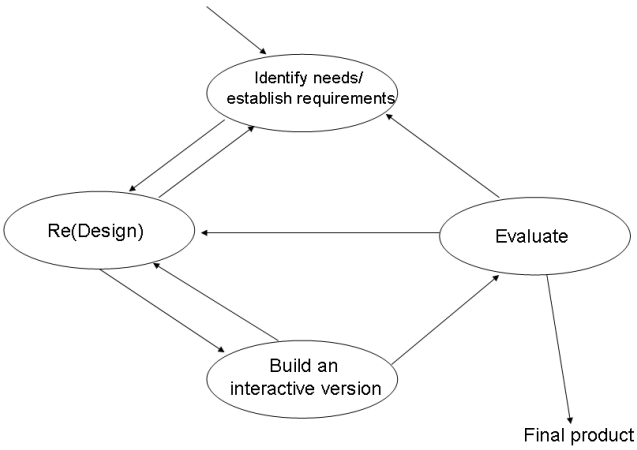
10 Arbejdsproces

I løbet af projektet har jeg anvendt forskellige metoder og teknikker fra forskellige grene af softwareudvikling for at opnå de mål, jeg har sat for resultatet af projektet. I det følgende gør jeg rede for disse, og hvordan jeg overordnet har brugt dem i projektet.

10.1 Beskrivelse af procesmodel

Da udviklingen af programmet kræver meget interaktion med de kommende brugere af systemet, virkede det mest naturligt at benytte den simple interaktionsdesign-model [7, s. 186] som procesmodel. Men da det samtidig er et software udviklings-projekt, har jeg valgt at udvide den simple interaktionsdesign model med ting fra de gængse softwareudviklingsmodeller, se f.eks. [10].

Den simple interaktionsdesign model har den fordel, at den bygger på den formodning, at man er nødt til at arbejde i iterationer. De fleste softwareudviklingsmodeller er såkaldte 'vandfaldsmodeller', som antager, at når man er færdig med et trin i modellen (f.eks. analyse eller design), så har man gjort det så godt, at det ikke er nødvendigt at vende tilbage til dette trin. Det er min erfaring, at denne tankegang er håbløst optimistisk, især når man som i dette tilfælde arbejder på et projekt, hvor man hele tiden skal have slutbrugerne for øje.

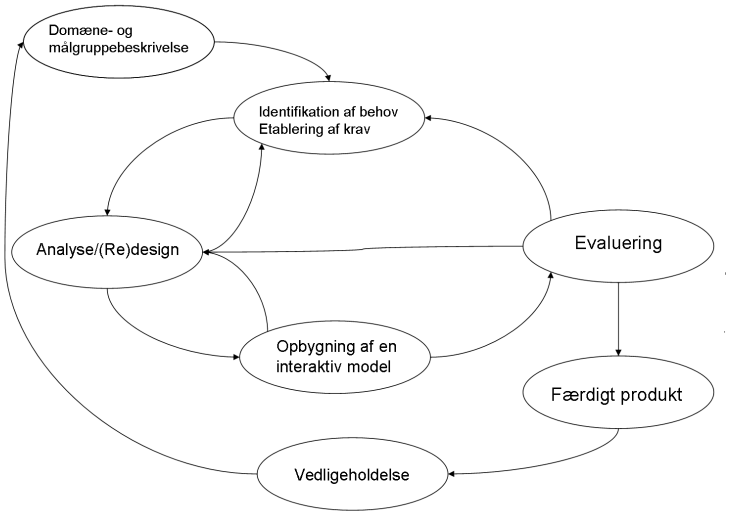


Figur 3: Original Processmodel.

På den anden side er den simple interaktionsdesign model på mange punkter *for* simpel. Den simple procesmodel består den kun af fem punkter, nemlig "Identifikation af behov. Etablering af krav", "Analyse/(Re)design", "Opbygning af en interaktiv model", "Evaluering" og "Færdigt produkt", se figur 3. På grund af omfanget af mit projekt fandt jeg det nødvendigt at medtage de ekstra punkter "Domæne- og målgruppebeskrivelse" og "Vedligeholdelse" fra softwareudviklingsmodeller. Især

det første er essentielt i et projekt at denne størrelse, da det er vigtigt, man kender de omgivelser slutproduktet skal benyttes i, for at være sikker på at det man laver, stemmer overens, med det slutbrugerne har brug for.

Mine overvejelser har resulteret i procesmodellen i figur 4. Jeg starter med "Domæne- og målgruppebeskrivelse", bevæger mig rundt i modellen som det måtte vise sig nødvendigt, for at slutte med et "Færdigt produkt". "Vedligeholdelse" berører jeg kun overfladisk i afsnit 15.



Figur 4: Processmodel.

For at få en bedre forståelse for, hvad de forskellige punkter dækker over, lavede jeg en brainstorm baseret på hvad jeg vidste om softwareudvikling. I denne brainstorm koncentrerede jeg mig om hvordan de forskellige punkter kunne uddybes, og hvilke observationer eller analyser der var nødvendige at foretage, for at opnå så godt et resultat som muligt. Inspiration til brainstormen fik jeg især fra [4] og [7], da disse to bøger lægger stor vægt på brugervenligheden i et program. [10] var også meget anvendelig, da den beskriver et projektforløb i sin helhed - fra domænebeskrivelse til test.

Min brainstorm resulterede i, at de forskellige punkter i figur 4 udspecificeres som beskrevet i det følgende. Det er ikke alle elementerne, der bliver berørt i alle iterationer af modellen.

Generelt kan det siges, at jeg gennem hele forløbet har lagt stor vægt på participatory design - dvs. at få brugerne til at være med til at designe programmet. Eftersom jeg ikke har nogen pratisk erfaring med at bruge 'Benny', har det været nødvendigt at drage så megen nytte af brugernes erfaringer som muligt, og derfra få idéer til programmet, finde ud af hvad der var nødvendigt, hvad der ville være overflødigt, og hvad der ville være en lille ekstra, rar ting at have med. Det ville slet ikke have været muligt for mig at lave dette projekt uden at inddrage slutbrugerne, hvorfor participatory design var et af de elementer, jeg lagde ekstra meget vægt på.

- **Domæne- og målgruppebeskrivelse**
  - Studier af dokumentation om det eksisterende system
  - Observation af brugere
  - Interviews
- **Identifikation af behov/etablering af krav**
  - Studier af faglitteratur
  - Observation af brugere
  - Interviews
  - Beskrivelse af målgrupper således at det bliver tydeligt hvilke målgrupper, der stiller hvilke krav
  - Krav der skal overvejes:
    * Funktionelle krav - krav om programmets funktionalitet
    * Brugeroplevelseskrav - krav om hvilken oplevelse brugeren får ud af at benytte programmet
    * Brugerkrav - krav om hvordan programmet skal tilpasse sig brugerne
    * Brugervenlighedskrav - krav om hvor enkelt programmet skal være at bruge
- **Analyse/(Re)design**
  - Skitser
  - UML-diagrammer [1]
  - Participatory design [7, s. 306] - få slutbrugerne til at komme med idéer og kritik i designfasen
  - Overvej og hvis muligt benyt:
    * Designprincipper [7, s. 20] - hvordan systemet skal se ud og reagere når det bruges. De mest brugte er *synlighed* (at de nødvendige funktioner er lette at finde), *feedback* (at brugeren bliver gjort klar over, at systemet har registreret en given handling), *begrænsninger* (funktioner, der ikke er mulige at benytte i den givne situation bliver deaktiveret) og *konsistens* (at layout'et er designet som brugeren forventer og er vant til).
    * Gestaltlovene [4, s.108]

    Da disse supplerer hinanden snarere end at være i modstrid, er det muligt at benytte begge dele.
- **Opbygning af en interaktiv model**
  - Prototyper: low-fidelity → high-fidelity - fra simple til teknisk avancerede prototyper
  - Design af prototype baseret på design:

    * Skitser
    * PowerPoint præsentationer
    * Implementering i ASP.NET med C#

- **Evaluering**
  - DECIDE [7, s. 348] - et framework der bruges til at analysere og guide evalueringer. DECIDE giver den følgende (engelske) tjekliste til evalueringer:
    * Determine the overall goals that the evaluation addresses
    * Explore the specific questions to be answered
    * Choose the evaluation paradigm and techniques to answer the question
    * Identify the practical issues that must be addressed
    * Decide how to deal with the ethical issues
    * Evaluate, interpret and present the data
  - Heuristikinspektion [4, s. 152]- se på programmet med brugerens øjne, for at se om designprincipperne er blevet overholdt. En heuristik er en anerkendt regel for konstruktion af gode brugergrænseflader. Dvs. sæt brugeren i centrum, gør brugerens muligheder synlige og vær hjælpsom når brugeren har problemer.
  - Interviews
  - Tænke-højt test [4, s. 122] - brugere anvender programmet mens der tænkes højt for at teste brugervenligheden
  - Funktionel test - test af om programmet opfylder kravspecifikationen
  - Strukturel test - test af alle grene i programmet

- **Vedligeholdelse**
  - Fejlretning
  - Videreudvikling - hvilket igen kan ske ved hjælp af ovenstående procesmodel

Det følgende er en beskrivelse af arbejdsprocessen under hele projektforløbet. Der er en klar sammenhæng mellem denne beskrivelse og procesmodellen beskrevet ovenfor.

10.2 Indledende manøvrer

Eftersom DTI udelukkende benytter sig af Windows maskiner, så de ingen grund til, at CRM-systemet skulle være platform-uafhængigt. Jeg blev derfor bedt om skrive det i ASP.NET med brug af C#, da det så ville være lettere at udvide senere. Jeg havde ingen erfaring med hverken ASP.NET, HTML (som ASP.NET i vid udstrækning baseres på) eller C#, så i begyndelsen blev meget af min tid brugt på at lære

disse sprog. Det passede meget godt rent planlægningsmæssigt, da jeg alligevel ikke kunne komme ordentligt i gang med selve analysen og designet af opgaven, før jeg havde fået snakket med nogle potentielle slutbrugere af programmet.

Programmet skal i stor udstrækning baseres på det allerede eksisterende program, 'Benny'. Dette gav både fordele og ulemper i mit videre arbejde. Fordele, fordi jeg så kunne få inspiration fra programmet. Ulemper, fordi det også er muligt at få for *meget* inspiration. Det kan være svært at tænke nyt, når man allerede er blevet præsenteret for en mulig løsning.

Heldigvis fik jeg megen hjælp fra de brugere, jeg interviewede i denne henseende. Af gode grunde kendte de 'Benny' væsentligt bedre, end jeg gjorde, og havde helt på det rene hvilke elementer der virkede godt, og hvilke jeg endelig skulle ændre. I følge dem har 'Benny' mange gode features, men problemet er, at de er for besværlige at bruge. Det er muligt at fremskaffe alle de oplysninger, man har brug for om en kunde, men det er ikke logisk, hvor man skal lede efter dem. Hvis man f.eks. søger efter et firma, får man en side med firmaets navn, adresse og telefonnummer. Navnet er et link til kontaktpersonen, telefonnummeret er et link til firmaets NACE-kode (en kode der beskriver hvilken branche firmaet er i) osv. Oplysningerne er der, men man kan ikke tænke sig frem til, hvor man skal kigge efter dem, man er nødt til at vide det. Noget af det, de syntes, var vigtigst at få med i et nyt program, var en mere logisk brugerflade og et mere logisk søgesystem.

10.3 Domæne- og målgruppebeskrivelse

Min første reelle opgave i projektet var at lære så meget som muligt om domænet. Altså finde ud af hvad der menes med et Customer-Relationship Management system generelt, og mere specifikt hvad DTI forstår ved det. Jeg studerede al den dokumentation og faglitteratur, jeg kunne komme i nærheden af for at få en grundlæggende viden om området. Dette gav mig indsigt nok til at lave kvalificerede interviews af slutbrugerne.

Derudover var det også nødvendigt at vide, i hvilken sammenhæng mit program skulle bruges, så jeg havde mulighed for at vide, hvilke funktionaliteter det var vigtigst at få implementeret.

Gorm Bjerre er den egentlige webmaster for 'Benny'. Han var så venlig at mødes med mig og give mig en grundig forklaring af, hvordan 'Benny' fungerer. Jeg troede oprindeligt at 'Benny' var selve kundedatabasen, men det viser sig at 'Benny' er et søgeprogram som håndtere dels den database, jeg skal bruge, dels en anden database, som har med det rent økonomiske aspekt af DTI's arbejde at gøre. Gennem vores samtale fik jeg større forståelse for, hvad det egentlig var, de forventede af mig, når jeg skulle lave et nyt søgeprogram. Det hjalp mig også til at finde ud af, hvilket domæne programmet skal bruges i, og hvem der var mine primære målgrupper.

Resultatet kan ses i afsnit 6 og afsnit 7.

10.4 1. iteration

I løbet af den første iteration blev opgaven defineret nærmere, og jeg prøvede at komme med forslag til en mulig løsning af projektet.

Identifikation af behov / etablering af krav

For overhovedet at komme i gang med projektet var det nødvendigt at vide mere om dels hvad 'Benny' bliver brugt til i øjeblikket, og dels hvilke tanker forskellige brugere af 'Benny' havde gjort sig om en mulig forbedring.

Det kom hurtigt frem gennem samtalerne, at der var mange ting brugerne savnede ved det nuværende system. Især lod struktureringen af oplysninger meget tilbage at ønske. Nogle steder kunne de samme oplysninger findes på mange forskellige sider, andre steder var det nødvendigt at gå ind på 3-4 forskellige sider for at finde f.eks. alle oplysninger om et givet firma. Det var også for besværligt at opdatere databasen med nye informationer, da man i nogle tilfælde ikke kunne gøre det direkte gennem 'Benny', men var nødt til at åbne et nyt program for at ændre oplysningerne.

Igennem interviews og samtaler med ansatte på DTI fik jeg fastlagt de aktuelle behov og hvilke krav, mit program skulle opfylde for at være brugbart for dem. De endelige krav kan ses i afsnit 8.2.

Analyse/design

På baggrund af de krav og behov jeg havde fået identificeret, lavede jeg en brainstorm for at få et overblik over hvilke idéer, jeg havde til projektet. Der var stadig en masse spørgsmål, jeg skulle have afklaret, men nu var jeg i det mindste i stand til at lave overfladiske use case diagrammer. De endelige diagrammer samt beskrivelser af de forskellige funktionaliteter kan ses i afsnit 11.

Opbygning af en interaktiv model

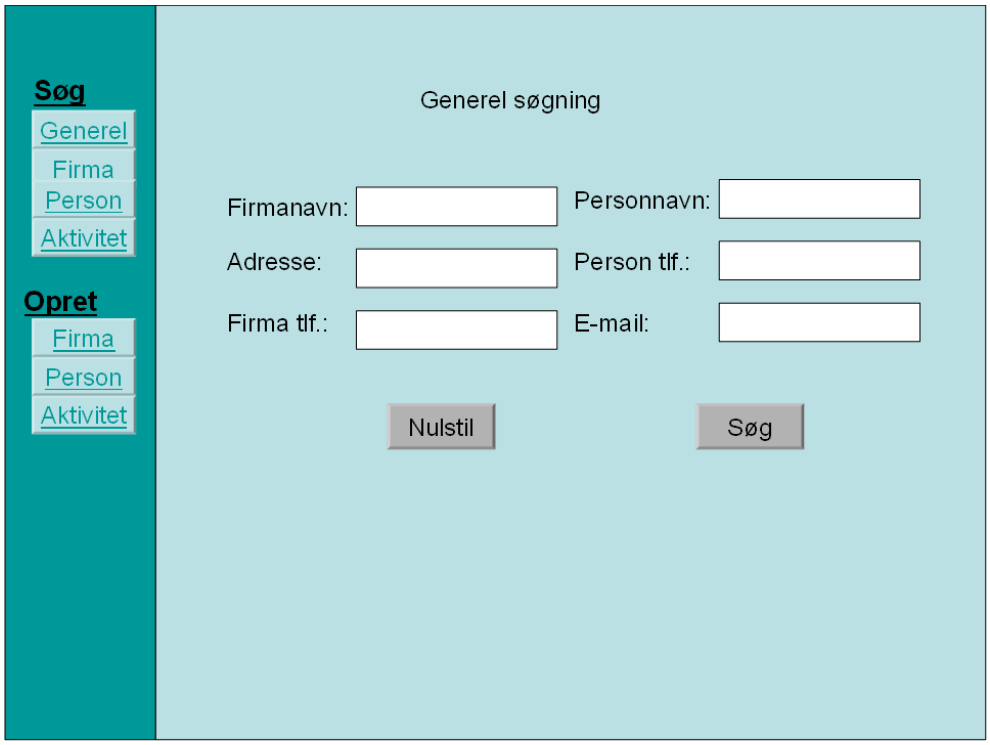
For bedre at kunne visualisere de tanker, jeg havde gjort mig, og for at have noget konkret at basere mine interviews på, lavede jeg nogle simple skitser af forslag til de forskellige skærmbilleder, dvs. en forside til generel søgning, specifik søgning på firma, person eller projekt og sider til at oprette/redigere informationer om firma, person eller projekt. På baggrund af disse skitser lavede jeg en elektronisk interaktiv model af skærmbillederne i MS PowerPoint, som kunne afprøves og evalueres af mine kontakter på DTI. Min væsentligste grund til straks at gå væk fra den low-fidelity prototype, som en papirversion er, var, at mine kontakter sad både i Århus og Taastrup, og at lave prototypen elektronisk var den simpleste måde at få den sendt ud til alle involverede. Det krævede lidt mere tid i første omgang, men var det værd i det lange løb, da jeg kunne få flere mennesker til at evaluere prototypen.

Skærmbillederne var meget ens i opbygning. To udvalgte eksempler på 'Søg' hhv. 'Opret' kan ses i figur 5 og figur 6.

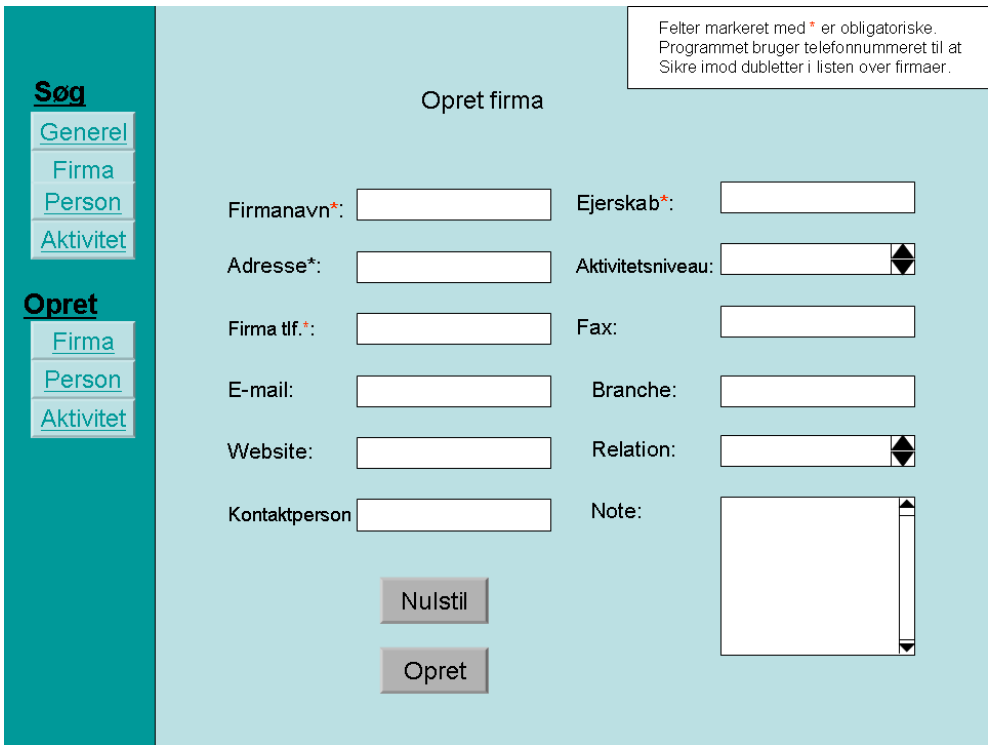
Skærmbillederne er stort set selvforklarende. Knapperne til venstre i billedet styrer navigationen, 'Søg' finder resultaterne af de søgekriterier, man har indtastet, 'Opret' opretter et nyt firma med de indtastede oplysninger, og 'Nulstil' fjerner al tekst fra felterne.

Evaluering

Jeg lavede en intern evaluering af skærmbillederne for at sikre mig, at de ikke var for rodede, og jeg benyttede mig af heuristikinspektion for at være sikker på, at jeg havde



Figur 5: En prototype af forsiden.



Figur 6: En prototype af at oprette firmaer.

overholdt de generelle designprincipper, der skulle gøre siderne mere brugervenlige, som f.eks. Gestaltlovene. Umiddelbart så det sådan ud. Lovene om lukkethed og nærhed var blevet overholdt i og med, at siden var delt op i en navigationsdel til venstre og en informationsdel til højre, og knapper, der var relaterede, sad nær ved hinanden.

Ved et videomøde fremlagde jeg mine forslag for Gorm Bjerre og to sælgere ved DTI, Susanne Christoph og Margit Limkilde Bloch, for at høre, om jeg var på rette spor.

Reaktionerne var generelt positive, dog var der forslag til rettelser af ordvalg og layout. De dokumenter, jeg havde læst, brugte en anden terminologi, end sælgerne på DTI gjorde, og det var vigtigt at få ændret ordene, så alle brugerne ville vide, hvad der blev snakket om. F.eks. havde jeg for ethvert firma bedt om, at et 'ejerskab' blev indtastet, så man vidste, hvem der først og fremmest havde kontakt til et specifikt firma. De andre forstod først ikke, hvad jeg mente, for de kalder den person en 'salgsansvarlig'. Tilsvarende blev 'aktivitet' ændret til 'projekt' og forskellige andre småretteIser blev foreslået.

10.5 2. iteration

I løbet af anden iteration blev projektet konkretiseret, efterhånden som jeg fik feedback fra flere sælgere ved DTI.

Identifikation af behov/etablering af krav

Ved videomødet fik jeg mere præcist at vide, hvad folk forventede af et søgesystem til deres kundedatabase. Rent funktionelt var der ikke så meget at være i tvivl om. Man skulle kunne søge på et firma, en kontaktperson eller et projekt samt oprette disse og ændre oplysninger allerede gemt i databasen. For at gøre det så intuitivt og effektivt at bruge som muligt, var der dog visse behov til layout'et, som jeg skulle sørge for at opfylde. Dette bliver beskrevet i afsnit 12.1.

Derudover hjalp en af sælgerne mig ved at skrive en meget detaljeret email om hendes forventninger til programmet, se appendiks F.

Analyse/Re-design

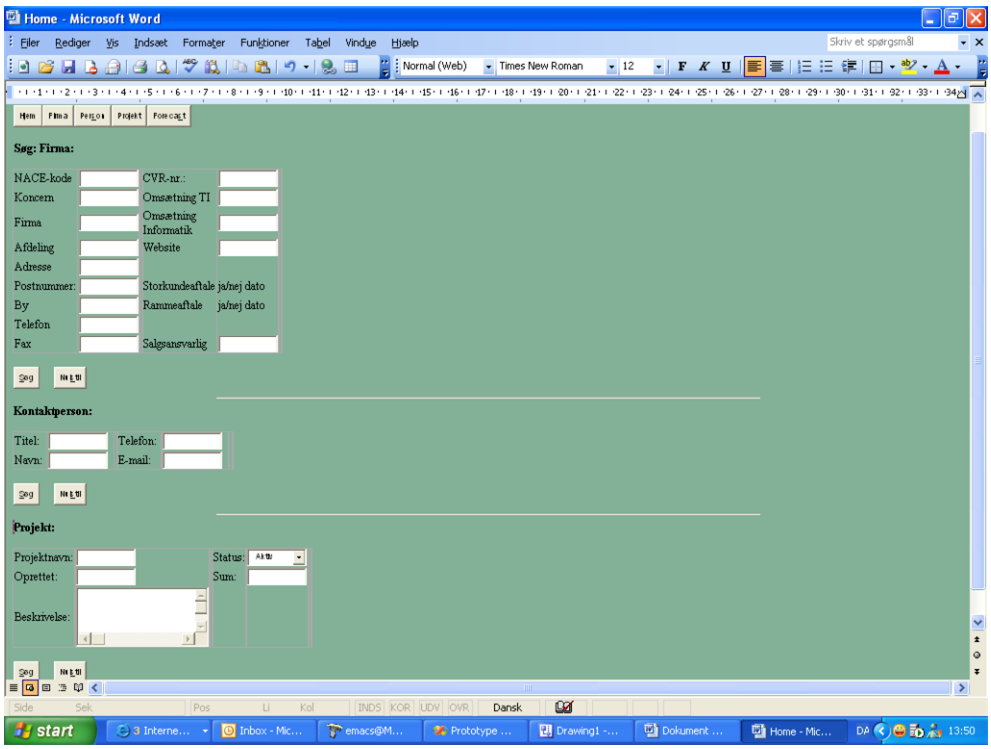
Ud fra de forslag, der var kommet frem ved videomødet og anden feedback, kunne jeg redesigne de forskellige brugergrænseflader til at indeholde de funktionaliteter og det layout, som sælgerne ønskede. Det indebar blandt andet at samle flere oplysninger på samme skærmbillede, så navigationen blev gjort hurtigere, og at ændre nogle af de ordvalg jeg oprindeligt havde truffet.

Opbygning af en interaktiv model

På baggrund af de nye oplysninger jeg havde fået, lavede jeg en ny version af skærmbillederne, for at være sikker på at jeg havde forstået sælgerne rigtigt.

Forsiden blev den, der ændrede sig mest drastisk i udseendet, da sælgerne var enige om, at det ville være rart med flere søgemuligheder allerede der. Det er så

meningen at man kun kan *søge* på forsiden. Vil man oprette eller redigere informationer, skal man bruge de andre sider, hvor man også kan foretage mere detaljerede søgninger. Det nye forslag til en forside kan ses i figur 7.



Figur 7: Andet udkast til en mulig forside.

På baggrund af deres ønsker om så få forskellige skærmbilleder som muligt, besluttede jeg at 'søg', 'opret' og 'rediger' skulle foregå udfra samme skærmbillede i stedet for fra tre forskellige skærmbilleder som først tænkt. Jeg var dermed gået fra 11 sider til fem.

Programmet består altså af fem sider, opdelt efter funktionalitet. De fem sider hedder "Forside", "Firma", "Person", "Projekt" og "Forecast". Jeg har valgt at benytte ordet 'Person' i stedet for 'Kontaktperson', da man kan bruge denne side til både at søge efter kontaktpersoner blandt DTI's kunder og efter ansatte hos DTI selv.

Da der ikke var så meget mere at gøre ved selve designet, gik jeg i gang med noget af implementeringen, da programmet indeholder mange småfunktioner, der godt kan implementeres uafhængigt af både hinanden og brugergrænsefladen.

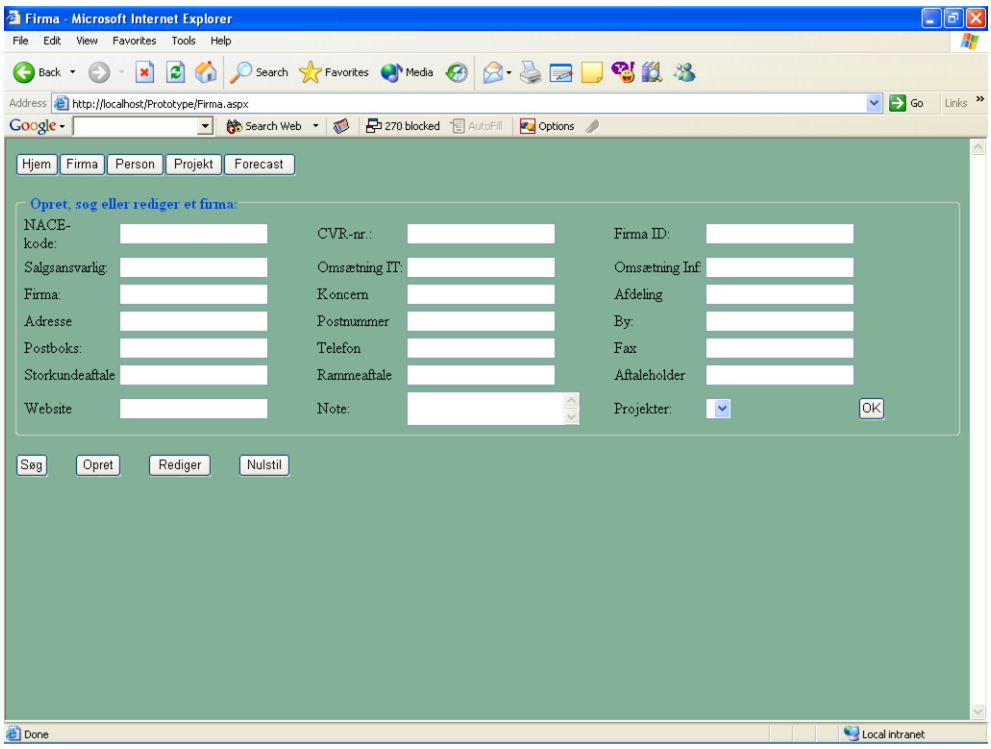
Programmet skal bruges af en meget stor database, derfor valgte jeg i første omgang at konstruere en lille, urelateret 'prøve-database', som jeg kunne koncentrere mig om. På denne måde kunne jeg sørge for, at funktionerne og især mine SQL-statements virkede som forventet, før jeg brugte ressourcer på at indsamle informationer fra hele databasen.

Det viste sig at være en god idé, da Visual Studio.NET's fejlmeddelelser ikke er så brugervenlige, som man kunne ønske sig. Det var derfor nødvendigt at have et overblik over hele databasen, for at kunne finde frem til, hvad der gik galt hvor under implementeringen.

Derudover var det væsentligt lettere at bekræfte, at de udførte SQL-statements returnerede de rigtige værdier, når jeg kendte databasens indhold og derfor vidste, præcis hvad de forskellige statements burde returnere, og ikke blot *at* de burde returnere noget. På denne måde kunne jeg bevare overblikket over resultaterne.

Evaluering

Jeg aftalte et møde med Susanne Christoph og viste hende de nye skitser af skærmbillederne, jeg havde lavet. Hun var tilfreds med de ændringer, jeg havde lavet, og havde kun ganske få rettelser (udelukkende ændringer i syntaks). Især var hun glad for, at 'søg', 'opret' og 'rediger' var samlet på en side, fordi dette betød at hvis man opdagede en fejl i en søgning, var man ikke nødt til at skifte skærmbillede, for at ændre oplysningen.



Figur 8: `Firma.aspx` med 'søg', 'opret' og 'rediger'.

10.6 3. iteration

Jeg var nu nået så langt, at jeg skulle have kontakt til Gorm Bjerre for at få adgang til en 'sandkasse' på 'Benny', således at jeg kunne begynde at implementere CRM-systemet, uden at det ville påvirke de oplysninger, andre på DTI har adgang til.

Identifikation af behov/etablering af krav

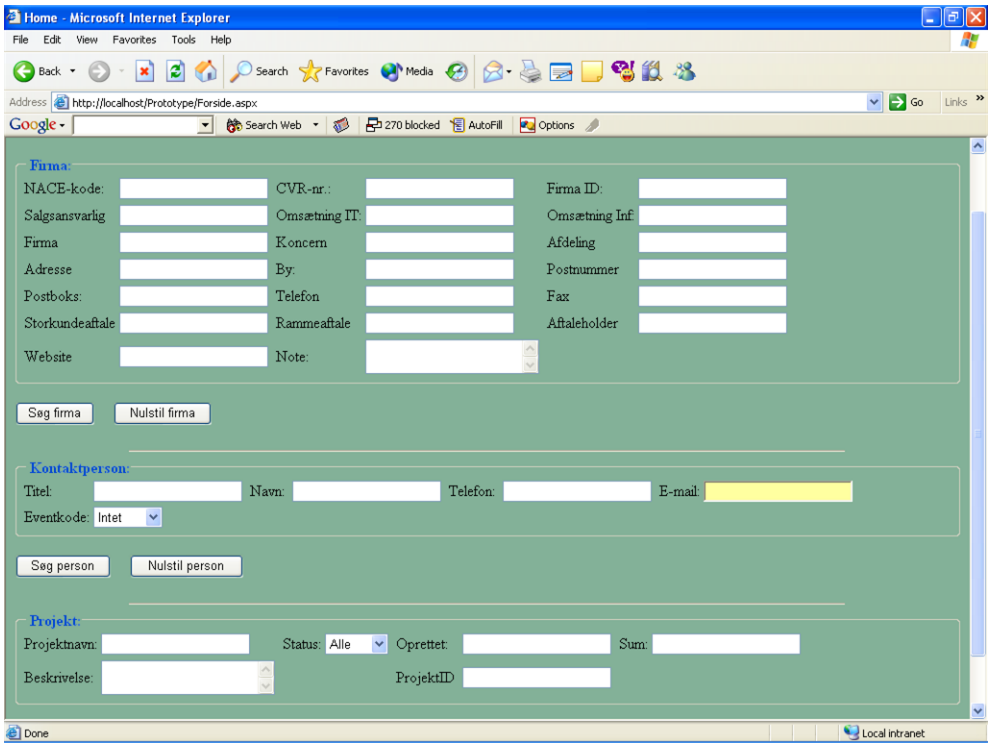
Ved mødet med sælgeren var der kommet et enkelt nyt krav frem. For at programmet kan opfylde så mange af sælgernes behov som muligt, er det nødvendigt, at det

også kan bruges til deres interne rapporter. Ethvert projekt har en sum (størrelsen på tilbudet) og en ordresandsynlighed (hvor sandsynligt det er at lige netop DTI får ordren), og ud fra disse kan man beregne et forecast (hvor mange penge DTI umiddelbart regner med at få). Til de interne rapporter er det relevant at vide, hvor mange projekter en sælger er involveret i, og hvad forecastet er for alle disse projekter. Det ville ikke være praktisk anvendeligt, hvis man var nødt til at gå ind i hvert enkelt projekt for at finde ud af dette, hvorfor det var ønskeligt at have en funktion, hvor man søgte på en enkelt sælger og fik en liste over alle de projekter, han/hun var involveret i sammen med de relevante informationer om projektet.

Analyse/Re-design

Computerskærme nutildags har ofte en rimelig høj opløsning, og da programmet kun skal bruges af computere på DTI, kunne jeg godt regne med, at de mindst havde en opløsning på 1024×768. I forhold til tidligere forslag til lay-out'et var der en masse spildplads, jeg godt kunne udnytte, hvis jeg tog højde for dette. Igen er ændringen mest synlig på forsiden, som kan ses i figur 9

Jeg håbede, at jeg helt kunne undgå, at siden scrollede ved at lave disse ændringer. Det var desværre ikke tilfældet, men resultatet var alligevel bedre, da det nu er tydeligere, at der er mere på siden.



Figur 9: Tredje udkast til en mulig forside.

Opbygning af en interaktiv model

Takket være 'sandkassen' havde jeg nu adgang til de databaser, programmet skulle hente oplysninger fra, og kunne derfor langt om længe begynde at tilpasse de funktioner, jeg havde implementeret i sidste iteration til den rigtige database. Den største udfordring var nu at finde ud af, hvordan resultaterne af søgningerne bedst kunne præsenteres, da man selv med meget præcise søgninger ikke kunne være sikker på, at der kun kom et resultat pga. de gengangere, der fandtes i databaserne, omtalt i afsnit 6. Jeg valgte, at med mindre der virkelig kun kom ét svar, bliver resultaterne af søgningen vist i en tabel, hvorfra man kan vælge det ønskede resultat og derved få vist alle oplysningerne om det givne firma/person/projekt.

Firmaer, personer og projekter har alle et unikt identifikationsnummer, som kan benyttes til at finde resten af oplysningerne. For at kunne videresende informationer fra en side til en anden benyttede jeg mig af `Sessions` til at gemme søgningens ID. `Sessions` er en funktion i ASP.NET. Ved at gemme tekststrenge i en `Session` kan alle siderne hente tekststrengen frem, så længe den givne `Session` er gyldig (typisk 20 minutter). `Sessions` bliver mest brugt til at sørge for at et login udløber efter et prædefineret tidsinterval, men kan også benyttes til at dele informationer mellem forskellige sider.

Nogle af sælgerne havde nævnt ønsket om, at forsiden skulle have et 'parent/child' forhold mellem firmasøgningen og personsøgningen, og mellem firmasøgningen og projektsøgningen således at hvis man søgte på et firma, ville alle kontaktpersoner og projekter ved det firma også blive vist. I teorien kunne jeg godt se det praktiske ved dette, men i praksis betød det, at en given søgning kom til at tage op til 3 gange så lang tid. Umiddelbart mente jeg ikke, at det kunne betale sig, men da det ikke er mig, der skal benytte det endelige produkt, valgte jeg at implementere det alligevel, så man, når man søger vha. firmasøgefelterne på forsiden, kan vælge om man kun vil søge på et firma ('Søg firma'), eller om man også vil have resultater for personer og projekter ('Søg alt').

Da man kan være interesseret i at lave mere eller mindre detaljerede søgninger, er det nok at have indtastet noget i blot ét af søgefelterne.

Evaluering

Jeg viste nogle af sælgerne det foreløbige program, og de var heldigvis meget tilfredse med det. Jeg havde forstået deres ønsker omkring layout rigtigt, programmet opfyldte deres krav i og med, at det kan søge og redigere i databaserne. Jeg bad dem om selv at finde rundt i programmet, i stedet for at jeg bare præsenterede dem for det, for at få en fornemmelse af programmets brugervenlighed. De syntes at programmet umiddelbart virkede intuitivt og let at bruge. En beskrivelse af de opgaver de løste kan ses i appendiks E.2.

Der kom ingen forslag til rettelser af det jeg allerede havde lavet, men nogle af de elementer, jeg havde nedprioriteret, blev jeg bedt om at tage med alligevel, hvis det var muligt.

10.7 4. iteration

I denne fjerde og sidste iteration blev de sidste tilføjelser implementeret i programmet inden program og database blev overgivet til Gorm Bjerre, så det kunne blive lagt op på DTI's server til almen brug.

Identifikation af behov/etablering af krav

Fra sidste evaluering kom der ingen *nye* krav, men elementer jeg tidligere blot have forstået som værende 'nice-to-have' blev uddybet til at være reelle behov.

Det drejede sig mest om småting i form af oplysninger der ikke var at finde i den oprindelige database, og som jeg derfor ville vide hvor vigtige var at have med, før jeg oprettede en ny databasetabel.

Det eneste større krav var, at det skulle være muligt at hente en liste over de 'events', der er tilknyttet en kontaktperson. Med 'events' menes der, om de skal have tilsendt julekort, katalog mv. Det er ikke blot gjort med at have en liste over events, man kan vælge imellem, da det skal være muligt at oprette nye events efterhånden, som det bliver relevant. På den anden side vil det heller ikke være formålstjeneligt blot at have et notefelt, hvori man kan skrive alle events, da man så risikerer, at den samme event bliver skrevet på mange forskellige måder (f.eks. julekort / julekort 2005 / julekort dec. 2005) og derfor bliver svær at søge på.

Af personlig erfaring vidste jeg, at det også var nødvendigt at have mulighed for at skrive notater om kontaktpersoner for at give flere søgemuligheder. Jeg var flere gange ude for, at nogen på DTI skulle have fat i mig men ikke vidste, hvad jeg hed. Min far arbejder på DTI, og de vidste naturlivis hvad han hed, men da jeg er blevet gift, har jeg skiftet efternavn. I 'Benny' er det ikke muligt at søge på f.eks. "Claus Tønderings datter". Det ville være en fordel, hvis jeg kunne få det med i programmet.

Det gik op for mig, at det var nødvendigt at begrænse adgangen til programmet, så det ikke var alle og enhver, der kunne redigere databasen.

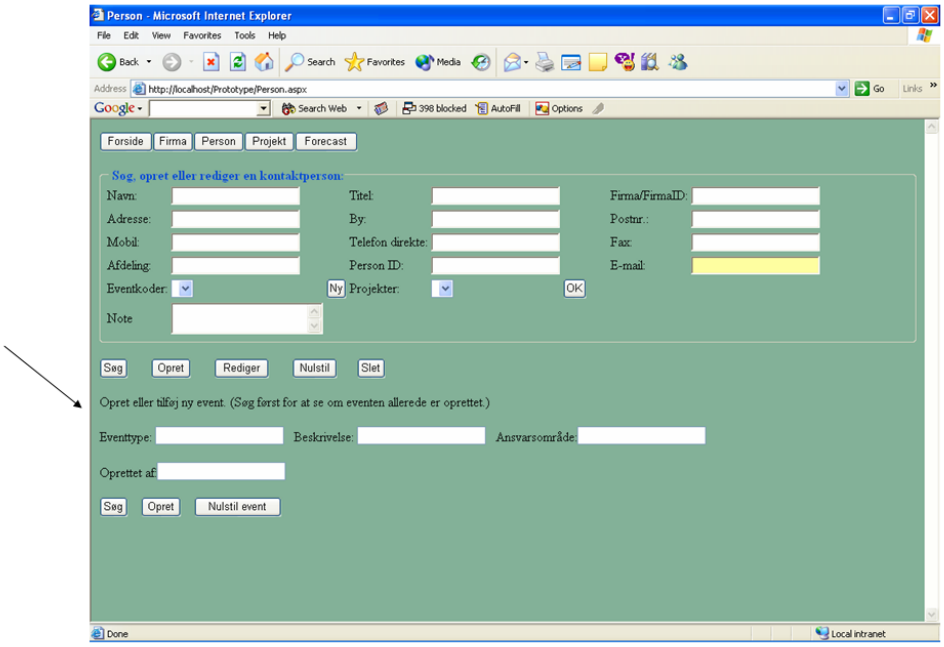
Analyse/Re-design

Jeg tilføjede en sjette side til programmet, som udelukkende håndterede login. Alle ansatte på DTI har et prædefineret brugernavn og password som er gemt i en databasetabel sammen med hvilket center de arbejder for og deres status (om de er nuværende eller tidligere ansatte). Alle nuværende ansatte i Center for Informatik har adgang til programmet.

For at undgå forvirring på `Person.aspx` besluttede jeg mig for, at funktionerne nødvendige for at tilføje eller oprette nye events ikke skulle vises, før de skulle bruges. Jeg valgte derfor, at når oplysninger om en person vises, ser man en liste over de events der allerede er knyttet til personen og ved siden af den liste en knap, hvorpå der står 'Ny'. Trykker man på den knap vises flere tekstbokse og knapper så det er muligt at tilføje events til personens liste. Hvis nødvendigt kan man også oprette helt nye events.

Opbygning af en interaktiv model

Designet beskrevet ovenfor blev implementeret. Resultatet kan ses i figur 10



Figur 10: `Person.aspx` når en ny event skal tilføjes.

Derudover fjernede jeg de felter fra forsiden, der ikke kunne bruges til at søge efter for at holde den så simpel som muligt.

Evaluering

Ved denne sidste evaluering lavede jeg en heuristik-inspektion af programmet for at sikre mig, at jeg ikke undervejs i projekt-forløbet var blevet så optaget af programmets funktionalitet, at jeg helt havde glemt at tage højde for designprincipperne, Gestalt-lovene o.l. Jeg var tilfreds med resultatet af denne inspektion, da jeg kom frem til, at siderne er bygget logisk op, og især lovene om lighed, nærhed og lukkethed er overholdt.

Jeg udførte en funktionel test for at sikre mig, at kravspecifikationen var blevet overholdt.

Udover disse test skulle programmet naturligvis også evalueres af 'rigtige' brugere for at teste brugervenligheden. Jeg kontaktede Susanne Christoph, som fortalte mig, at der var et møde for alle sælgerne ved Center for Informatik tirsdag den 29. marts. Jeg var meget velkommen til at deltage, vise mit program frem, og få sælgerne til at afprøve det.

Især ved denne sidste evaluering benyttede jeg mig af DECIDE-frameworket til at lede evalueringen. DECIDE-frameworket består af følgende 6 trin:

1. Fastlægge det overordnede mål for evalueringen.
2. Udspecificere de spørgsmål der ønskes besvaret.
3. Udvælge de evalueringsteknikker der skal bruges til at besvare spørgsmålene.
4. Identificere de praktiske opgaver der skal løses forinden.
5. Beslutte hvordan de etiske aspekter skal håndteres.
6. Evaluering og presentation af resultaterne.

Det overordnede mål for evalueringen var, at se om programmet virkede som brugerne havde forventet det, og om de havde let ved at bruge det. Hvis programmet havde alvorlige fejl eller mangler, eller hvis det viste sig ikke at være så let at bruge som ønsket, ville programmet ikke blive brugt.

De vigtigste spørgsmål, der skulle besvares, var om sælgerne kunne finde ud af at bruge programmet med et minimum af vejledning og/eller instruktion, og om det udfyldte de behov de havde til det.

Til evalueringen valgte jeg at bruge tænke højt test, for at se programmet i funktion i dets rette miljø. Testen skulle laves i den 'sandkasse', jeg havde fået stillet til rådighed, for ikke at påvirke de data, der ligger i den database, andre benytter.

Desværre var jeg så uheldig, at mødet blev aflyst i sidste øjeblik, og da webmasteren endnu ikke havde haft tid til at uploade programmet til deres server kunne brugervenlighedstesten ikke foregå ved et virtuelt møde, som det ellers blev foreslået. Heldigvis har jeg været tæt involveret med slutbrugerne igennem hele processen, så jeg har haft mulighed for at foretage små brugervenlighedstest undervejs, som alle har været tilfredsstillende. Jeg håber dog alligevel på at få mulighed for at lave en mere gennemgribende test så hurtigt som muligt.

11 Analyse

Ud fra de funde krav i afsnit 8.2 blev problemet analyseret for at finde ud af, hvad præcist programmet skal kunne, for derved at kunne udarbejde den endelige kravspecifikation. Resultatet er beskrevet ved hjælp af UML. Jeg har brugt UML-notationen fra [1].

11.1 Use Cases

Det første skridt i analysen er at definere hvilke brugere, såkaldte aktører, der skal anvende det færdige produkt, og hvilke opgaver de skal kunne løse med produktet.

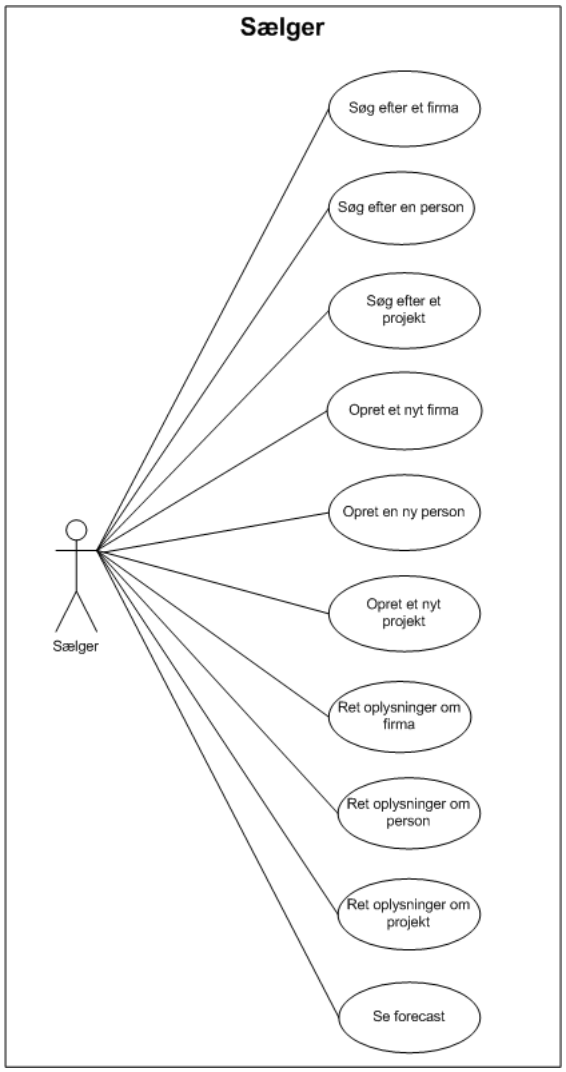
Så vidt jeg har kunnet udlede fra samtaler med kommende brugere, er der fire forskellige slags aktører: sælgere, analytikere, call service og phonere. En kort beskrivelse af de forskellige aktører og deres brug af programmet kan ses i tabel 1

Aktør	**Beskrivelse**
Sælgere	Sælgerne skal kunne bruge programmet til at gemme informationer om nye kunder, finde og opdatere informationer om gamle kunder, oprette nye projekter, lave forecasts mm. Det er ikke nødvendigt, at de kan se deres egen forecast-oversigt, men udfra personlig interesse vil de nok ønske at kunne gøre det alligevel.
Analytikere	Analytikerne skal kunne bruge programmet til at se forecasts fra sælgerne, således at de kan udfærdige rapporter om de forskellige sælgere og se, hvad DTIs indtjeninger bliver for en given periode.
Call service	Call service skal kunne benytte programmet til brug ved fakturering af kunder. De skal både kunne finde oplysninger om, hvor stort et tilbud, der er givet på et projekt, og om kunden har nogle specielle rabatordninger med DTI.
Phonere	Phonere skal kunne finde telefonnumre på potentielle kunder gennem programmet.

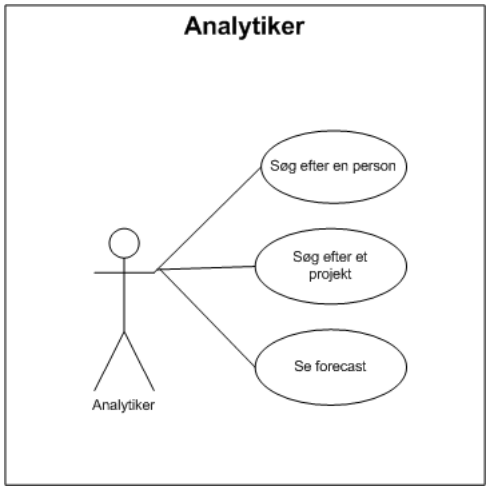
Tabel 1: Oversigt over aktører

For at give en oversigt over de forskellige opgaver, aktørerne skal kunne løse ved hjælp af programmet, har jeg for hver aktør inkluderet et use case diagram. De forskellige use cases er beskrevet efterfølgende. Use case diagrammerne kan ses i figur 11 - figur 14. De forskellige use cases er udledt af de funktionelle krav.

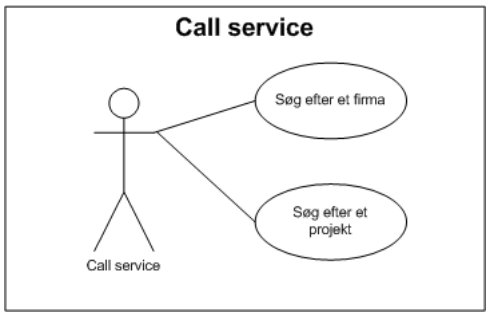
De handlinger, der skal foretages for at udføre de forskellige use cases, er beskrevet i det følgende. Der er ved hver use case beskrivelse givet en reference til det tilsvarende krav i kravspecifikationen. Ved alle beskrivelserne antages det, at man starter på siden `Forside.aspx`.



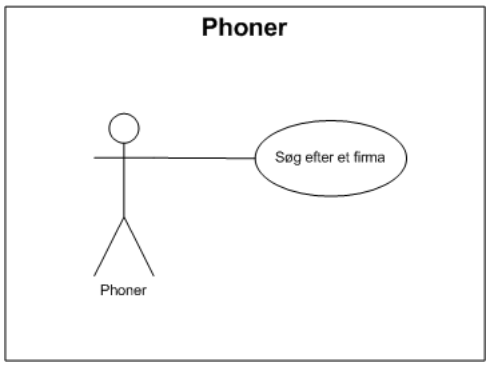
Figur 11: Use case diagram hørende til sælgere.



Figur 12: Use case diagram hørende til analytikere.



Figur 13: Use case diagram hørende til call service.



Figur 14: Use case diagram hørende til phonere.

Beskrivelser

Jeg har valgt at bruge 'sælger' som aktør i beskrivelserne, da sælgerne skal kune udføre alle opgaverne, mens analytikere, call service og phonere kun har brug for nogle af dem.

Generel og kombineret søg (K 1.4)

Aktion	*Systemsvar*
1. Sælgeren indtaster de relevante søgekriterier og trykker "Søg alt".	2. Systemet viser tre tabeller med resultater over firmaer, personer og projekter.
3. Sælgeren vælger det ønskede resultat fra tabellerne.	4. Systemet finder alle de gemte oplysninger om firmaet, personen eller projektet frem og viser dem på enten `Firma.aspx`, `Person.aspx` eller `Projekt.aspx`.

Alternativer
Trin 1-2. Sælgeren glemmer at indtaste nogle søgekriterier. Systemet giver en fejlmeddelelse. Tilbage til trin 1.
Trin 1-2. I stedet for "Søg alt" vælger sælgeren enten "Søg firma", "Søg person" eller "Søg projekt". Systemet viser en tabel med resultaterne.
Trin 3-4. Søgningen gav ingen resultater. Systemet informerer sælgeren om dette. Tilbage til trin 1.

Søg efter et firma (K 1.1)

Aktion	*Systemsvar*
1. Sælgeren vælger "Firma".	2. Systemet viser `Firma.aspx`.
3. Sælgeren indtaster de relevante søgekriterier og trykker "Søg".	4. Systemet viser en tabel over resultaterne.
5. Sælgeren vælger det ønskede firma fra tabellen.	6. Systemet finder alle de gemte oplysninger om firmaet frem og viser dem på skærmen.

Alternativer
Trin 3. Sælgeren glemmer at indtaste nogle søgekriterier. Systemet giver en fejlmeddelelse. Tilbage til trin 3.
Trin 4-6. Søgningen gav ingen resultater. Systemet informerer sælgeren om dette. Tilbage til trin 3.

Søg efter en person (K 1.2 og K 1.4)

Aktion	*Systemsvar*
1. Sælgeren vælger "Person".	2. Systemet viser `Person.aspx`.
3. Sælgeren indtaster de relevante søgekriterier og trykker "Søg".	4. Systemet viser en tabel over resultaterne.
5. Sælgeren vælger den ønskede person fra tabellen.	6. Systemet finder alle de gemte oplysninger om personen frem og viser dem på skærmen.

Alternativ
Trin 3. Sælgeren glemmer at indtaste nogle søgekriterier. Systemet giver en fejlmeddelelse. Tilbage til trin 3.
Trin 4-6. Søgningen gav ingen resultater. Systemet informerer sælgeren om dette. Tilbage til trin 3.

Søg efter et projekt (K 1.3 og K 1.4)

Aktion	*Systemsvar*
1. Sælgeren vælger "Projekt".	2. Systemet viser `Projekt.aspx`.
3. Sælgeren indtaster de relevante søgekriterier og trykker "Søg".	4. Systemet viser en tabel med resultaterne.
5. Sælgeren vælger det ønskede projekt fra tabellen.	6. Systemet finder alle de gemte oplysninger om projektet frem og viser dem på skærmen.

Alternativer
Trin 3. Sælgeren glemmer at indtaste nogle søgekriterier. Systemet giver en fejlmeddelelse. Tilbage til trin 3.
Trin 4-6. Søgningen gav ingen resultater. Systemet informerer sælgeren om dette. Tilbage til trin 3.

Opret et nyt firma (K 1.5)

Aktion	*Systemsvar*
1. Sælgeren vælger "Firma".	2. Systemet viser `Firma.aspx`.
3. Sælgeren indtaster de kendte oplysninger og trykker "Opret".	4. Systemet gemmer firmaet i databasen.

Alternativer
Trin 3-4. Firmaet findes allerede i databasen. Systemet giver en fejlmeddelelse. Tilbage til trin 3.
Trin 3-4. Sælgeren får ikke indtastet alle de nødvendige oplysninger. Systemet giver en fejlmeddelelse. Tilbage til trin 3.

Opret en ny person (K 1.6)

Aktion	*Systemsvar*
1. Sælgeren vælger "Person".	2. Systemet viser `Person.aspx`.
3. Sælgeren indtaster de kendte oplysninger og trykker "Opret".	4. Systemet gemmer personen i databasen.

Alternativer
Trin 3-4. Personen findes allerede i databasen. Systemet giver en fejlmeddelelse. Tilbage til trin 3.
Trin 3-4. Sælgeren får ikke indtastet alle de nødvendige oplysninger. Systemet giver en fejlmeddelelse. Tilbage til trin 3.

Opret et nyt projekt (K 1.7)

Aktion	*Systemsvar*
1. Sælgeren vælger "Projekt".	2. Systemet viser `Projekt.aspx`.
3. Sælgeren indtaster de kendte oplysninger og trykker "Opret".	4. Systemet gemmer projektet i databasen.

Alternativer
Trin 3-4. Projektet findes allerede i databasen. Systemet giver en fejlmeddelelse. Tilbage til trin 3.
Trin 3-4. Sælgeren får ikke indtastet alle de nødvendige oplysninger. Systemet giver en fejlmeddelelse. Tilbage til trin 3.

Ret oplysninger om firma (K 1.8)

Aktion	*Systemsvar*
1. Sælgeren finder et specifikt firma som beskrevet ovenfor.	2. Systemet viser `Firma.aspx` med de relevante oplysninger.
3. Sælgeren ændrer nogle eller alle oplysninger og trykker "Rediger".	4. Systemet spørger om sælgeren er sikker.
5. Sælgeren svarer 'Ja'.	6. Systemet overskriver de gamle oplysninger med de nye.

Alternativ
Trin 5-6. Sælgeren svarer nej. Tilbage til trin 3.

Ret oplysninger om person (K 1.8)

Aktion	*Systemsvar*
1. Sælgeren finder en specifik person som beskrevet ovenfor.	2. Systemet viser `Person.aspx` med de relevante oplysninger.
3. Sælgeren ændrer nogle eller alle oplysninger og trykker "Rediger".	4. Systemet spørger om sælgeren er sikker.
5. Sælgeren svarer 'Ja'.	6. Systemet overskriver de gamle oplysninger med de nye.

Alternativ
Trin 5-6. Sælgeren svarer nej. Tilbage til trin 3.

Ret oplysninger om projekt (K 1.8)

Aktion	*Systemsvar*
1. Sælgeren finder et specifikt projekt som beskrevet ovenfor.	2. Systemet viser `Projekt.aspx` med de relevante oplysninger.
3. Sælgeren ændrer nogle eller alle oplysninger og trykker "Rediger".	4. Systemet spørger om sælgeren er sikker.
5. Sælgeren svarer 'Ja'.	6. Systemet overskriver de gamle oplysninger med de nye.

Alternativ
Trin 5-6. Sælgeren svarer nej. Tilbage til trin 3.

Se forecast (K 1.9)

Aktion	*Systemsvar*
1. Sælgeren vælger "Forecast".	2. Systemet viser `Forecast.aspx`.
3. Sælgeren indtaster sit navn og trykker "Find forecast".	4. Systemet viser en oversigt over de projekter sælgeren er involveret i og deres forecasts.

12 Designspecifikation

I designfasen er målet at finde ud af, hvordan programmet skal virke, og hvordan de forskellige dele af programmet skal hænge sammen.

I det følgende vil jeg beskrive programmet, der overordnet består af tre dele: En brugerflade, en C#-del med de funktioner der kaldes via brugerfladen og en database. Jeg vil slutte af med at give en beskrivelse af den overordnede struktur.

12.1 Brugerflade

Jeg har lavet fem webforms i ASP.NET. ASP.NET sammenknytter almindelig HTML med et programmeringssprog som f.eks. C#.

Brugerfladen er i høj grad baseret på de idéer, jeg fik fra sælgerne tidligt i forløbet. Det var den almene holdning, at det var vigtigere, at navigationen blev hurtig, end at siderne blev simple. Naturligvis skulle de stadig være til at overskue, men de slutbrugere, jeg talte med, mente alle, at det var vigtigt, at man hurtigt kunne komme frem til den side, man havde brug for. Så længe det var logisk sat op, ville det ikke forvirre dem, at der kom flere funktioner på samme side. Efter at have fået lidt mere erfaring med 'Benny' kan jeg godt forstå dem. Det er ikke hensigtsmæssigt, hvis man skal ind på fem forskellige sider blot for at finde noget så simpelt som f.eks. en persons telefonnummer. Brugerflade-delen består derfor kun af seks dele: login-siden, forsiden, firma-delen, person-delen, projekt-delen og forecast-delen.

Siderne ligner hinanden så meget som muligt, for at sikre konsistens i programmet og gøre det så enkelt som muligt for brugerne at lære programmet at kende.

12.2 C#-delen

Uden den bagvedliggende kode er brugerfladen ikke meget bevendt. Til samtlige aspx-sider hører der en aspx.cs-side, som indeholder den funktionalitet siden har brug for for at kunne hente informationer fra databasen. Til kommunikationen med databasen bruges C# pakken `data.sqlClient`. C#-delen sørger for forbindelsen til databasen og for at udføre SQL-kommandoer samt at præsentere resultaterne på en overskuelig måde.

12.3 Database

Programmet er baseret på en allerede fungerende database. Jeg har ikke været med til at designe denne database, men er gået ud fra den, som den var.

Databasen er en MS SQL-database. Alle informationer om firmaer, personer og projekter er taget herfra. Jeg har kun brugt ganske få af de databasetabeller, der er inkluderet i databasen, nemlig `Firma`, `Person`, `Postnr`, `FirmaLogBog`, `FirmaKoder`, `FirmaKodeRegistreringer`, `KontaktRegistreringer` og `KontaktKoder`. Disse tabeller er de eneste, der omhandler firmaer, personer og projekter, og er derfor de eneste, der er relevante for mit program.

Derudover har jeg selv tilføjet fire ekstra tabeller, `Forecast`, `FirmaAftale`, `FirmaKoncern` og `PersonNote`.

Da det viste sig nødvendigt at tilføje en login-side, blev det også nødvendigt at medtage `KursusAnsvarlig` som indeholder de relevante oplysninger om ansatte ved DTI.

Databasedesignet over de brugte tabeller kan ses i E/R-diagrammet i figur 1 i afsnit 4, og det tilhørende databaseskema kan ses i figur 15. Der er benyttet samme notation som i [11]. Databasetabellerne er meget omfattende med mange forskellige attributter. For overskuelighedens skyld har jeg valgt kun at medtage de attributter, der bliver anvendt i programmet. Databasetyperne kan ses i appendiks D.

Databasen er på Boyce-Codd Normal Form (BCNF), hvilket betyder, at der ikke kan forekomme redundans eller uregelmæssigheder ved ændringer i databasen. Bevis herfor kan findes i appendiks C.

For at kunne benytte alle felterne på skærmbillederne er det nødvendigt at tilføje ekstra databasetabeller, da ikke alle informationerne var at finde i de oprindelige tabeller. Jeg har derfor tilføjet nogle ekstra tabeller (markeret med * i figur 15), hvor jeg mente det var mest presserende. På baggrund af de interviews jeg havde med slutbrugerne, vidste jeg, hvilke informationer de mest savnede at have adgang til gennem 'Benny'. Jeg valgte derfor at prioritere netop disse tabeller højest, da jeg således kunne medtage disse informationer. Især `Forecast` blev anset som et 'must' at få med, da dette emne slet ikke bliver behandlet i 'Benny'.

12.4 Overordnet struktur

De seks webforms er næsten uafhængige, så det er let at skifte sider ud, hvis det skulle blive nødvendigt. Den eneste forbindelse mellem siderne er de direkte links via navigationsknapperne øverst på siderne (som kalder en simpel `Redirect`-metode) og de nødvendige "Sessions", der sørger for at overføre oplysninger videre fra en side. Der bruges "Sessions" 3 forskellige steder: for at overføre brugernavnet fra `Login.aspx` til alle de andre sider; for at overføre oplysninger fra `Forside.aspx` til `Firma.aspx`, `Person.aspx` og `Projekt.aspx`; og for at overføre oplysninger fra `Firma.aspx` og `Person.aspx` til `Projekt.aspx`. Derudover er de tre hoveddele sat sammen i en lagstruktur, hvor C#-delen sørger for kommunikationen mellem brugerfladen og databasen. Denne opbygning er oplagt når man benytter ASP.NET. Det gør det samtidig nemmere at udskifte enten brugerflade eller database, hvis det skulle vise sig nødvendigt.

Denne opbygning betyder, at de fleste handlingsforløb følger samme mønster. Brugeren kommunikerer med programmet via brugerfladen, der kalder metoder i de tilhørende aspx.cs-sider, der kommunikerer med databasen. Der returneres i omvendt rækkefølge, således at brugerfladen kan opdateres hvis nødvendigt. Dette er illustreret i det generelle sekvensdiagram vist i figur 16. Et konkret sekvensdiagram for use casen `Opret person` er vist i figur 17.

Firma{FirmaID, Firmanavn1, Telefonnr, Telefax, Adresse1, Postboks, Postnr, salgsKontakt, OprettetAf, Oprettet, OpdateretAf, Opdateret}
Indeholder de mest generelle oplysninger om et firma.

*FirmaAftale{FirmaAftaleID, FirmaID, StorKundeAftale, RammeAftale, AftaleHolder, Note}
Fortæller om et givet firma har specielle rabatordninger med DTI, og hvem der er ansvarlig for dem.

*FirmaKoncern{FirmaKoncernID, FirmaID, Koncern, Afdeling}
Fortæller hvilken koncern og afdeling et firma hører under.

FirmaKoder{FirmaKodeID, FirmaKode, FirmaKodeTypeNavn}
Fortæller hvilke koder et firma har.

FirmaKodeRegistreringer{FirmaKodeRegistreringID, FirmaKodeID, FirmaID}
Knytter en FirmaKode til et firma.

Postnr{Postnr, Bynavn}
Giver forbindelse mellem postnummer og bynavn.

Person{PersonID, FirmaID, Navn, Stilling, Telefonnr, Afdeling, Email, OprettetAf, Oprettet, OpdateretAf, Opdateret}
Indeholder de mest generelle oplysninger om både kontaktpersoner og ansatte ved DTI.

*PersonNote{PersonNoteID, PersonID, Note}
Benyttes til de informationer `Person` ikke dækker.

FirmaLogBog{FirmaLogbogID, FirmaID, Overskrift, LogDato, Opfoelgning, KundeKontakt, LogTekst, TilbudsBeloeb, TilbudsUdloeb, Oprettet, OprettetAf, OpdateretAf, Opdateret}
Indeholder de mest generelle oplysninger om projekter ved DTI.

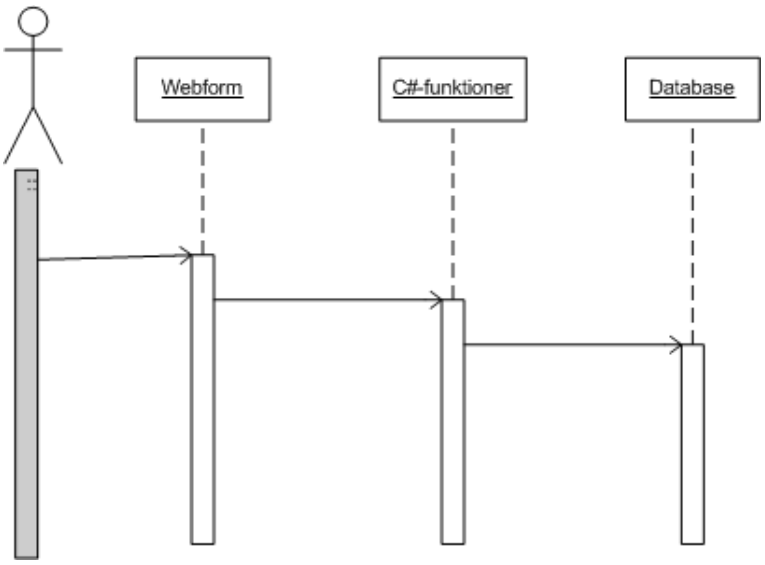
*Forecast{ForecastID, FirmaLogbogID, FirmaID, ForecastBeloeb, BestCase, WorstCase, OrdreSSH, AnsvarTI, ProjektDeltagerTI, Okonomi, Indflydelse, OprettetAf, Oprettet, OpdateretAf, Opdateret, Note}
Indeholder informationer om et projekts forecast, projektdeltagere ved DTI og hos firmaet samt et notefelt til ekstra oplysninger.

KontaktRegistreringer{KontaktRegistrerID, PersonID, KontaktKodeID, OprettetAf}
Knytter personer og kontaktkoder sammen.

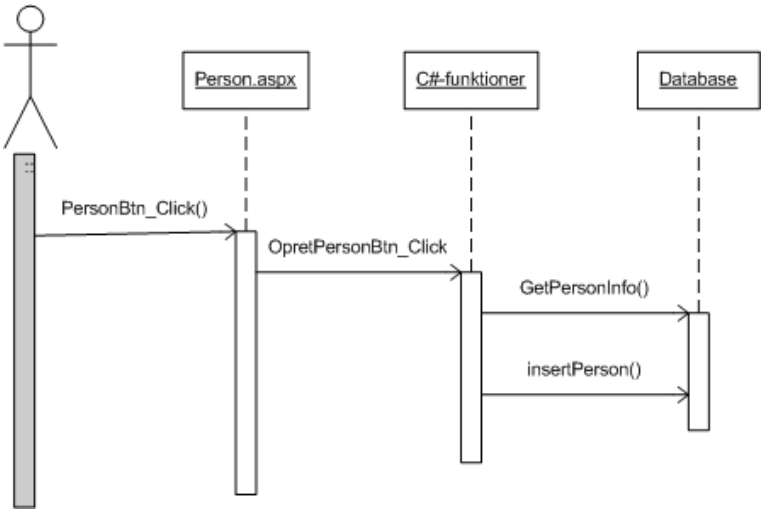
KontaktKoder{KontaktKodeID, AnsvarligAfdelingNavn, KontaktKodeTypeNavn, KortNavn, OprettetAf}
Benyttes til at beskrive de forskellige events.

KursusAnsvarlig{KursusAnsvarligID, KursusAnsvarlig, CenterKode, Status}
Giver oplysninger om hvilket center en ansat arbejder for og vedkommendes status.

Figur 15: Databaseskema. Tabeller markeret med * er tilføjet til databasen af mig.



Figur 16: Generelt sekvensdiagram.



Figur 17: Sekvensdiagram for `Opret person`.

13 Test

For at kunne bekræfte at et program virker som forventet, er det vigtigt at udføre relevante tests på programmet. Jeg har udført en funktionel test og en brugervenlighedstest. Det er i praksis ikke muligt at bevise, at et program er fejlfrit. De forskellige tests kan blot påvise, at programmet virker som ønsket også i mere usædvanlige situationer.

På grund af kodeopbygningen mener jeg ikke, der er grund til at udføre en struktrurel test på programmet. I en strukturel test undersøger man, om alle grene af programmets kode virker korrekt ved at konstruere en test-suite hvor `if`-sætninger eksekveres i situationer, hvor betingelsen er hhv. sand og falsk, hvor `for`-løkker eksekveres 0, 1 og flere gange, `try-catch`-sætninger rammes, og alle metoder kaldes. En sådan test-suite er dog ikke nødvendig i denne situation, da koden er opbygget således, at en funktionel test også vil ramme alle grene af programmet. Der er ingen `if/else-` eller `try-catch`-sætninger, der ikke bliver eksekveret, og `for`-løkkerne har ingen grænsetilfælde, hvor man kunne risikere at de reagerede anderledes. Hvis den funktionelle test viser, at programmet virker som forventet, er alle grene af programmets kode blevet ramt. Det vil derfor være overflødigt at lave en særskilt test for at efterprøve disse situationer.

13.1 Funktionel test

Den funktionelle test tager udgangspunkt i principperne i [9]. Formålet med en funktionel test er at påvise, at programmet virker efter hensigten. Udgangspunktet er at systemet skal opfylde kravspecifikationen. Denne test er også kendt som 'Black-box testing', da man er ligeglad med selve implementeringen af programmet men kun interesserer sig for, om programmets svar er som forventet, se [10, s. 443].

Testskema og resultat kan ses i appendiks E.1

13.2 Brugervenlighedstest

Det er svært at lave en præcis test for brugervenligheden, da der ikke er nogle værdier, man kan måle på. Jeg har valgt at teste brugervenligheden ved observere slutbrugerne og få kommentarer fra dem, mens de benyttede programmet.

Når man taler med brugerne, er det dog vigtigt at holde sig for øje, at det, de siger, ikke nødvendigvis stemmer overens med det, de tænker, da problemer kan blive undervuderet med et "...det er sikkert bare mig...".

I brugervenlighedstesten vil jeg først interviewe brugerne om deres opfattelse af programmet og benytter derefter "tænke højt test" som beskrevet i [4]. Testen foregår ved, at brugeren skal løse nogle predefinerede opgaver ved hjælp af programmet og prøve at beskrive deres tanker og erfaringer mens de benytter det. Mens brugeren løser opgaverne, er det vigtigt ikke ufrivilligt at komme til at give hints, da det er vigtigt at se, hvordan brugeren selv anvender programmet. Det understreges at det ikke er brugeren eller vedkommendes færdigheder, der testes, men kun programmet.

Resultaten af testen kan læses i appendiks E.2.

14 Konklusion

Gennem projektforløbet er jeg nået frem til et anvendeligt program, der kan benyttes som alternativ til den del af "Benny", der drejer sig om firma, kontaktperson og projekt. Programmet er i stand til at søge efter, redigere i og oprette nye firmaer, kontaktpersoner og projekter i DTI's database. Derudover er programmet ikke blot et alternativ til, men også en udvidelse af "Benny", da det inkluderer informationer om et projekts forecast. Pga. de afgrænsninger jeg satte for projektet, er der dog stadig oplysninger, man kun kan få fra "Benny", f.eks. informationer om de kurser DTI tilbyder; mit program er derfor ikke ment som en erstatning for "Benny" men udelukkende som et alternativ eller en udvidelse.

Programmets anvendelighed understøttes af de tests, jeg har lavet. For det første opfylder det de funktionelle krav, jeg definerede i kravspecifikationen. Dette er underbygget af den funktionelle test. Sælgerne kan finde de oplysninger, de har brug for, og let holde databasen opdateret. Det er hurtigere at foretage en søgning gennem dette program end gennem "Benny", da alle oplysninger er samlet ét sted. Selvom søgetiden 'per søgning' er den samme, skal man navigere gennem færre skærmbilleder, og foretage færre søgninger for at finde de oplysninger, man har brug for.

For det andet er de sælgere, jeg har talt med og testet programmet på, tilfredse med programmet. De mener, det vil lette deres daglige arbejde, og er enige om, at det er en forbedring til det eksisterende program. Da jeg har siddet på DTI og arbejdet, har jeg haft tæt kontakt med slutbrugerne hele vejen igennem, og de har dermed påvirket udviklingen af programmet med idéer til funktionalitet og layout. Samtidig har de været behjælpelige med en masse oplysninger om deres daglige arbejde, som har været nødvendige for analysen og designet af programmet. På grund af uheldige omstændigheder var det desværre ikke muligt at lave en 'officiel' brugervenlighedstest inden den 31. marts. Disse betragtninger er derfor baseret på de test, jeg foretog løbende under projekforløbet.

Da de databasetabeller, programmet skal benytte, er opstået ved 'knopskydning' og ikke efter en grundig analyse, var der visse oplysninger, man ikke kan finde i tabellerne, og som sælgerne så som væsentlige mangler. For at få inkluderet så mange af disse oplysninger som muligt, så jeg mig nødsaget til at oprette fire nye tabeller i databasen. I nogle tilfælde kunne man have undgået at oprette nye tabeller og i stedet blot oprette nye kolonner i allerede eksisterende tabeller. Jeg valgte dog ikke at gøre dette, da det var min opfattelse, at der var for stor risiko for at dette ville resultere i fejl i databasetabellerne.

Man skal logge på for at bruge programmet, for at undgå at uvedkommende får adgang til databasetabellerne.

Jeg har fokuseret på at ende med at have et velfungerende program, der opfylder de funktionelle krav. Der er dog stadig punkter, hvor programmet kan udvides og forbedres. Idéer til dette er nævnt i afsnit 15.

Det ovenstående er, hvad projektet er resulteret i rent programmelmæssigt. Størstedelen af min tid er brugt på de overvejelser og analyser af opgaven som er beskrevet i afsnit 10.

Jeg har lært meget ved at udvikle et program fra idé til implementation. Ved at arbejde alene i stedet for i gruppe har jeg ikke haft nogen sparringspartner i mit

daglige arbejde og har derfor selv måtte løse de problemer, jeg stødte ind i undervejs. Gennem arbejdet med dette eksamensprojekt har jeg fået mange erfaringer omkring software-udvikling og samarbejde med personer uden teknisk baggrund. Disse erfaringer vil jeg få stor glæde af i mit videre arbejde som civilingeniør.

15 Analyse af udvidelsesmuligheder

Da jeg i dette projekt har været begrænset af en tidsfrist, er der flere ting, jeg gerne ville have inkluderet, men som jeg desværre ikke havde tid eller mulighed for.

Det, jeg ser som den vigtigste udvidelse, er at inkludere de oplysninger, jeg ikke havde adgang til.

Langt de fleste af oplysningerne lå enten i den database jeg havde adgang til, eller var overhovedet ikke dokumenteret, så jeg kunne derfor selv oprette nye tabeller. Der var dog oplysninger, som f.eks. et firmas omsætning dels hos DTI og dels hos Centeret for Informatik, som ligger i et helt andet databasesystem, jeg ikke umiddelbart kan hente informationer fra. For ikke at lave redundante og dermed muligivis fejlbehæftede databasetabeller, valgte jeg ikke selv at oprette en tabel for disse oplysninger. Det er min faste overbevisning, at det er muligt at få knyttet den anden database til programmet, men at jeg blot ikke havde mulighederne for at gøre det. I en udvidelse af programmet ville det være oplagt at inkludere disse informationer.

En anden mulig udvidelse ville være at tilføje nye elementer til programmet. Jeg har som nævnt kun koncentreret mig om DTI's kunder og de informationer, der direkte angår dem. Der ligger mange andre oplysninger gemt i DTI's database, som programmet kunne sættes til at håndtere. Et muligt eksempel er alle de kurser og eksaminationer, som DTI tilbyder.

Programmet er ikke på nuværende tidpunkt udstyret med en intelligent søgefunktion, som man f.eks. ser det hos Google. Udover den mulige aa/å forvirring tager søgefunktionen ikke højde for stavefejl. En sådan forbedring ville øge brugervenligheden.

Man kan på nuværende tidspunkt ikke slette firmaer, personer eller projekter eller fjerne eventkoder fra en persons eventkodeliste. Jeg har valgt ikke at tage disse elementer med i denne version af programmet, da det ville blive ret omstændigt at sørge for, at databasetabellerne forblev konsistente efter sletning af oplysninger. Det ville dog være en oplagt funktionalitet at udvide programmet med.

Jeg har koncentreret mig om de oplysninger, der blev anset for at være 'need to have' for at gøre programmet så anvendeligt som muligt. Der er dog stadig mange 'nice-to-have' elementer som det ville være oplagt at udvide programmet med. Et eksempel på dette er en liste over en kontaktpersons interesseområder, så man f.eks. kunne finde alle kontaktpersoner med en interesse for C#, hvis man vidste, man havde et tilbud, der ville interesse dem. Et andet eksempel ville være at knytte en event til en ansat på DTI og en udløbsdato, så man ved, hvem der synes, kontaktpersonen skal modtage det givne post og hvor længe. På nuværende tidpunkt må sådanne oplysninger gemmes i en persons note.

DTI og forfatteren er i besiddelse af kildekoden til programmet.

16 Bibliografi

[1] Bennett, Farmer and McRobb, *Object-oriented systems analysis and design using UML*, 2002, 2nd. edition, McGraw-Hill.

[2] Huges and Cotterell, *Software Project Management*, 2002, 3rd edition, The McGraw-Hill Companies.

[3] Mitchell, Scott, *SAMS Teach yourself ASP.NET*, 2003, Sams.

[4] Molich, Rolf, *Brugervenligt webdesign*, 2002, Ingeniøren-bøger.

[5] Navathe, Elmasri, *Fundamentals of Database Systems*, 1997, 3rd. edition, Addison-Wesley.

[6] Petzold, Charles, *Programming Microsoft Windows with C#*, 2002, Microsoft Publishing.

[7] Preece, Jennifer and Rogers, *Interaction Design : beyond human-computer interaction*, 2002, John Wiley and Sons, Inc.

[8] PLS Rambøll Management, *CRM i Danmark 2003*, 2003, `http://www.microsoft.dk/crm`.

[9] Sestoft, Peter, *Systematic software test*, 1998, `http://www.dina.dk/~sestoft/programmering/struktur.pdf`.

[10] Sommerville, Ian, *Software Engineering*, 2001, 6th. edition, Addison-Wesley.

[11] Ullman, J.D and Widom, J., *A First Course in Database Systems*, 1997, Prentice Hall.

[12] Vinje, Poul Staal, *Projektledelse af systemudvikling*, 2000, 2. udgave, Ingeniøren|Bøger.

A Ordliste

'Benny' Kælenavnet for det nuværende søgesystem til DTI's kundedatabase.

Boyce-Codd Normal Form (BCNF) I en database der er på BCNF opstår der ikke uregelmæssigheder eller redundans ved ændringer i databasen.

brugerkrav Krav, der omhandler, hvordan programmet skal tilpasses målgrupperne.

brugeroplevelseskrav Krav, der omhandler den oplevelse, brugeren får af at bruge programmet.

brugervenlighed Udtryk for hvor enkelt og intuitivt programmet er at bruge.

brugervenlighedstest Test for hvor brugervenligt programmet er.

databaseskema Et skema med databasens relationer, attributter og nøgler.

datakrav Krav der specificerer de krævede data.

DECIDE Framework der bruges til at guide evalueringer.

designprincipper Beskrivelser af hvordan systemet skal se ud og reagere når det bruges.

domæne Det miljø programmet skal bruges i.

E/R-diagram (Entity/Relationship-diagram) Grafisk repræsentation af en database der viser entiteter og forbindelser.

funktionelle krav Krav der specificerer, hvad programmet skal kunne.

funktionel test Test der undersøger om programmet opfylder de *funktionelle krav*.

Gestaltlovene Love om hvordan vi opfatter helhed i billeder. Af eksempler kan nævnes *loven om lighed*, *loven om nærhed*, *loven om forbundethed* og *loven om lukkethed*.

heuristikinspektion En form for *brugervenlighedstest* hvor man ser på programmet med brugerens øjne for at se, om det lever op til bestemte krav, såkaldte heuristikker.

high-fidelity prototype En *prototype* der er lagt en del arbejde i. Er tættere på det færdige produkt end en *low-fidelity prototype*

interaktionsdesign (Interaction Design, ID) Gren indenfor teknologiudvikling der omhandler design af brugerflader.

iteration Del af en arbejdsprocess som kan gentages efter behov. Består i dette tilfælde af et eller flere af følgende stadier: "Domæne- og målgruppebeskrivelse", "Identifikation af behov/Etablering af krav", "Analyse/(Re)design", "Opbygning af interaktiv model", "Evaluering", "Færdigt produkt" og "Vedligeholdelse".

loven om forbundethed Elementer der er forbundet vha. linier eller lignende og derfor opfattes som hørende sammen.

loven om lighed Elementer der ligner hinanden og derfor opfattes som hørende sammen.

loven om lukkethed Elementer der er grupperet sammen, evt. lukket inde af en ramme eller lignende og derfor opfattes som hørende sammen.

loven om nærhed Elementer der sidder i nærheden af hinanden og derfor opfattes som hørende sammen.

low-fidelity prototype En hurtig, billig *prototype* lavet tidligt i procesforløbet. Typisk bare en papirtegning eller andet, der er let at ændre.

menneske-computer interaktion (Human-Computer Interaction, HCI) Studier i hvordan mennesker interagerer med computere.

NACE Kode DTI bruger til at vise, hvilken branche et firma er i.

participatory design Metode indenfor softwareudvikling, hvor slutbrugerne tages med på råd gennem hele forløbet.

procesmodel Den model, man følger under procesforløbet. Modellen viser i hvilken rækkefølge, man koncentrerer sig om de forskellige elementer i softwareudvikling.

prototype En model af en løsning. Kan være mere eller mindre avanceret. Har til formål at visualisere en produktidé.

sandkasse Et virtuelt miljø hvor programmer afvikles. Ændringer i databaser mv. påvirker ikke det almindelige arbejdsmiljø.

sekvensdiagram *UML*-diagram der viser et muligt handlingsforløb mellem menneske og program.

SQL (Structured Query Language) Sprog der gør det muligt at lave forespørgsler til en SQL-kompatibel database.

strukturel test Test der undersøger om alle grene af programmet bliver kørt.

tænke-højt test Test der undersøger *brugervenligheden* af et program. Brugere bliver bedt om at anvende programmet til at løse specifikke opgaver og skal tænke højt imens.

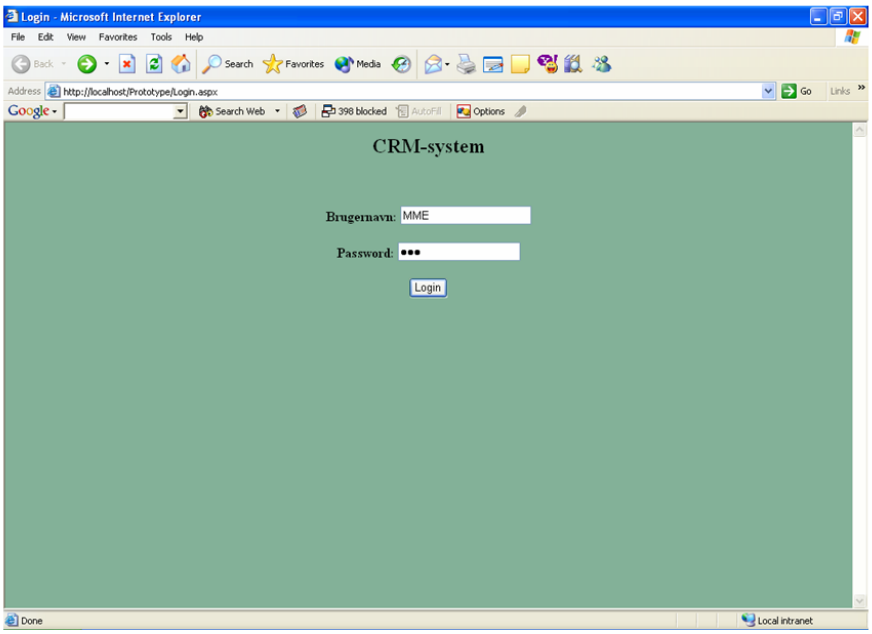
UML (Unified Modelling Language) Notation der gør det muligt at dokumentere analyse og design af et program, især vha. diagrammer.

use case Beskriver, fra brugerens synspunkt, de opgaver, brugeren skal være i stand til at løse ved hjælp af programmet.

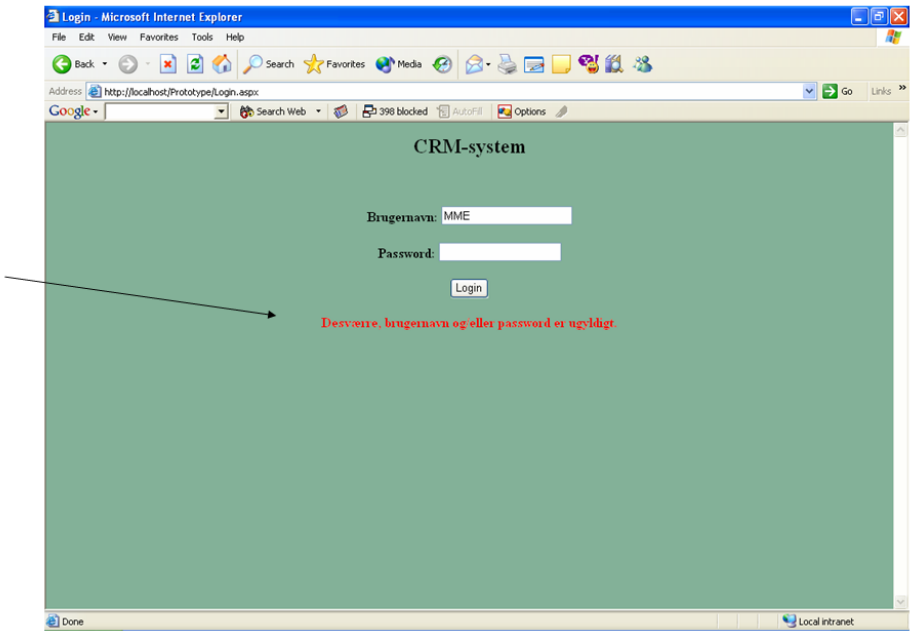
B Skærmbilleder

I dette afsnit er udvalgte skærmbilleder fra programmet vist. For overskuelighedens skyld er billederne organiseret efter, hvilken del af programmet de hører til. Jeg har valgt ikke at tage alle skærmbilleder med, da mange af dem ligner hinanden for meget, til at det vil være nødvendigt eller interessant at se dem alle. F.eks. bliver eksempler på søgninger uden resultater kun vist én gang (i afsnit B.2), da udseendet er tilsvarende på de andre skærmbilleder.

B.1 Login.aspx

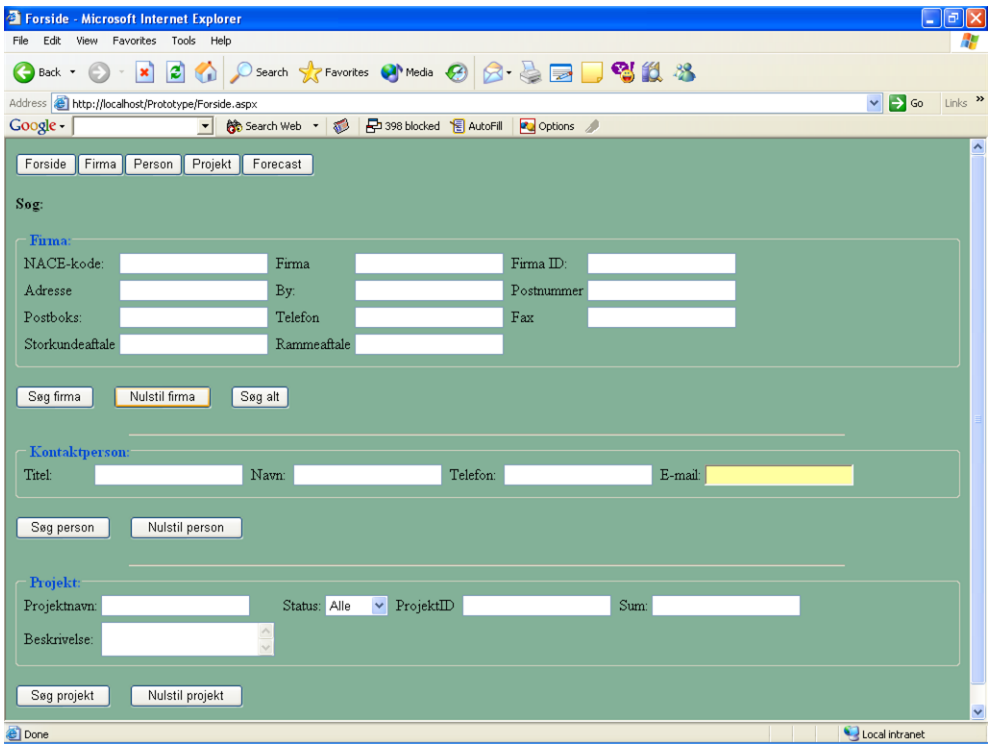


Figur 18: `Login.aspx`

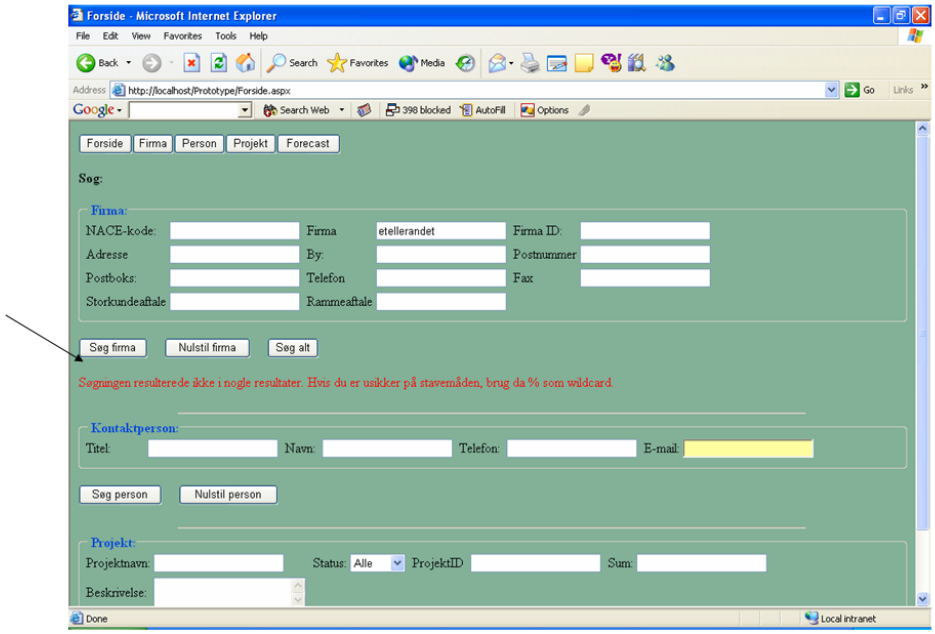


Figur 19: `Login.aspx` hvor brugernavn og password er ugyldigt.

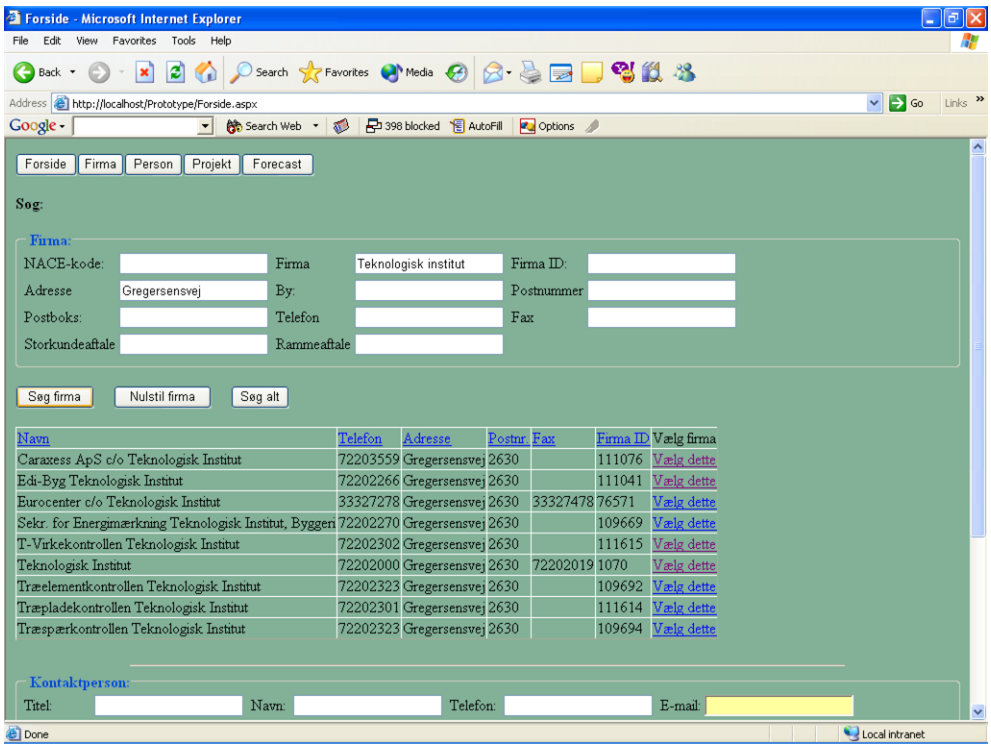
B.2 Forside.aspx



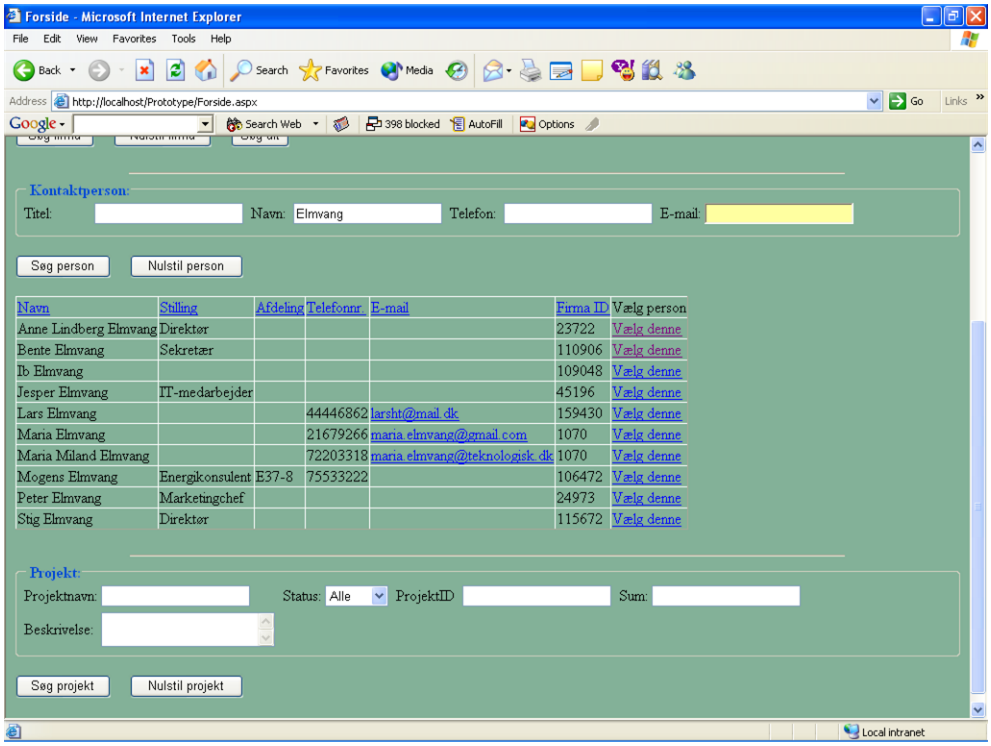
Figur 20: `Forside.aspx`



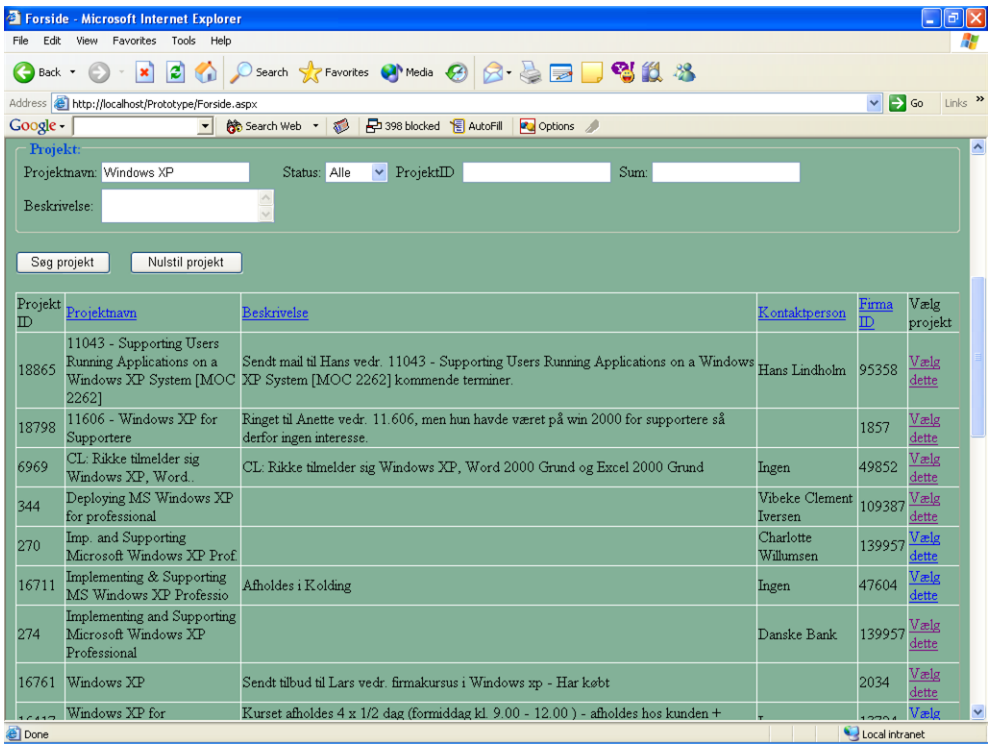
Figur 21: `Forside.aspx` hvor firmasøgning ikke giver noget resultat.



Figur 22: `Forside.aspx` efter succesfuld søgning på firma.

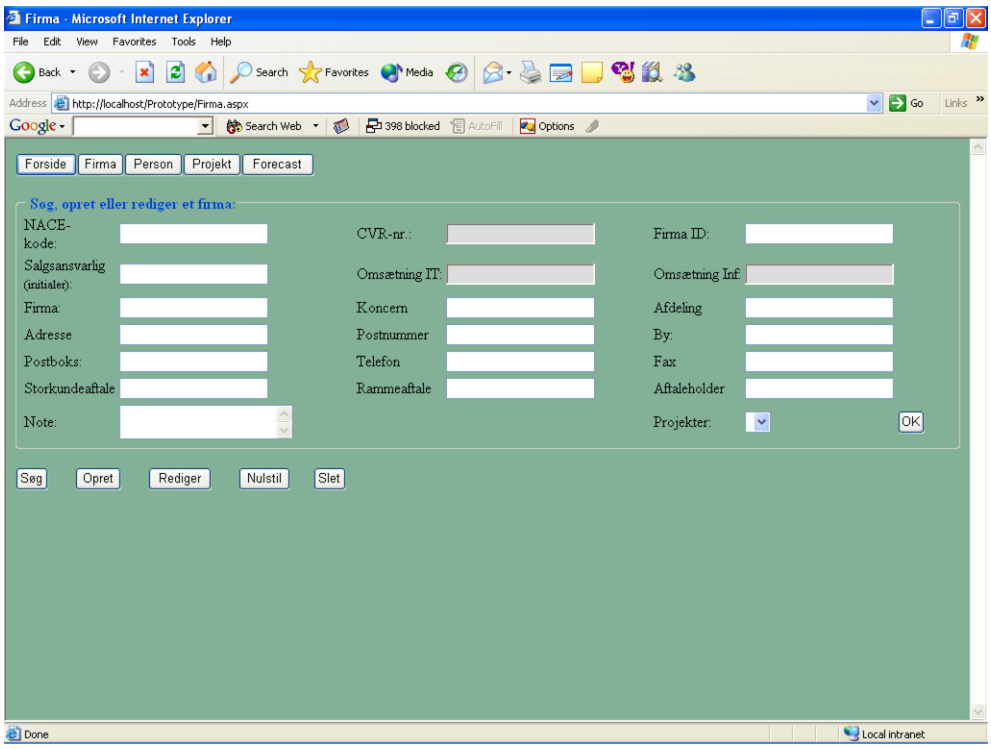


Figur 23: `Forside.aspx` efter succesfuld søgning på person.

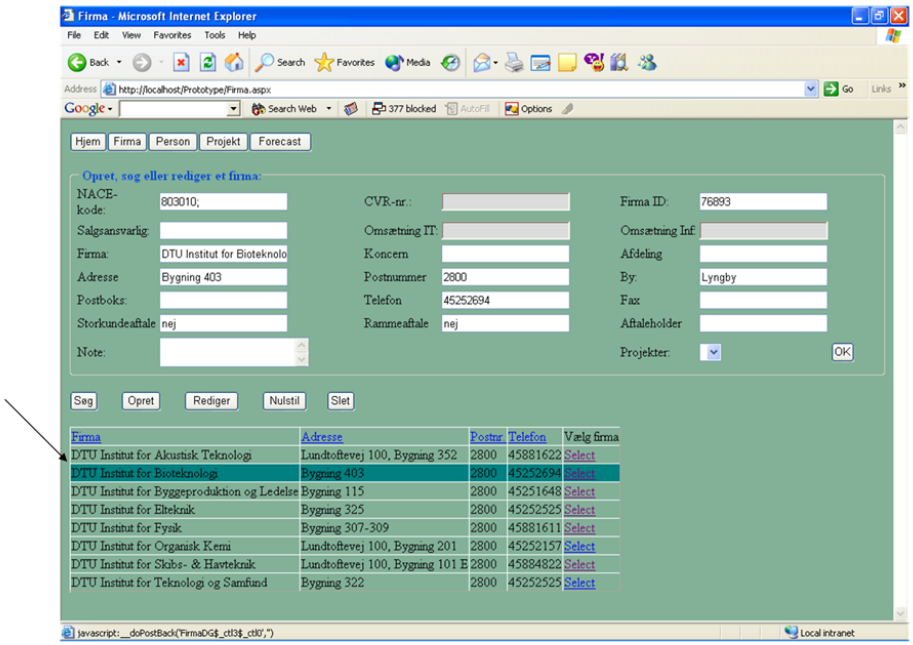


Figur 24: `Forside.aspx` efter successfuld søgning på projekt.

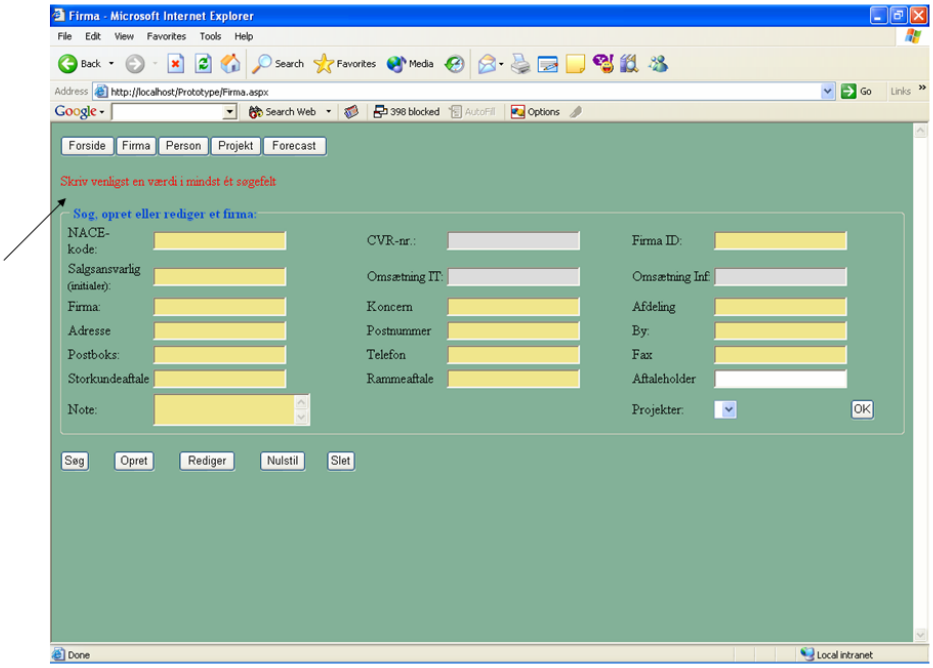
B.3 Firma.aspx



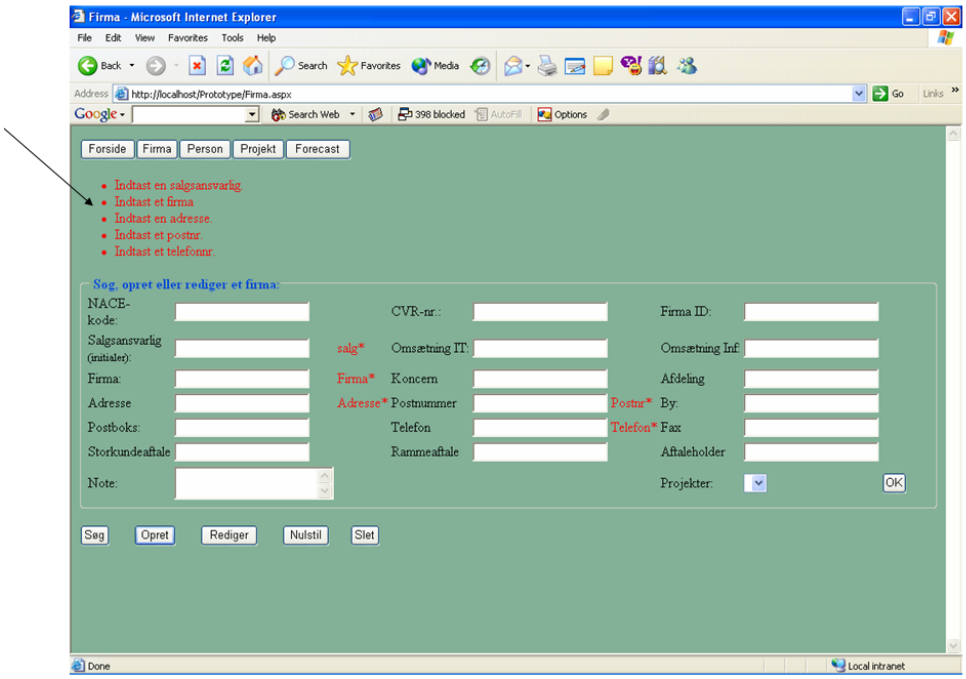
Figur 25: `Firma.aspx`



Figur 26: Resultat af søgning på "DTU" hvor et specifikt resultat er valgt.

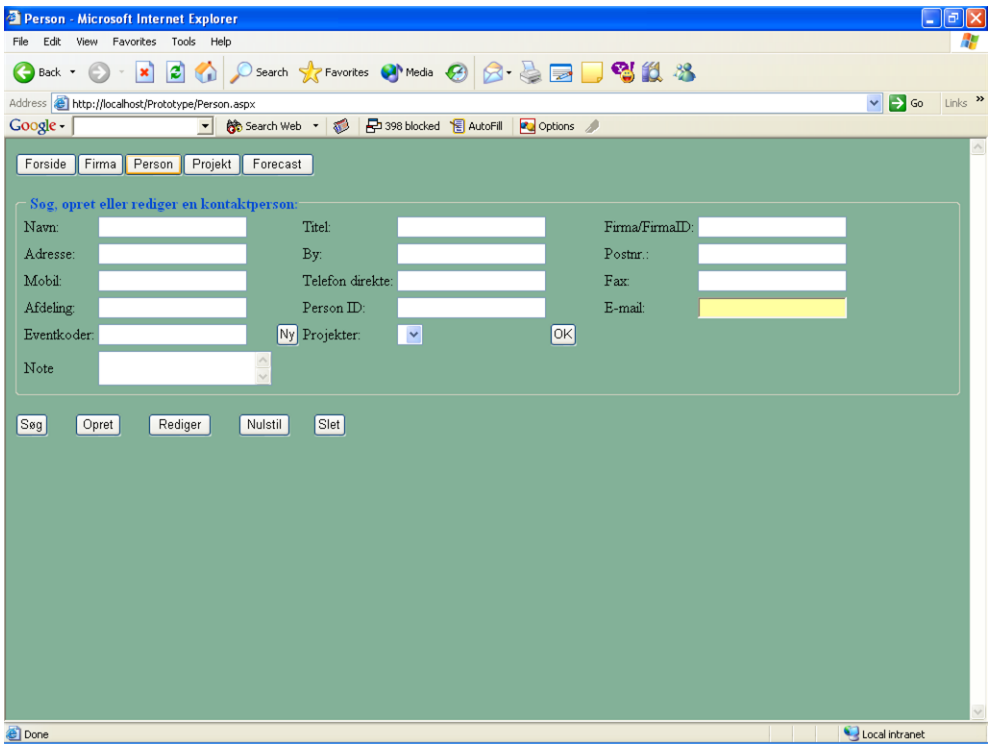


Figur 27: Fejlmeddelelse, hvis man prøver at søge efter et firma uden at have indtastet nogle oplysninger.

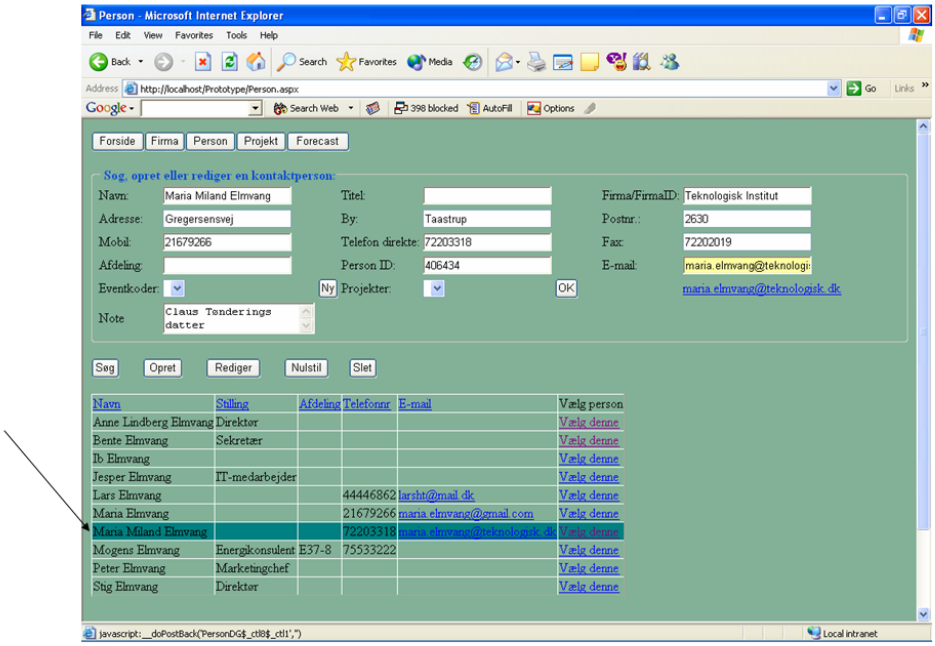


Figur 28: Fejlmeddelelse, hvis man prøver at oprette et firma uden at have indtastet de nødvendige oplysninger.

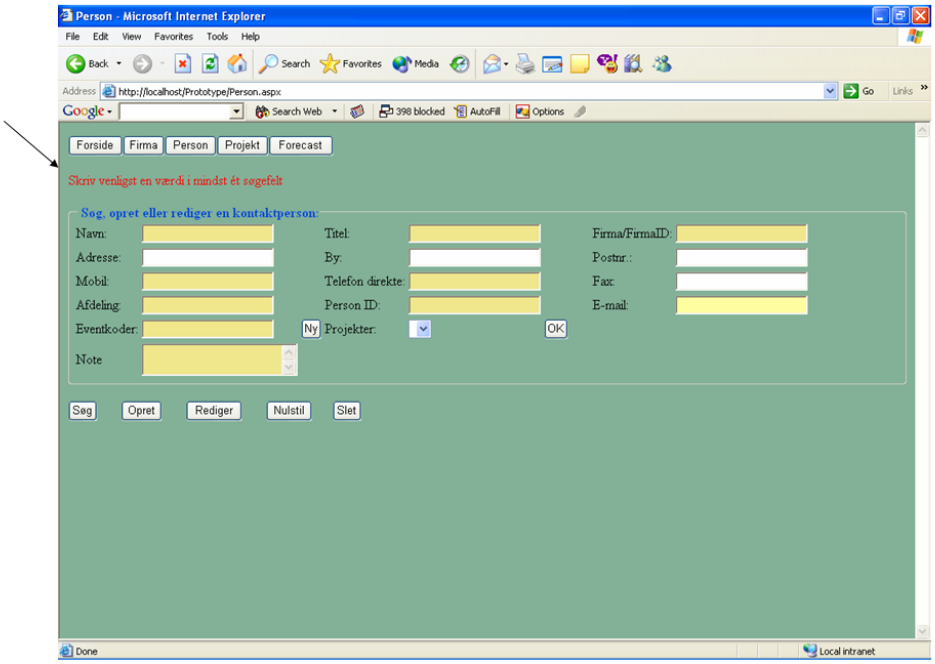
B.4 Person.aspx



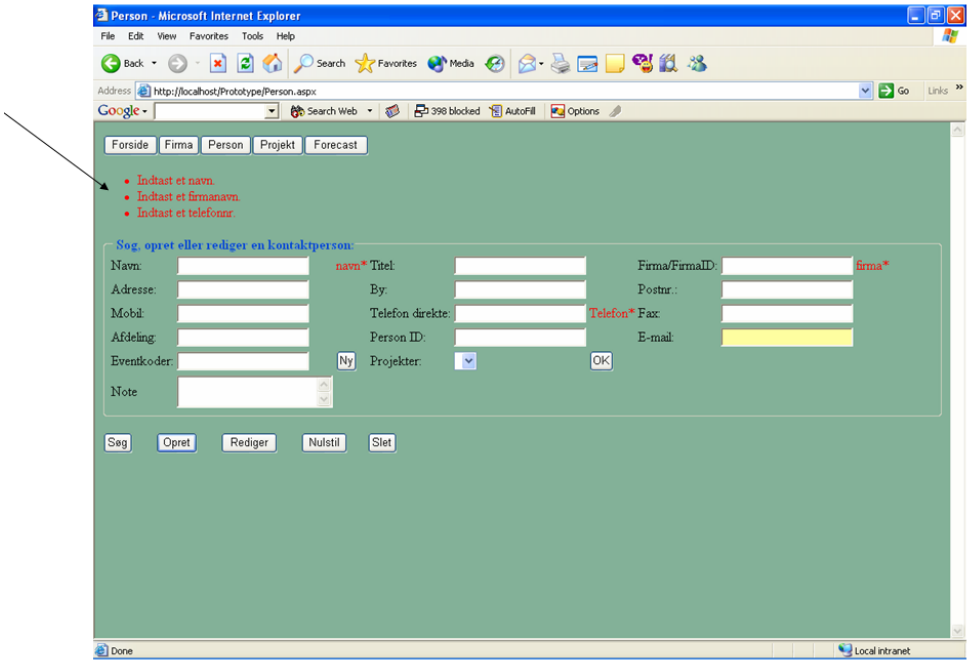
Figur 29: `Person.aspx`



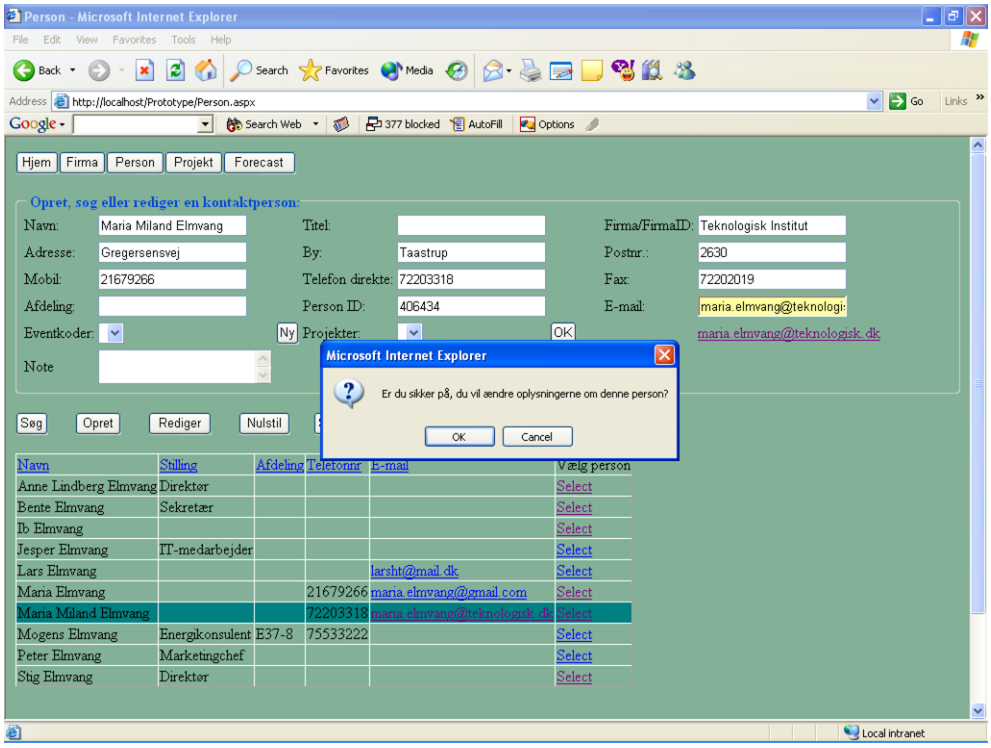
Figur 30: Resultat af søgning på "Elmvang" hvor et specifikt resultat er valgt.



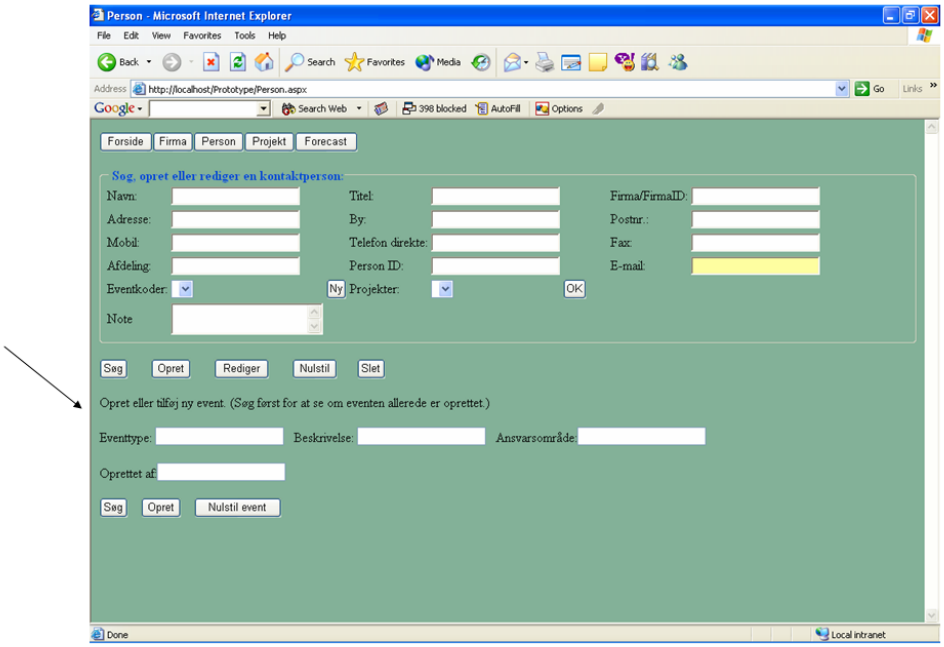
Figur 31: Fejlmeddelelse, hvis man prøver at søge efter en kontaktperson uden at have indtastet nogle oplysninger.



Figur 32: Fejlmeddelelse, hvis man prøver at oprette en person uden at have indtastet de nødvendige oplysninger.

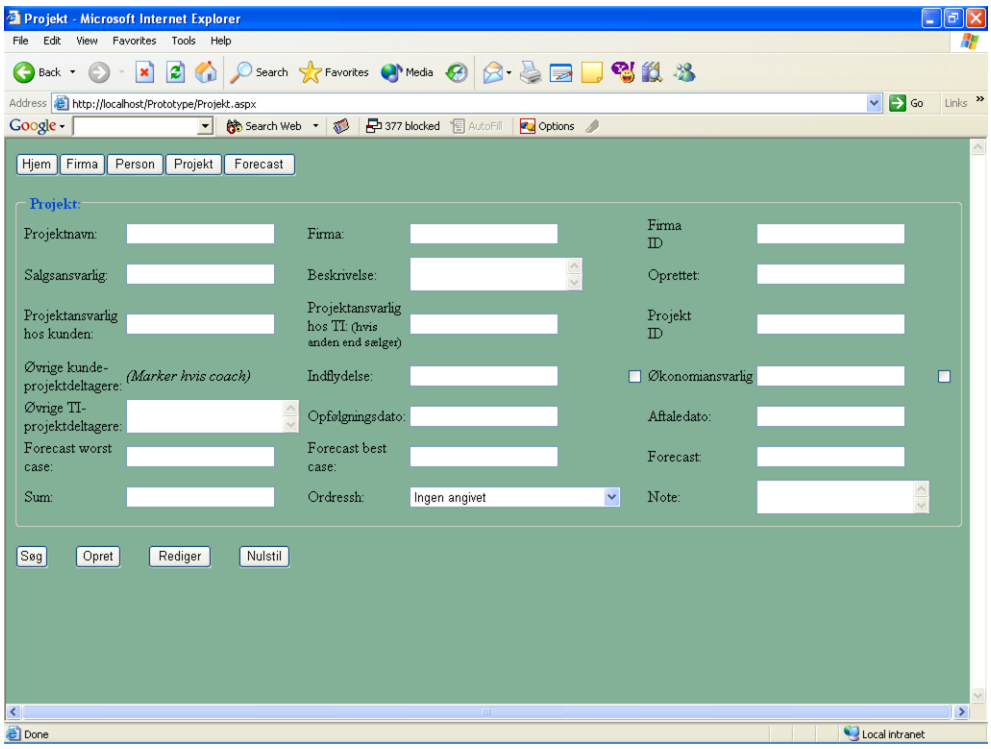


Figur 33: Dialogvindue når man forsøger at redigere en persons oplysninger.

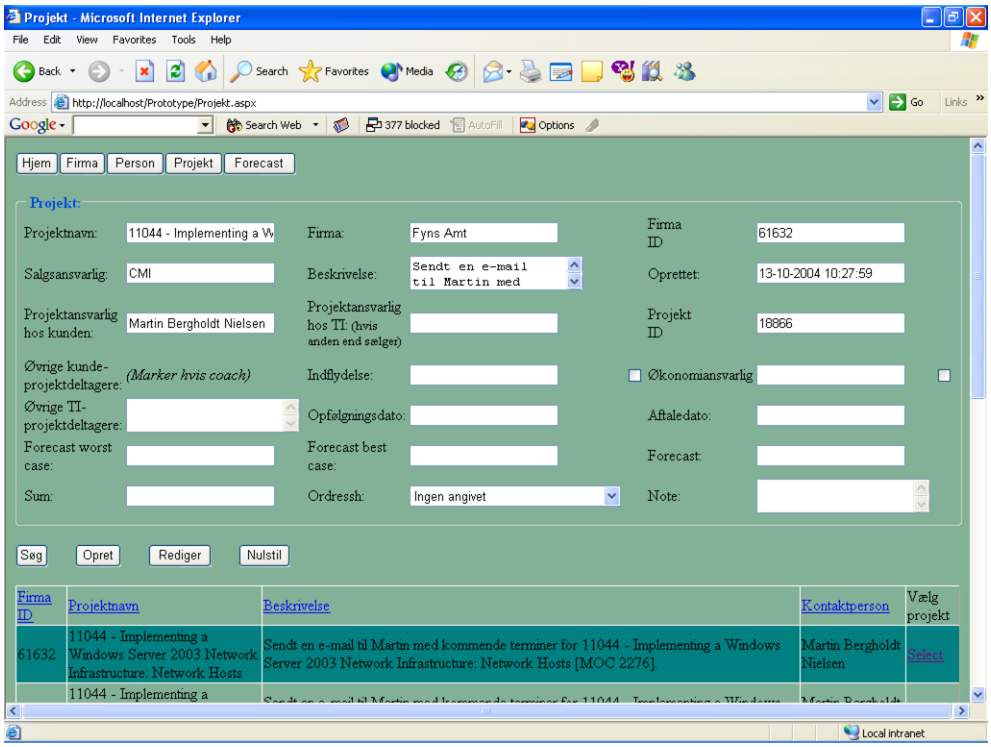


Figur 34: `Person.aspx` med mulighed for at tilføje eller oprette 'events'

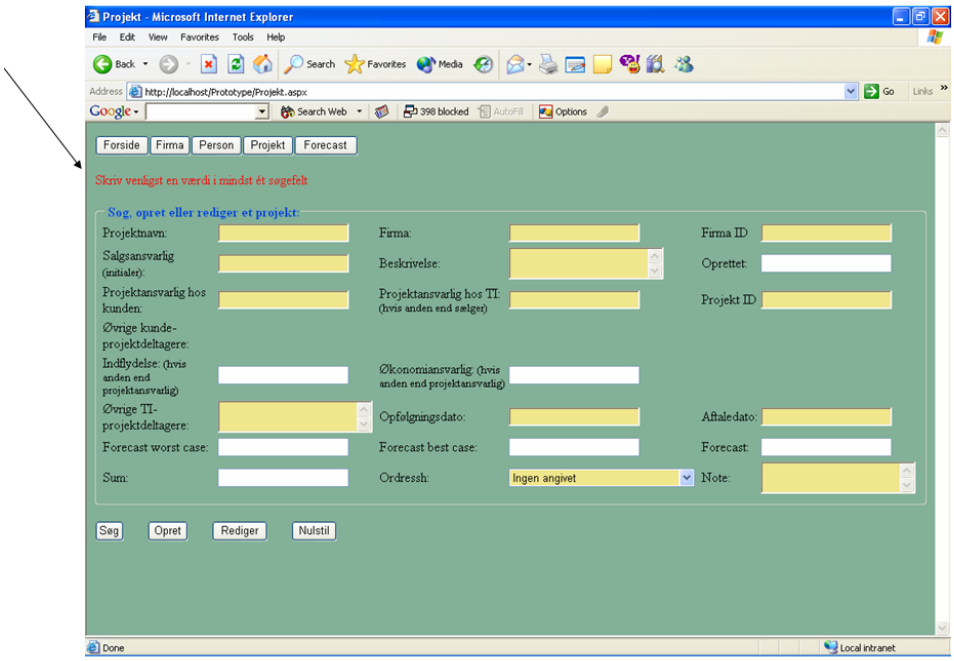
B.5 Projekt.aspx



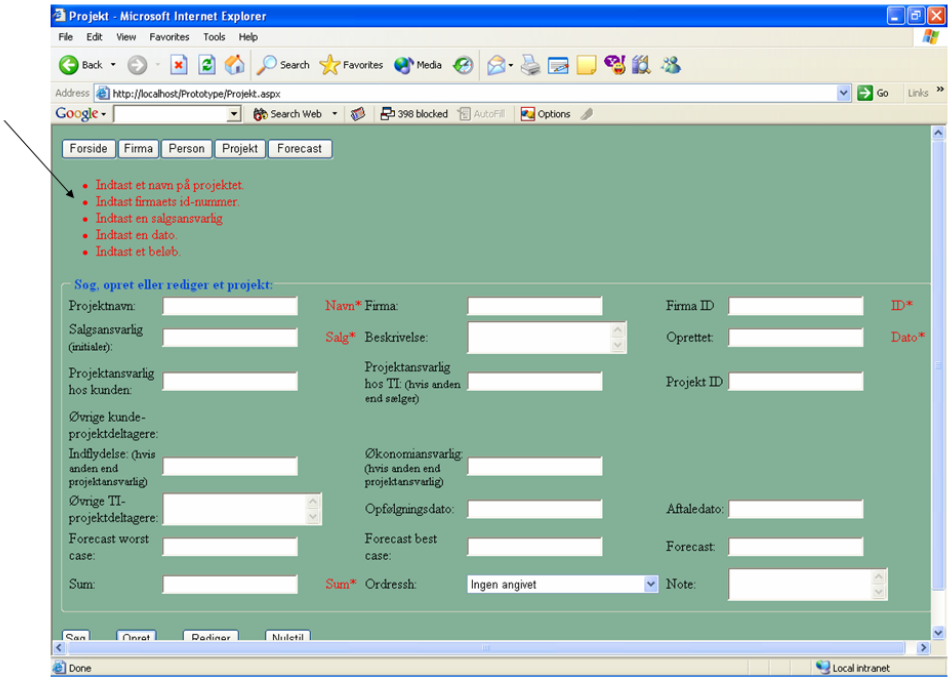
Figur 35: `Projekt.aspx`



Figur 36: Resultat af søgning på "Implementing Windows" hvor et specifikt resultat er valgt.

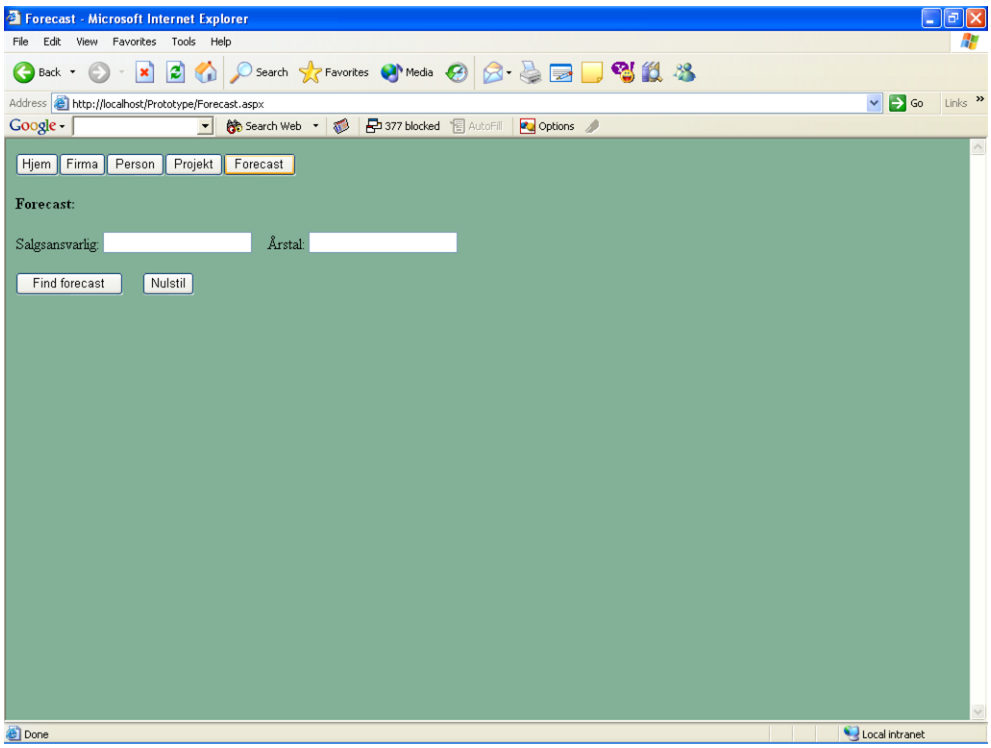


Figur 37: Fejlmeddelelse, hvis man prøver at søge efter et projekt uden at have indtastet nogle oplysninger.

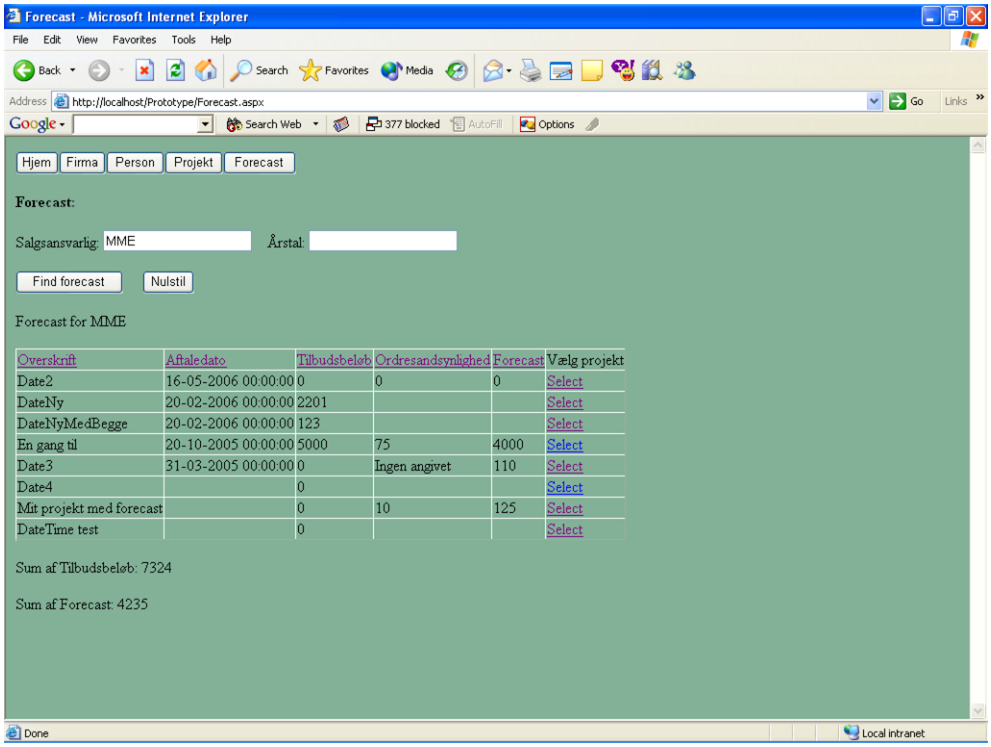


Figur 38: Fejlmeddelelse, hvis man prøver at oprette et projekt uden at have indtastet de nødvendige oplysninger.

B.6 Forecast.aspx



Figur 39: `Forecast.aspx`



Figur 40: `Forecast.aspx` efter søgning på "MME".

C Bevis for at databasen er på BCNF

For en database af denne størrelse er det vigtigt, at den er på Boyce-Codd Normal Form (BCNF), da det betyder, at der ikke kan forekomme redundans eller uregelmæssigheder ved ændring i den. Da jeg kun selv har designet ganske få af databasetabellerne og til det benyttede mit objekt-orienterede design og ikke en databasedesignmodel som sikrer at en database er på BCNF, vil jeg i det følgende bevise at hele databasen er på BCNF.

Ifølge [11, s. 140f] er der et ganske simpel krav, der skal være opfyldt, for at en database er på BCNF. Definitionen er frit oversat og lyder som følger:

Definition: En relation, R er på BCNF hvis og kun hvis der for alle ikke-trivielle afhængigheder $A_1A_2 \cdots A_n \rightarrow B_1B_2 \cdots B_n$ i R, gælder, at $\{A_1A_2 \cdots A_n\}$ er en supernøgle for R.

Med *ikke-triviel afhængighed* forstås der, at mindst et B ikke findes blandt A'erne.

For alle relationerne gælder det, at den eneste ikke-trivielle afhængighed har tabellens nøgle og kun den på venstresiden af den ikke-trivielle afhængighed.

For at underbygge dette postulat er det nødvendigt at lave en analyse af tabellerne for at bevise, at det virkelig er tilfældet. Jeg vil her beskrive analysen af **Firma**-relationen. Analysen af de andre relationer foregår på samme måde.

Den eneste ikke-triville afhængighed i denne tabel er følgende:
FirmaID → Firmanavn1, Telefonnr, Telefax, Adresse1, Postboks, Postnr, salgsKontakt

Der er ingen af attributterne på højresiden af afhængigheden, der er afhængige af hinanden. Man kan let forestille sig et firma har flere telefonnumre eller adresse, og kan derfor ikke udlede nogle af informationerne ud fra navnet alene. Tilsvarende kan man ikke udlede adressen fra postnummeret, da flere postnumre godt kan have samme gadenavn. Da **FirmaID** er supernøgle for relationen, gælder det, at relationen er på BCNF.

På samme måde ser man også at de øvrige relationerne er på BCNF, hvilket betyder, at hele databasen er på BCNF.

D Databasetyper

Den følgende tabel giver en oversigt over de forskellige typer i databasetabellerne brugt i programmet, se figur 15. Selve databasen indeholder mange flere tabeller, men det er kun de følgende der har været relevante for mit projekt, da de andre ikke omhandler DTIs kunder - dvs. firmaer, personer og projekter.

Tabel 2: Oversigt over databasetyper

Kolonne	**Type**	**Kommentar**
Firma		
FirmaID	Int(4)	AUTOINCREMENT. Primær nøgle.
Firmanavn1	varchar(70)	Skal udfyldes.
Telefonnr	varchar(25)	
Telefax	varchar(25)	
Adresse1	varchar(50)	
Postboks	varchar(14)	
Postnr	varchar(10)	Skal udfyldes.
salgsKontakt	varchar(4)	
OprettetAf	varchar(8)	Skal udfyldes.
Oprettet	datetime(8)	Skal udfyldes.
OpdateretAf	varchar(8)	
Opdateret	datetime(8)	
FirmaAftale		
FirmaAftaleID	Int(4)	AUTOINCREMENT. Primær nøgle.
FirmaID	int(4)	Skal udfyldes.
StorKundeAftale	varchar(20)	
RammeAftale	varchar(20)	
AftaleHolder	varchar(50)	
Note	varchar(1000)	
FirmaKoncern		
FirmaKoncernID	int(4)	AUTOINCREMENT. Primær nøgle.
FirmaID	int(4)	Skal udfyldes.
Koncern	varchar(50)	
Afdeling	varchar(50)	
FirmaKoder		
FirmaKondeID	int(4)	AUTOINCREMENT. Primær nøgle.
FirmaKode	varchar(50)	
FirmaKodeTypeNavn	varchar(25)	Skal udfyldes.
FirmaKodeRegistreringer		
FirmaKodeRegistreringID	int(4)	AUTOINCREMENT. Primær nøgle.
FirmaKodeID	int(4)	
FirmaID	int(4)	Skal udfyldes.
Postnr.		
Postnr	varchar(10)	Skal udfyldes. Primær nøgle.

Fortsættes på næste side

Tabel 2 – fortsat fra forrige side

Kolonne	Type	Kommentar
Bynavn	varchar(30)	
Person		
PersonID	int(4)	AUTOINCREMENT. Primær nøgle.
FirmaID	int(4)	Skal udfyldes.
Navn	varchar(40)	Skal udfyldes.
Stilling	varchar(55)	
Telefonnr	varchar(25)	
Afdeling	varchar(55)	
Email	varchar(100)	
OprettetAf	varchar(8)	Skal udfyldes.
Oprettet	datetime(8)	Skal udfyldes.
OpdateretAf	varchar(8)	
Opdateret	datetime(8)	
PersonNote		
PersonNoteID	int(4)	AUTOINCREMENT. Primær nøgle.
PersonID	int(4)	Skal udfyldes.
Note	varchar(1000)	
FirmaLogBog		
FirmaLogbogID	int(4)	AUTOINCREMENT. Primær nøgle.
FirmaID	int(4)	Skal udfyldes.
Overskrift	varchar(80)	Skal udfyldes.
LogDato	datetime(8)	Skal udfyldes.
Opfoelgning	datetime(8)	
KundeKontakt	varchar(250)	
LogTekst	varchar(7500)	
TilbudsBeloeb	int(4)	
TilbudsUdloeb	datetime(8)	
Oprettet	datetime(8)	
OprettetAf	varchar(8)	
OpdateretAf	varchar(50)	
Opdateret	datetime(8)	
Forecast		
ForecastID	int(4)	AUTOINCREMENT. Primær nøgle.
FirmaLogbogID	int(4)	Skal udfyldes.
FirmaID	int(4)	Skal udfyldes.
ForecastBeloeb	int(4)	
BestCase	int(4)	
WorstCase	int(4)	
OrdreSSH	varchar(50)	
AnsvarTI	varchar(50)	
ProjektDeltagerTI	varchar(50)	
Okonomi	varchar(50)	
Indflydelse	varchar(50)	
OprettetAf	varchar(50)	Skal udfyldes.
Oprettet	datetime(8)	Skal udfyldes.
OpdateretAf	varchar(50)	

Fortsættes på næste side

Tabel 2 – fortsat fra forrige side

Kolonne	Type	Kommentar
Opdateret	datetime(8)	
Note	varchar(1000)	
KontaktRegistreringer		
KontaktRegisterID	int(4)	AUTOINCREMENT. Primær nøgle.
PersonID	int(4)	Skal udfyldes.
KontaktKodeID	int(4)	Skal udfyldes.
OprettetAf	varchar(50)	Skal udfyldes.
KontaktKoder		
KontaktKodeID	int(4)	AUTOINCREMENT. Primær nøgle
AnsvarligAfdelingNavn	varchar(50)	Skal udfyldes.
KontaktKodeTypeNavn	varchar(100)	Skal udfyldes.
KortNavn	varchar(100)	Skal udfyldes.
OprettetAf	varchar(50)	Skal udfyldes.
KursusAnsvarlig		
KursusAnsvarligID	int(4)	AUTOINCREMENT. Primær nøgle
KursusAnsvarlig	varchar(8)	
CenterKode	int(4)	
Status	varchar(8)	

E Testdokumentation

I det følgende er der givet testskemaer og kommentarer for den funktionelle test og brugervenlighedstesten.

E.1 Funktionel test

De forskellige testcases er sorteret efter hvilket vindue, man står i.

Tabel 3: Funktionel test

Testnr.	Beskrivelse	Forventet resultat	Faktisk resultat
Login.aspx			
1a	Login uden brugernavn	Fejlmeddelelse	Som forventet
1b	Login uden password	Fejlmeddelelse	Som forventet
1c	Login hvor brugernavn og password ikke passer sammen	Fejlmeddelelse	Som forventet
1d	Login med korrekt brugernavn og password	`Forside.aspx` vises	Som forventet
Forside.aspx			
2a	Vælge 'Forside'	Siden genopfriskes	Som forventet
2b	Vælge 'Firma'	`Firma.aspx` vises	Som forventet
2c	Vælge 'Person'	`Person.aspx` vises	Som forventet
2d	Vælge 'Projekt'	`Projekt.aspx` vises	Som forventet
2e	Vælge 'Forecast'	`Forecast.aspx` vises	Som forventet
2f	Vælge 'Søg firma' uden at have indtastet nogle søgekriterier	Fejlmeddelelse	Som forventet
2g	Vælge 'Søg firma' med søgekriterier indtastet	Resultat af søgningen	Som forventet
2h	Vælge 'Søg alt' uden at have indtastet nogle søgekriterier	Fejlmeddelelse	Som forventet
2i	Vælge 'Søg alt' med søgekriterier indtastet	Resultat af søgningen	Som forventet
2j	Vælge et firma blandt resultaterne	`Firma.aspx` vises med de relevante oplysninger	Som forventet
2k	Vælge 'Nulstil firma'	Al indtastet tekst fjernes fra firmasøgefelterne Eventuelle resultater eller fejlmeddelelser fjernes	Som forventet
2l	Vælge 'Søg person' uden at have indtastet nogle søgekriterier	Fejlmeddelelse	Som forventet
2m	Vælge 'Søg person' med søgekriterier indtastet	Resultat af søgningen	Som forventet
2n	Vælge en person blandt resultaterne	`Person.aspx` vises med de relevante oplysninger	Som forventet
2o	Vælge 'Nulstil person'	Al indtastet tekst fjernes fra personsøgefelterne. Eventuelle resultater eller fejlmeddelelser fjernes	Som forventet

Fortsættes på næste side

Tabel 3 – fortsat fra forrige side

Testnr.	Beskrivelse	Forventet resultat	Faktisk resultat
2p	Vælge 'Søg projekt' uden at have indtastet nogle søgekriterier	Fejlmeddelelse	Som forventet
2q	Vælge 'Søg projekt' med søgekriterier indtastet	Resultat af søgningen	Som forventet
2r	Vælge et projekt blandt resultaterne	`Projekt.aspx` vises med de relevante oplysninger	Som forventet
2s	Vælge 'Nulstil projekt'	Al indtastet tekst fjernes fra personsøgefelterne . Eventuelle resultater eller fejlmeddelelser fjernes	Som forventet.
Firma.aspx			
3a	Vælge 'Forside'	`Forside.aspx` vises	Som forventet
3b	Vælge 'Firma'	Siden genopfriskes	Som forventet
3c	Vælge 'Person'	`Person.aspx` vises	Som forventet
3d	Vælge 'Projekt'	`Projekt.aspx` vises	Som forventet
3e	Vælge 'Forecast'	`Forecast.aspx` vises	Som forventet
3f	Vælge 'Søg' uden at have indtastet nogle søgekriterier	Fejlmeddelelse, søgefelterne vises med gul baggrund	Som forventet
3g	Vælge 'Søg' med søgekriterier indtastet	Resultat af søgningen	Som forventet
3h	Vælge 'Nulstil'	Al indtastet tekst fjernes fra firmasøgefelterne. Eventuelle resultater eller fejlmeddelelser fjernes	Som forventet
3i	Vælge et firma blandt resultaterne	Informationerne vises i tekstfelterne. Det valgte firma high-lightes	Som forventet
3j	Redigere et firma uden at have alle de nødvendige oplysninger	Fejlmeddelelse og dialogvindue vises	Som forventet
3k	Redigere et firma	Dialogvindue	Som forventet
3l	Ændringerne accepteres	Rettelserne bliver gemt i databasen	Som forventet
3m	Ændringerne accepteres ikke	Ingen ændringer gemt	Som forventet
3n	Oprette et nyt firma uden at have alle de nødvendige oplysninger	Fejlmeddelelse	Som forventet
3o	Oprette et nyt firma	Firmaet bliver gemt i databasen	Som forventet
3p	Vælge et projekt og trykke 'Ok'	`Projekt.aspx` vises med de relevante oplysninger	Som forventet
Person.aspx			
4a	Vælge 'Forside'	`Forside.aspx` vises	Som forventet
4b	Vælge 'Firma'	`Firma.aspx` vises	Som forventet
4c	Vælge 'Person'	Siden genopfriskes	Som forventet
4d	Vælge 'Projekt'	`Projekt.aspx` vises	Som forventet
4e	Vælge 'Forecast'	`Forecast.aspx` vises	Som forventet

Fortsættes på næste side

Tabel 3 – fortsat fra forrige side

Testnr.	Beskrivelse	Forventet resultat	Faktisk resultat
4f	Vælge 'Søg' uden at have indtastet nogle søgekriterier	Fejlmeddelelse, søgefelterne vises med gul baggrund	Som forventet
4g	Vælge 'Søg' med søgekriterier indtastet	Resultat af søgningen	Som forventet
4h	Vælge 'Nulstil'	Al indtastet tekst fjernes fra søgefelterne. Eventuelle resultater eller fejlmeddelelser fjernes	Som forventet
4i	Vælge en person blandt resultaterne	Informationerne vises i tekstfelterne. Den valgte person high-lightes	Som forventet
4j	Redigere en person uden at have alle de nødvendige oplysninger	Fejlmeddelelse og dialogvindue vises	Som forventet
4k	Redigere en person	Dialogvindue	Som forventet
4l	Ændringerne accepteres	Rettelserne bliver gemt i databasen	Som forventet
4m	Ændringerne accepteres ikke	Ingen ændringer gemt	Som forventet
4n	Oprette en ny person uden at have alle de nødvendige oplysninger	Fejlmeddelelse	Som forventet
4o	Oprette en ny person	Personen bliver gemt i databasen	Som forventet
4p	Vælge 'Ny'	Tekstbokse og knapper til brug ved oprettelse af ny event vises	Som forventet
4q	Vælge 'Søg event' uden nogle søgefelter indtastet	Fejlmeddelelse	Som forventet
4r	Vælge 'Søg event' med søgekriterier indtastet	Resultat af søgning	Som forventet
4s	Vælge 'Tilføj'	Den valgte event bliver tilføjet til personens liste	Som forventet
4t	Vælge 'Nulstil Event'	Al indtastet tekst fjernes fra eventsøgefelterne. Eventuelle resultater eller fejlmeddelelser fjernes	Som forventet
4u	Oprette en ny event uden at have de nødvendige oplysninger	Fejlmeddelese	Som forventet
4v	Oprette en ny event	Eventen bliver gemt i databasen	Som forventet
4w	Vælge et projekt og trykke 'Ok'	`Projekt.aspx` vises med de relevante oplysninger	Som forventet
4x	Vælge personens email-adresse	Brugerens email-program åbnes, klar til at sende en email til personen	Som forventet
Projekt.aspx			
5a	Vælge 'Forside'	`Forside.aspx` vises	Som forventet
5b	Vælge 'Firma'	`Firma.aspx` vises	Som forventet
5c	Vælge 'Person'	`Person.aspx` vises	Som forventet

Fortsættes på næste side

Tabel 3 – fortsat fra forrige side

Testnr.	Beskrivelse	Forventet resultat	Faktisk resultat
5d	Vælge 'Projekt'	Siden genopfriskes	Som forventet
5e	Vælge 'Forecast'	`Forecast.aspx` vises	Som forventet
5f	Vælge 'Søg' uden at have indtastet nogle søgekriterier	Fejlmeddelelse, søgefelterne vises med gul baggrund	Som forventet
5g	Vælge 'Søg' med søgekriterier indtastet	Resultat af søgningen	Som forventet
5h	Vælge 'Nulstil'	Al indtastet tekst fjernes fra søgefelterne. Eventuelle resultater eller fejlmeddelelser fjernes	Som forventet
5i	Vælge et projekt blandt resultaterne	Informationerne vises i tekstfelterne. Det valgte projekt high-lightes	Som forventet
5j	Redigere et projekt uden at have alle de nødvendige oplysninger	Fejlmeddelelse og dialogvindue vises	Som forventet
5k	Redigere et projekt	Dialogvindue	Som forventet
5l	Ændringerne accepteres	Rettelserne bliver gemt i databasen	Som forventet
5m	Ændringerne accepteres ikke	Ingen ændringer gemt	Som forventet
5n	Oprette et nyt projekt uden at have alle de nødvendige oplysninger	Fejlmeddelelse	Som forventet
5o	Oprette et nyt projekt	Projektet bliver gemt i databasen	Som forventet
Forecast.aspx			
6a	Vælge 'Forside'	`Forside.aspx` vises	Som forventet
6b	Vælge 'Firma'	`Firma.aspx` vises	Som forventet
6c	Vælge 'Person'	`Person.aspx` vises	Som forventet
6d	Vælge 'Projekt'	`Projekt.aspx` vises	Som forventet
6e	Vælge 'Forecast'	Siden genopfriskes	Som forventet
6f	Vælge 'Søg' uden at have indtastet nogle søgekriterier	Fejlmeddelelse	Som forventet
6g	Vælge 'Søg' med søgekriterier indtastet	Resultat af søgningen	Som forventet
6h	Vælge 'Nulstil'	Al indtastet tekst fjernes fra søgefelterne. Eventuelle resultater eller fejlmeddelelser fjernes	Som forventet
6i	Vælge et projekt fra listen	`Projekt.aspx` vises med de relevante oplysninger	Som forventet

Under testen var jeg ude for en enkelt fejl, som jeg ikke kunne rekonstruere, og som derfor ikke er nævnt i testskemaet. Da jeg skulle redigere et projekt, fik jeg fejlmeddelelsen: "Data is of wrong type" (testcase 5k). Jeg prøvede at gentage testen, for at se præcis hvad der var gået galt, men denne gang virkede det, og jeg kunne ikke fremprovokere fejlen. Jeg må derfor antage, at det ikke var en gennemgående fejl.

Den funktionelle test viser, at programmet virker som forventet.

E.2 Brugervenlighedstest

Det var meningen, jeg skulle have lavet en brugervenlighedstest ved et sælgermøde den 29. marts. Dette møde blev desværre aflyst i sidste øjeblik, og jeg havde derfor ikke mulighed for at lave så gennemgribende en brugervenlighedstest, som jeg egentlig havde planlagt. Et nyt møde er blevet fastlagt i begyndelsen af april.

Det er min hensigt at stille sælgerne følgende opgaver i brugervenlighedstesten:

- Find et specifikt firma udfra telefonnummeret.
- Find ud af, hvilket firma en bestemt person arbejder for.
- Se din egen forecast (hvis aktuelt).
- Find ud af, hvilke projekter DTI har hos et specifikt firma.
- Opret en ny person.
- Opret et forecast for et projekt.

Jeg udførte en mere uformel brugervenlighedstest tidligere i forløbet, hvor to sælgere udførte alle opgaver bortset fra de to sidste. Sælgerne havde ingen problemer med at finde rundt i programmet og var ikke i tvivl om, hvilken fremgangsmetode de skulle benytte, for at finde de ønskede resultater.

F Bilag

Der er vedlagt følgende bilag:

- Email fra Susanne Christoph
- Brugervejledning
- Programkode

G E-mail

Det følgende er en email vedrørende programmets udformning, som jeg modtog fra Susanne Christoph den 24. november 2004.

```
Hej Maria

Her følger 2. del af input vedr. CRM og Benny.

Der skal være 3 stamkort:
1.                  Firmastamkort
2.                  Personstamkort
3.                  Projektstamkort

Ad. 1:

Firmastamkortet fungerer både som oprettelses-, redigerings- og
søgebillede og indeholder informationer som er firmarelaterede med
minimum det sæt af information, som også fremgår af oversigtsbilledet
(det i 3 niveauer). Herudover kan der være et notefelt, hvor relevant
information på firmaniveau anføres. F.eks. på GateTrade kan en note
være, at det er PostDK, TDC, Maersk og Danske Bank, der har etableret
og finansieret firmaet. Det kan også være navne på
konsulentvirksomheder/konsulenter, der arbejder for firmaet eller
noter om aftaler med konkurrerenter.

På firmastamkortet skal endvidere være tilknyttet aktive
rammeaftaler/storkundeaftaler, som skal kunne tilgås direkte fra
firmastamkortet.

Firmastamkort skal desuden kunne tilgås direkte fra oversigtsbilledet.

Ad. 2:

Personstamkortet fungerer både som oprettelses-, redigerings- og
søgebillede og indeholder informationer, som er personrelaterede, dog
således at det fremgår, hvilket firma og afdeling, personen er ansat
i.

Derudover skal personstamkortet indeholde information om

Titel
ansvarsområde
Direkte telefonnr
e-mail adresse
evt. fax
interesseområder (ny info - skal forventeligt oprettes i kodeform,
```

kan muligvis implementeres som eventkode)
liste over samtlige eventskoder, der er tilknyttet personen
liste over samtlige projekter, som personen er tilknyttet

Tilsvarende firmastamkortet kan der være et notefelt, hvor relevant information på personniveau anføres. F.eks. fritidsinteresser, familiære forhold eller at personen indgår i firmaets IT-udvalg, sikkerhedsudvalg etc.

Personstamkortet skal endvidere kunne tilgås direkte fra oversigtsbilledet.

Ad. 3:

Projektstamkortet fungerer både som oprettelses-, redigerings- og søgebillede og indeholder informationer, som er projektrelaterede, dog således at det fremgår, hvilket firma projektet er knyttet til og hvilke personer, der er knyttet til projektet.

Derudover skal projektstamkortet indeholde følgende information:

Projektnavn
Beskrivelse
Oprettelsesdato
Projektansvarlig hos kunden
Projektansvarlig hos TI, hvis anden end sælger
Liste over øvrige interessenter/projektdeltagere med standardiseret rolleangivelse (f.eks. budgetansvarlig, brugerrepræsentant, rådg. konsulent)
Liste over TI projektdeltagere/interessenter med standardiseret rolleangivelse (teknisk projektleder, udvikler, koordinator, sponsor..)

Og endelig forecastinformation:

Forventet aftaledato (dato for ordre/kontrakt-indgåelse)
Total projektsum (tilbudssum eller forventet tilbuds/kontraktsum)
Forecast worst case - minimum for forventet aftaleindgåelse
Forecast best estimate - bedste gæt, det der forekommer mest sandsynligt som projektet står
Forecast best case - hvis alt lykkes
Ordresandsynlighed (interessehenvendelse 10%, behov, beslutningstager og budget identificeret 25%, tilbud afleveret og TI shortlistet som én af 2 50%, kontraktforhandlinger i gang med TI alene 75%, aftalen underskrevet af begge parter 100%, sagen tabt 0%.
Forventet leveranceplan månedsfordelt med beløb baseret på best estimate x ordresandsynligheden

Informationer skal kunne ændres med et revisionsspor, der angiver seneste ændring og hvem der har ændret info.

Dette skal kunne danne grundlag for en rapport, som i et givet datointerval baseret på closing dato og sælger ‘‘interval’’ giver en oversigt over:

Firma
Projektnavn
Total projektsum
Closing dato
Worst case
Best estimate
Best case
Sandsynlighed
Vægtet projektsum
Leveranceplan for efterfølgende 6-12 mdr.

Med hensyn til mere generelle og tværgående søgninger må det være muligt, at etablere en rapportgenerator, hvor vi kan vælge forskellige kriterier ud:

Nacekode interval
Postnumre
Titel
Aftaleforhold
Omsætningstal
Osv.

H Brugervejledning

Dette er brugervejledningen for DTI's CRM system. Programmet er et søge- og opdateringsprogram til DTI's kundedatabase. Det håndterer oplysninger om firmaer, kontaktpersoner og projekter.

H.1 Brug af programmet

Programmet ligger på websider på DTI's intranet. Den direkte adresse kan fås ved henvendelse til enten forfatteren[2] eller Gorm Bjerre.

Ved alle søgninger gælder det, at '%' kan bruges som 'wildcard'. "Maria%Elmvang" finder altså både "Maria Elmvang" og "Maria Miland Elmvang". Vær dog opmærksom på at "Maria % Elmvang" kun finder "Maria Miland Elmvang", da søgeprogrammet tager alle mellemrum bogstaveligt.

Alle de steder hvor man får en liste af resultater, kan disse resultater sorteres efter de forskellige kolonner ved at klikke på den givne kolonne.

Login.aspx

For at benytte programmet er det nødvendigt at logge på vha. et brugernavn og password. Brugernavn og password kan fås ved henvendelse til enten forfatteren eller Gorm Bjerre.

Forside.aspx

På forsiden har man adgang til de fire andre sider, **Firma**, **Person**, **Projekt** og **Forecast**. Til mere avancerede søgninger anbefales det, at man går direkte til den relevante side. Ønsker man at oprette eller redigere oplysninger, foregår dette også på den relevante side.

Forsiden kan benyttes til generelle og kombinerede søgninger. Alle felterne på forsiden kan bruges som søgefelter.

For at finde et specifikt firma, indtaster man de oplysninger, man har, og klikker på "Søg firma". Resultatet bliver vist umiddelbart under knapperne. Ud fra oplysningerne kan man vælge det firma, man vil vide mere om ved at klikke på "Vælg dette" ud for det givne firma.

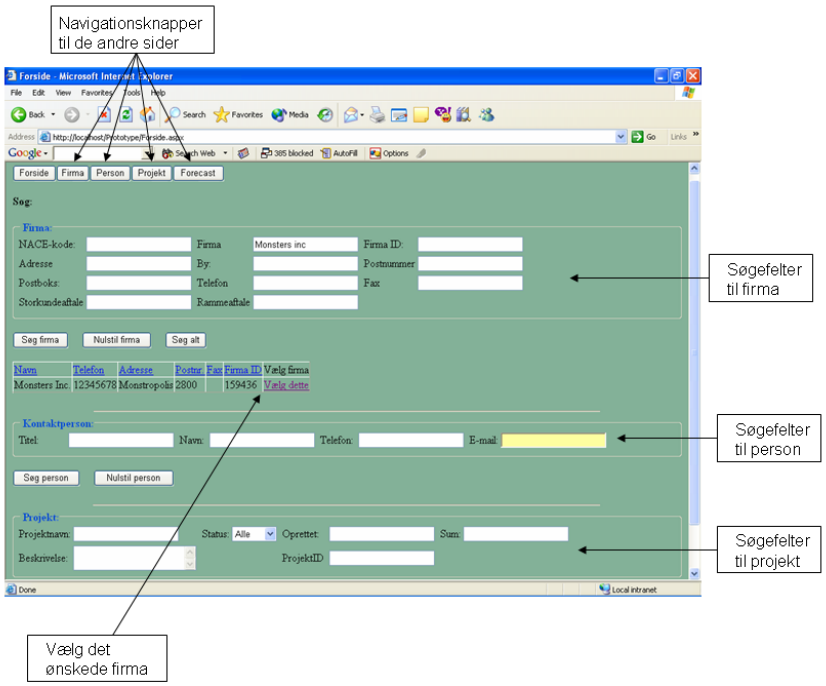
"Nulstil" tømmer alle tekstfelter og fjerner resultaterne.

Vælger man "Søg alt" i stedet for "Søg firma", får man også resultaterne på de personer, der arbejder for firmaet, og de projekter DTI er involveret i ved firmaet.

Fremgangsmåden er tilsvarende for at finde en bestemt person eller et bestemt projekt. For at præcisere søgningen kan man vælge at indtaste firmaets navn eller ID, så der kun søges efter personer/projekter i det relevante firma. Ved man ikke, hvilket firma personen/projektet tilhører, efterlader man blot disse felter tomme.

På grund af den måde datoer er gemt på i databasen, kan man desværre kun søge efter år, og ikke efter den præcise dato.

[2]Maria Elmvang: maria.elmvang@tondering.dk



Figur 41: `Forside.aspx`

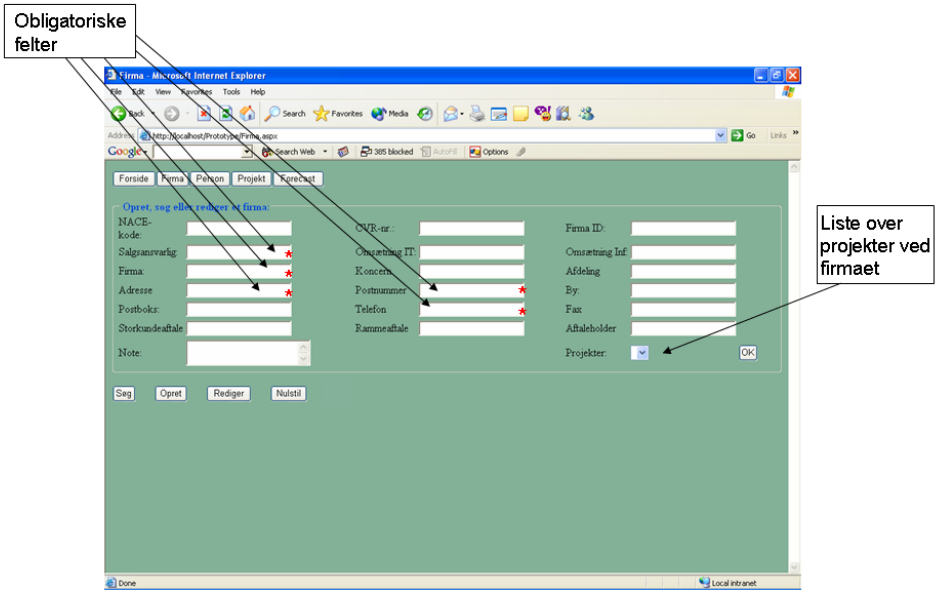
Firma.aspx

På firmasiden er det muligt at lave en mere detaljeret søgning efter et specifikt firma. Derudover kan man her også oprette eller redigere firmaer.

Søg: Søgningen foregår som på forsiden. Udover "Omsætning TI", "Omsætning Inf" og "CVR-nr." er alle felter søgbare. Hvis søgningen kun resulterer i ét svar, bliver oplysningerne med det samme vist i tekstfelterne. Er der flere svar, skal man som på forsiden vælge det ønskede firma fra listen. Hvis DTI har igangværende projekter ved firmaet, bliver disse vist i en 'dropdownlist', "Projekter". Man kan vælge at se flere oplysninger om et givet projekt ved at vælge det fra listen og trykke på "OK".

Opret: Ønsker man at oprette et nyt firma, *skal* felter markeret med en stjerne (*) i figur 42 udfyldes. Udover de obligatoriske felter kan man udfylde så mange eller så få oplysninger, som man måtte være i besiddelse af. Dog bliver de oplysninger, man måtte skrive i "NACE kode", "CVR-nr.", "Omsætning TI", "Omsætning Inf" og "Firma ID" ikke benyttet. Firmaet bliver gemt, når man klikker på "Opret".

Rediger: Ønsker man at redigere oplysningerne om et firma, skal man først søge efter firmaet, for at sikre, at man ikke ved en fejl kommer til at slette oplysninger, der ikke skulle være ændret. Også her skal felter markeret med en stjerne være udfyldt. Derudover kan man som ovenfor ændre alle oplysninger udover "NACE kode", "CVR-nr.", "Omsætning TI", "Omsætning Inf" og "Firma ID". Firmaets oplysninger bliver ændret, når man klikker på "Rediger" og bekræfter ændringerne.



Figur 42: `Firma.aspx`

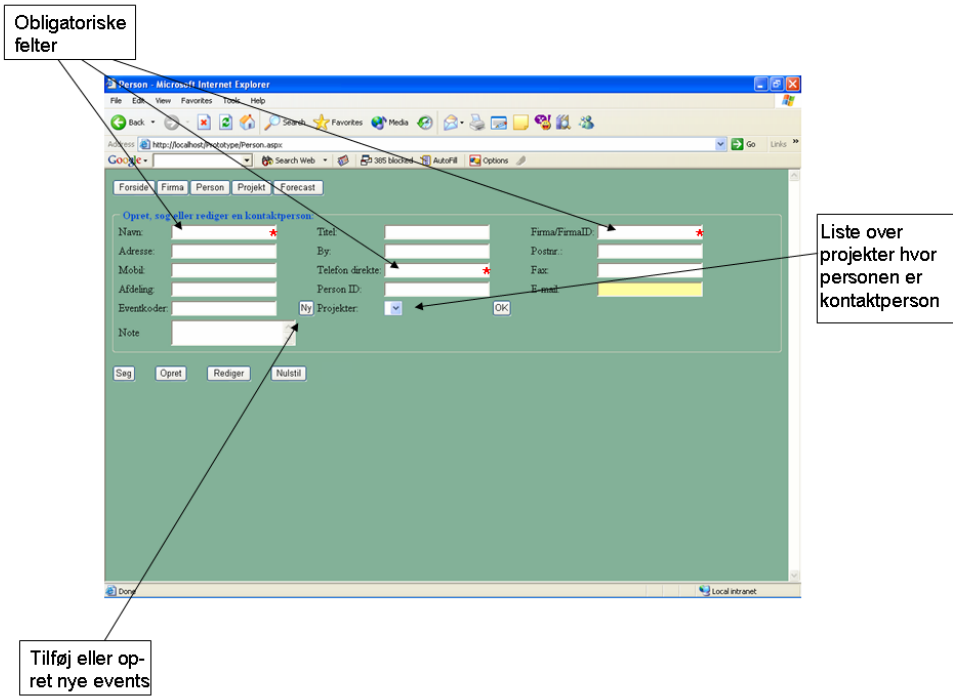
Person

På personsiden er det muligt at lave en mere detaljeret søgning efter en specifik person. Det gælder både kontaktpersoner ved firmaerne, og ansatte ved DTI. Derudover kan man her også oprette personer i databasen eller redigere eksisterende personoplysninger.

Søg: Søgningen foregår som på forsiden. Udover "Adresse", "By", "Postnr." og "Fax" er alle felter søgbare. Hvis søgningen kun resulterer i ét svar, bliver oplysningerne med det samme vist i tekstfelterne. Er der flere svar, skal man som på forsiden vælge den ønskede person fra listen. Hvis personen er kontaktperson for et projekt, bliver dette vist i en 'dropdownlist', "Projekter". Man kan vælge at se flere oplysninger om et givet projekt ved at vælge det fra listen og trykke på "OK". "Eventkode"-tekstboksen bliver til en 'dropdownlist', hvor personens events kan ses. Slutteligt bliver personens e-mail adresse vist både i en almindelig tekstboks og som et link. Hvis ens computer er sat op til det, kan man sende en e-mail til personen ved at klikke på det link.

Opret: Ønsker man at oprette en ny person, *skal* felter markeret med en stjerne (*) i figur 43 udfyldes. For at sikre mod inkonsistens i databasen skal man bruge firmaets ID i stedet for dets navn. Dette kan findes gennem `Firma.aspx`. Udover de obligatoriske felter kan man udfylde så mange eller så få oplysninger, som man måtte være i besiddelse af. Dog bliver oplysninger, man måtte skrive i "Adresse", "By", "Postnr.", "Fax", "Eventkoder" og "Person ID" ikke benyttet. Disse er nemlig knyttet til det firma, personen arbejder for, og ikke til personen selv. Personen bliver

gemt, når man klikker på "Opret". Det er først muligt at tilføje 'Eventkoder' (om personen skal have tilsendt julekort, katalog, mv.), efter personen er blevet oprettet.



Figur 43: `Person.aspx`

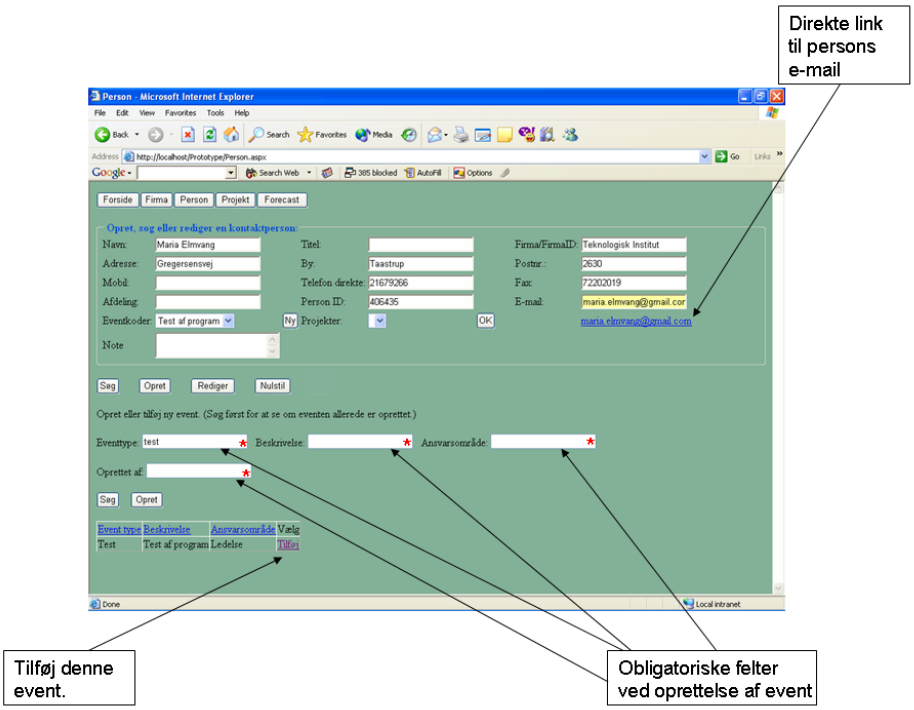
Rediger: Ønsker man at redigere oplysningerne om en person, skal man først søge efter personen for at sikre, at man ikke ved en fejl kommer til at slette oplysninger, der ikke skulle være ændret. Også her skal felter markeret med en stjerne være udfyldt. Derudover kan man ændre alle oplysninger udover de seks nævnt ovenfor. Skifter en person firma, skal man bruge firmaets ID i stedet for dets navn. Personens oplysninger bliver ændret, når man klikker på "Rediger" og bekræfter ændringerne.

"Adresse", "By", "Postnr." og "Fax" er tilknyttet et firma, ikke en person, og skal derfor ændres på `Firma.aspx`, hvis det er nødvendigt.

For at tilføje eller oprette eventkoder skal man klikke på "Ny" ud for "Eventkoder". Dette gør det muligt at søge efter en event eller oprette en ny hvis den ikke eksisterer allerede. Når man opretter en ny event, skal alle event-tekstbokse være udfyldt (se figur 44).

Når man har søgt efter/oprettet en event, bliver resultatet vist i en liste. Man kan nu tilføje en event til personens eventkodeliste ved at klikke på "Tilføj" ud for den ønskede event.

Eventkoder kan pt. ikke fjernes igen.



Figur 44: `Person.aspx` med event

Projekt.aspx

På projektsiden er det muligt at lave en mere detaljeret søgning efter et specifikt projekt. Derudover kan man her også oprette eller redigere projekter.

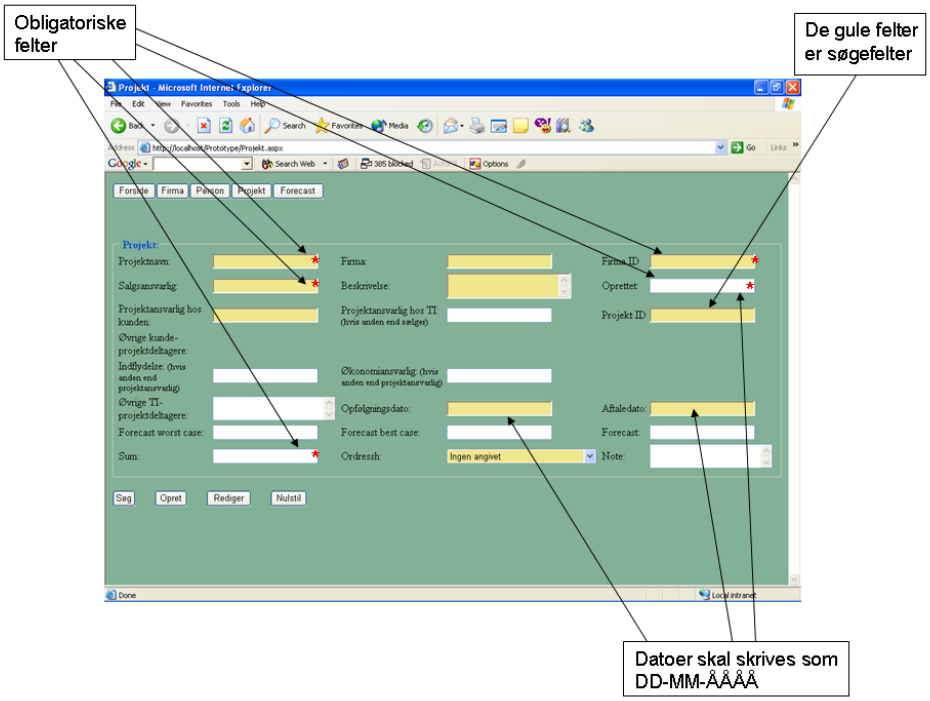
Søg: Søgningen foregår som på forsiden. Tekstfelterne vist med gult i figur 45 er søgbare. Hvis søgningen kun resulterer i ét svar, bliver oplysningerne med det samme vist i tekstfelterne. Er der flere svar, skal man som på forsiden vælge det ønskede projekt fra listen.

Opret:Ønsker man at oprette et nyt projekt, *skal* felter markeret med en stjerne (*) i figur 45 udfyldes. Udover de obligatoriske felter kan man udfylde så mange eller så få oplysninger, som man måtte være i besiddelse af. Dog bliver de oplysninger, man måtte skrive i "Projekt ID" ikke benyttet. Projektet bliver gemt når man klikker på "Opret".

Alle datoer skal skrives på formen DD-MM-ÅÅÅÅ (f.eks. 31-03-2005).

"Beskrivelse" bruges til beskrivelse af selve projektet; "Note" som logbog over ændringer af sum, forecast mv. F.eks. "Note: Sum ændret fra 0kr til 100kr 31-03-2005".

Rediger: Ønsker man at redigere oplysningerne om en projekt, skal man først søge efter projektet, for at sikre, at man ikke ved en fejl kommer til at slette oplysninger, der ikke skulle have været ændret. Også her skal felter markeret med en stjerne være udfyldt. Derudover kan man som ovenfor ændre alle oplysninger udover "ProjektID" og "Oprettet". Projektets oplysninger bliver ændret, når man klikker på "Rediger" og bekræfter ændringerne.

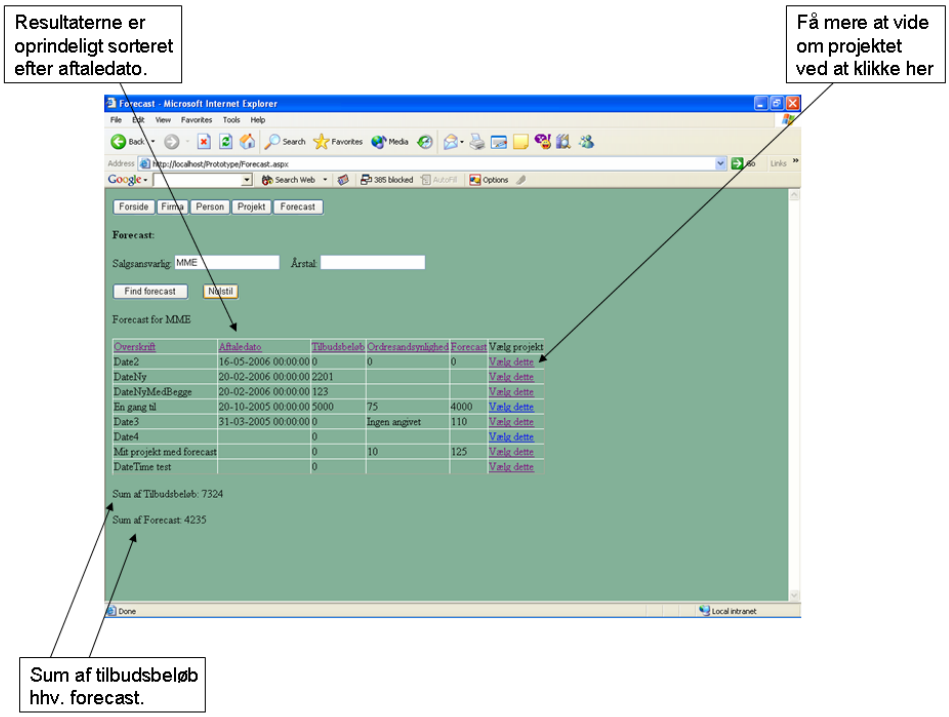


Figur 45: `Projekt.aspx`

Forecast.aspx

På forecastsiden kan man søge efter en sælgers projekter og derved få en liste over alle projekter, sælgeren er involveret i, og deres forecast. Ved søgning skal man skrive initialerne på den salgsansvarlige, men det er valgfrit, om man vil afgrænse søgningen ved brug af årstal eller ej.

For at finde en sælgers projekter, indtaster man oplysningerne og klikker på "Find forecast". Man får nu den relevante liste og kan ved hvert enkelt projekt få mere at vide om projektet ved at klikke på "Vælg dette". Nederst på siden står summen af alle forecasts og af alle tilbudsbeløb.



Figur 46: `Forecast.aspx`

I Programkode

I det følgende kan man læse koden bag programmet. Koden er opdelt efter filnavnene. 'Side.aspx' viser sidens HTML-opbygning. 'Side.aspx.cs' viser den bagvedliggende kode i C#.

Oplysninger om server, user id og password til databasen er blevet fjernet. Forbindelsen til SQL-serveren står derfor bare som `string connectionString;` uden detaljer.

I.1 Login.aspx

```
<%@ Page language="c#" Codebehind="Login.aspx.cs" AutoEventWireup="false"
   Inherits="Prototype.Login" %>
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN" >
<HTML>
   <HEAD>
      <title>Login</title>
      <meta name="GENERATOR" Content="Microsoft Visual Studio .NET 7.1">
      <meta name="CODE_LANGUAGE" Content="C#">
      <meta name="vs_defaultClientScript" content="JavaScript">
      <meta name="vs_targetSchema" content="http://schemas.microsoft.com/intellisense/ie5">
      <META http-equiv="Content-Type" content="text/html; charset=windows-1252">
      <meta content="Microsoft Visual Studio .NET 7.1" name="GENERATOR">
      <meta content="C#" name="CODE_LANGUAGE">
      <meta content="JavaScript" name="vs_defaultClientScript">
      <meta content="http://schemas.microsoft.com/intellisense/ie5" name="vs_targetSchema">
   </HEAD>
   <body bgColor="#83b198">
      <form id="Form2" runat="server">
         <P align="center"><FONT size="5"><STRONG>CRM-system</STRONG></FONT></P>
         <P align="center"><STRONG><FONT size="5"></FONT></STRONG> </P>
         <P align="center"><STRONG>Brugernavn:
               <asp:TextBox id="UserTB" runat="server"></asp:TextBox></STRONG></P>
         <P align="center"><STRONG>Password:
               <asp:TextBox id="PasswordTB" runat="server" TextMode="Password"></asp:TextBox></P>
         <P align="center">
            <asp:Button id="LoginBtn" runat="server" Text="Login"></asp:Button></P>
         <P align="center">
            <asp:Label id="Label1" runat="server" ForeColor="Red" Visible="False">
    Desværre, brugernavn og/eller password er ugyldigt.</asp:Label></P>
         </STRONG>
      </form>
   </body>
</HTML>
```

I.2 Login.aspx.cs

```
using System;
using System.Collections;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Web;
```

```
using System.Web.SessionState;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Web.UI.HtmlControls;
using System.Configuration;


namespace Prototype
{
   /// <summary>
   /// Summary description for Login.
   /// </summary>
   public class Login : System.Web.UI.Page
   {
      protected System.Web.UI.WebControls.TextBox UserTB;
      protected System.Web.UI.WebControls.TextBox PasswordTB;
      protected System.Web.UI.WebControls.Label Label1;
      protected System.Web.UI.WebControls.Button LoginBtn;

      private void Page_Load(object sender, System.EventArgs e)
      {
         // Put user code to initialize the page here
      }

      public string GetSpecificInfo(string info, string id)
      {
         string connectionString;
         System.Data.SqlClient.SqlConnection myConnection =
 new System.Data.SqlClient.SqlConnection(connectionString);
         string   myScalarQuery = "SELECT " + info + " FROM [KursusAnsvarlig]
WHERE [KursusAnsvarlig].KursusAnsvarlig = '" + id + "'";

         System.Data.SqlClient.SqlCommand myCommand =
new System.Data.SqlClient.SqlCommand(myScalarQuery, myConnection);
         myCommand.Connection.Open();
         string svar = myCommand.ExecuteScalar().ToString();
         myConnection.Close();

         return svar;
      }

      #region Web Form Designer generated code
      override protected void OnInit(EventArgs e)
      {
         //
         // CODEGEN: This call is required by the ASP.NET Web Form Designer.
         //
         InitializeComponent();
         base.OnInit(e);
      }

      /// <summary>
      /// Required method for Designer support - do not modify
      /// the contents of this method with the code editor.
      /// </summary>
      private void InitializeComponent()
```

```
      {
         this.LoginBtn.Click += new System.EventHandler(this.LoginBtn_Click);
         this.Load += new System.EventHandler(this.Page_Load);

      }
      #endregion

      private void LoginBtn_Click(object sender, System.EventArgs e)
      {
         string brugernavn = UserTB.Text;
         string kode = PasswordTB.Text;
         if(brugernavn.ToLower() != kode.ToLower())
         {
            Label1.Visible = true;
         }
         else
         {
            try
            {
               string centerkode = GetSpecificInfo("CenterKode", brugernavn);
               string status = GetSpecificInfo("Status", brugernavn);
               if(status == "OK" && centerkode == "110")
               {
                  Session["Login"] = brugernavn;
                  Response.Redirect("Forside.aspx");
               }
               else
               {
                  Label1.Visible = true;
               }
            }
            catch
            {
               Label1.Visible = true;
            }
         }
      }
   }
}
```

I.3 Forside.aspx

```
<%@ Page language="c#" Codebehind="Forside.aspx.cs" AutoEventWireup="false"
 Inherits="Prototype.Forside" %>
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN" >
<HTML>
   <HEAD>
      <title>Forside</title>
      <META http-equiv="Content-Type" content="text/html; charset=windows-1252">
      <meta content="Microsoft Visual Studio .NET 7.1" name="GENERATOR">
      <meta content="C#" name="CODE_LANGUAGE">
      <meta content="JavaScript" name="vs_defaultClientScript">
      <meta content="http://schemas.microsoft.com/intellisense/ie5" name="vs_targetSchema">
   </HEAD>
```

```
<body bgColor="#83b198">
   <form id="Form1" runat="server">
      <p>
         <!-- navigationsknapper -->
     <asp:button id="home" runat="server" CausesValidation="False" Text="Forside" />
     <asp:button id="FirmaBtn" runat="server" CausesValidation="False" Text="Firma" />
     <asp:button id="PersonBtn" runat="server" CausesValidation="False" Text="Person" />
     <asp:button id="ProjektBtn" runat="server" CausesValidation="False" Text="Projekt" />
     <asp:button id="ForecastBtn" runat="server" CausesValidation="False" Text="Forecast" />
     </p>
      <!-- Firmatabel -->
      <P><STRONG>Søg:</STRONG>
         <asp:label id="Label1" runat="server"></asp:label>
         <asp:label id="Label2" runat="server"></asp:label></P>
      <FIELDSET class="width780px"><LEGEND class="minifont"><B>Firma:</B>
         </LEGEND>
         <table class="minifont">
            <tr>
               <td>NACE-kode:</td>
               <td><asp:textbox id="NaceTB" runat="server"></asp:textbox></td>
               <td>Firma</td>
               <td><asp:textbox id="FirmaTB" runat="server"></asp:textbox></td>
               <td>Firma ID:</td>
               <td><asp:textbox id="FirmaIdTB" runat="server"></asp:textbox></td>
            </tr>
            <TR>
               <TD>Adresse</TD>
               <TD><asp:textbox id="AdresseTB" runat="server"></asp:textbox></TD>
               <TD>By:</TD>
               <TD><asp:textbox id="ByTB" runat="server"></asp:textbox></TD>
               <td></td>
               <td>Postnummer</td>
               <td><asp:textbox id="PostnrTB" runat="server"></asp:textbox></td>
            </TR>
            <tr>
               <td>Postboks:</td>
               <td><asp:textbox id="PostboksTB" runat="server"></asp:textbox></td>
               <TD>Telefon</TD>
               <TD><asp:textbox id="TelefonTB" runat="server"></asp:textbox></TD>
               <TD>Fax</TD>
               <TD><asp:textbox id="FaxTB" runat="server"></asp:textbox></TD>
            </tr>
            <TR>
               <TD>Storkundeaftale</TD>
               <TD><asp:textbox id="StorKundeTB" runat="server"></asp:textbox></TD>
               <TD>Rammeaftale</TD>
               <TD><asp:textbox id="RammeTB" runat="server"></asp:textbox></TD>
            </TR>
         </table>
      </FIELDSET>
      <!-- Søg + nulstil -->
      <P><asp:button id="SoegFirmaBtn" runat="server" CausesValidation="False" Text="Søg firma"
     </asp:button>    
     <asp:button id="NulstilFirma" runat="server" CausesValidation="False" Text="Nulstil firma"
     </asp:button>    
     <asp:button id="soegAltBtn" runat="server" CausesValidation="False" Text="Søg alt">
```

```
</asp:button></P>
 <P><asp:label id="Label3" runat="server"></asp:label></P>
 <P><asp:datagrid id="FirmaDG" runat="server" AutoGenerateColumns="False"
        AllowSorting="True" OnSortCommand="FirmaDG_sort">
       <SelectedItemStyle BackColor="Teal"></SelectedItemStyle>
       <Columns>
          <asp:BoundColumn DataField="Firmanavn1" SortExpression="FirmaNavn1"
        HeaderText="Navn"></asp:BoundColumn>
          <asp:BoundColumn DataField="telefonnr" SortExpression="telefonnr"
        HeaderText="Telefon"></asp:BoundColumn>
          <asp:BoundColumn DataField="Adresse1" SortExpression="Adresse1"
        HeaderText="Adresse"></asp:BoundColumn>
          <asp:BoundColumn DataField="Postnr" SortExpression="Postnr"
        HeaderText="Postnr."></asp:BoundColumn>
          <asp:BoundColumn DataField="Telefax" SortExpression="Telefax"
        HeaderText="Fax"></asp:BoundColumn>
          <asp:BoundColumn DataField="firmaid" SortExpression="firmaID"
        HeaderText="Firma ID"></asp:BoundColumn>
          <asp:ButtonColumn Text="V&#230;lg dette" HeaderText="V&#230;lg firma"
         CommandName="Select"></asp:ButtonColumn>
       </Columns>
    </asp:datagrid></P>
 <HR width="744" SIZE="2">
 <!-- Persontabel -->
 <fieldset class="width780px"><legend class="minifont"><b>Kontaktperson:</b>
    </legend>
    <table class="minifont">
       <tr>
          <td>Titel:</td>
          <td><asp:textbox id="TitelTB" runat="server"></asp:textbox></td>
          <td>Navn:</td>
          <td><asp:textbox id="NavnTB" runat="server"></asp:textbox></td>
          <td>Telefon:</td>
          <td><asp:textbox id="PersTelefonTB" runat="server"></asp:textbox></td>
          <td>E-mail:</td>
          <td><asp:textbox id="EmailTB" runat="server"></asp:textbox></td>
       </tr>
    </table>
 </fieldset>
 <!-- Søg + nulstil -->
 <P><asp:button id="soegPersonBtn" runat="server" CausesValidation="False"
        Text="Søg person"></asp:button>    
    <asp:button id="nulstilPerson" runat="server" CausesValidation="False"
        Text="Nulstil person"></asp:button></P>
 <P><asp:label id="Label4" runat="server"></asp:label></P>
 <P><asp:datagrid id="PersonDG" runat="server"
        AutoGenerateColumns="False" AllowSorting="True">
       <SelectedItemStyle BackColor="Teal"></SelectedItemStyle>
       <Columns>
          <asp:BoundColumn Visible="False" DataField="PersonID" HeaderText="PersonID"/>
          <asp:BoundColumn DataField="navn" SortExpression="Navn" HeaderText="Navn" />
          <asp:BoundColumn DataField="stilling" SortExpression="stilling"
        HeaderText="Stilling"></asp:BoundColumn>
          <asp:BoundColumn DataField="Afdeling" SortExpression="Afdeling"
        HeaderText="Afdeling"></asp:BoundColumn>
          <asp:BoundColumn DataField="telefonnr" SortExpression="telefonnr"
```

```
        HeaderText="Telefonnr."></asp:BoundColumn>
          <asp:HyperLinkColumn DataNavigateUrlField="email"
        DataNavigateUrlFormatString="mailto:{0}" DataTextField="email"
             SortExpression="email" HeaderText="E-mail"></asp:HyperLinkColumn>
          <asp:BoundColumn DataField="FirmaID" SortExpression="firmaID"
        HeaderText="Firma ID"></asp:BoundColumn>
          <asp:ButtonColumn Text="V&#230;lg denne" HeaderText="V&#230;lg person"
        CommandName="Select"></asp:ButtonColumn>
       </Columns>
    </asp:datagrid></P>
 <!-- Projekttabel -->
 <HR width="744" SIZE="2">
 <fieldset class="width780px"><legend class="minifont"><b>Projekt:</b>
    </legend>
    <table class="minifont">
       <tr>
          <td>Projektnavn:</td>
          <td><asp:textbox id="ProjektTB" runat="server"></asp:textbox></td>
          <td>Status:</td>
          <td><asp:dropdownlist id="StatusDropDownList" runat="server">
                <asp:ListItem Value="-10">Alle</asp:ListItem>
                <asp:ListItem Value="50">Aktiv</asp:ListItem>
                <asp:ListItem Value="100">Lukket</asp:ListItem>
                <asp:ListItem Value="0">Tabt</asp:ListItem>
             </asp:dropdownlist></td>
          <td>ProjektID</td>
          <td><asp:textbox id="ProjektidTB" runat="server"></asp:textbox></td>
          <td>Sum:</td>
          <td><asp:textbox id="SumTB" runat="server"></asp:textbox></td>
       </tr>
       <tr>
          <td>Beskrivelse:</td>
          <td><asp:textbox id="BeskrivelseTB" runat="server" TextMode="MultiLine" />
          </td>
       </tr>
    </table>
 </fieldset> <!-- Søg + nulstil -->
 <P>
    <asp:button id="soegProjektBtn" runat="server" CausesValidation="False"
Text="Søg projekt"></asp:button>    
    <asp:button id="nulstilProjekt" runat="server" CausesValidation="False"
Text="Nulstil projekt"></asp:button></P>
 <P>
    <asp:label id="Label5" runat="server"></asp:label></P>
 <P>
    <asp:datagrid id="ProjektDG" runat="server" AutoGenerateColumns="False"
AllowSorting="True">
       <SelectedItemStyle BackColor="Teal"></SelectedItemStyle>
       <Columns>
          <asp:BoundColumn DataField="FirmaLogbogID" HeaderText="Projekt ID" />
          <asp:BoundColumn DataField="Overskrift" SortExpression="Overskrift"
        HeaderText="Projektnavn"></asp:BoundColumn>
          <asp:BoundColumn DataField="LogTekst" SortExpression="LogTekst"
        HeaderText="Beskrivelse"></asp:BoundColumn>
          <asp:BoundColumn DataField="KundeKontakt" SortExpression="KundeKontakt"
        HeaderText="Kontaktperson"></asp:BoundColumn>
```

```
                <asp:BoundColumn DataField="firmaID" SortExpression="firmaID"
               HeaderText="Firma ID"></asp:BoundColumn>
                <asp:ButtonColumn Text="V&#230;lg dette" HeaderText="V&#230;lg projekt"
               CommandName="Select"></asp:ButtonColumn>
              </Columns>
            </asp:datagrid></P>
      </form>
   </body>
</HTML>
```

I.4 Forside.aspx.cs

```
using System;
using System.Collections;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Web;
using System.Web.SessionState;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Web.UI.HtmlControls;
using System.Configuration;

namespace Prototype
{
   /// <summary>
   /// Summary description for Forside.
   /// </summary>
   public class Forside : System.Web.UI.Page
   {
      // Navigation
      protected System.Web.UI.WebControls.Button home;
      protected System.Web.UI.WebControls.Button FirmaBtn;
      protected System.Web.UI.WebControls.Button PersonBtn;
      protected System.Web.UI.WebControls.Button ProjektBtn;
      protected System.Web.UI.WebControls.Button ForecastBtn;

      // Firma
      protected System.Web.UI.WebControls.Button NulstilFirma;
      protected System.Web.UI.WebControls.Button SoegFirmaBtn;
      protected System.Web.UI.WebControls.TextBox PostnrTB;
      protected System.Web.UI.WebControls.TextBox FirmaTB;
      protected System.Web.UI.WebControls.TextBox NaceTB;
      protected System.Web.UI.WebControls.TextBox AdresseTB;
      protected System.Web.UI.WebControls.TextBox FaxTB;
      protected System.Web.UI.WebControls.TextBox TelefonTB;
      protected System.Web.UI.WebControls.TextBox PostboksTB;
      protected System.Web.UI.WebControls.TextBox ByTB;
      protected System.Web.UI.WebControls.TextBox StorKundeTB;
      protected System.Web.UI.WebControls.TextBox RammeTB;
      protected System.Web.UI.WebControls.DataGrid FirmaDG;
      protected System.Web.UI.WebControls.TextBox FirmaIdTB;
      //Person
      protected System.Web.UI.WebControls.Button soegPersonBtn;
```

```
   protected System.Web.UI.WebControls.Button nulstilPerson;
   protected System.Web.UI.WebControls.TextBox TitelTB;
   protected System.Web.UI.WebControls.TextBox PersTelefonTB;
   protected System.Web.UI.WebControls.TextBox NavnTB;
   protected System.Web.UI.WebControls.TextBox EmailTB;
   protected System.Web.UI.WebControls.DataGrid PersonDG;
   // Projekt
   protected System.Web.UI.WebControls.Button soegProjektBtn;
   protected System.Web.UI.WebControls.Button nulstilProjekt;
   protected System.Web.UI.WebControls.TextBox ProjektTB;
   protected System.Web.UI.WebControls.TextBox SumTB;
   protected System.Web.UI.WebControls.TextBox BeskrivelseTB;
   protected System.Web.UI.WebControls.DataGrid ProjektDG;
   protected System.Web.UI.WebControls.DropDownList StatusDropDownList;

   protected System.Web.UI.WebControls.Label Label1;
   protected System.Web.UI.WebControls.TextBox ProjektidTB;
   protected System.Web.UI.WebControls.Label Label3;
   protected System.Web.UI.WebControls.Label Label4;
   protected System.Web.UI.WebControls.Label Label5;
   protected System.Web.UI.WebControls.Button soegAltBtn;
   protected System.Web.UI.WebControls.Label Label2;

   private void Page_Load(object sender, System.EventArgs e)
   {
      if(Session["Login"] == null)
      {
         Response.Redirect("Login.aspx");
      }
      else
      {
         TextBox[] TextBoxFirma = new TextBox[]{FirmaIdTB, PostboksTB,PostnrTB,ByTB,FirmaTB,
                                          NaceTB,AdresseTB,FaxTB,TelefonTB,
                                          StorKundeTB,RammeTB};
         TextBox[] TextBoxPerson = new TextBox[]{TitelTB, PersTelefonTB, NavnTB, EmailTB};
         TextBox[] TextBoxProjekt = new TextBox[]{ProjektTB, SumTB, BeskrivelseTB};

         for(int i = 0; i < TextBoxFirma.Length; i++)
         {
            SetDefaultButton(this.Page, TextBoxFirma[i], SoegFirmaBtn);
         }
         for(int i = 0; i < TextBoxPerson.Length; i++)
         {
            SetDefaultButton(this.Page, TextBoxPerson[i], soegPersonBtn);
         }
         for(int i = 0; i < TextBoxProjekt.Length; i++)
         {
            SetDefaultButton(this.Page, TextBoxProjekt[i], soegProjektBtn);
         }
      }
   }

System.Data.SqlClient.SqlDataReader GetFirmaInfo(string firmanavn, string firmaID,
     string telefon, string adresse, string postnr, string by, string postboks, string fax,
     string nace, string ramme, string storkunde, string sortby)
{
```

```
    if(firmaID == "")
    { firmaID = "%"; }
    string connectionString;

    System.Data.SqlClient.SqlConnection sqlConnection =
      new System.Data.SqlClient.SqlConnection(connectionString);

    string firmanavnAlt = findaa(firmanavn);
    string adresseAlt = findaa(adresse);
    string byAlt = findaa(by);

    string   queryString = "SELECT * FROM [Firma]
      WHERE ( [Firma].Firmanavn1 LIKE '%" + firmanavn + "%'
      OR [Firma].Firmanavn1 LIKE '%" + firmanavnAlt + "%')" +
      "AND [Firma].telefonnr LIKE '%" + telefon + "'" +
      "AND ( [Firma].Adresse1 LIKE '%" + adresse + "'
      OR [Firma].Adresse1 LIKE '%" + adresseAlt + "')" +
      "AND [Firma].Postnr LIKE '%" + postnr + "'" +
      "AND [Firma].FirmaID LIKE '" + firmaID + "'" +
      "AND [Firma].Postnr IN (SELECT Postnr FROM [Postnr]
      WHERE ( [Postnr].Bynavn LIKE '%" + by + "%'
      OR [Postnr].Bynavn LIKE '%" + byAlt + "%') )";
    if(fax != "")
    {
       queryString += "AND [Firma].Telefax LIKE '%" + fax + "'";
    }
    if(nace != "")
    {
       queryString += "AND [Firma].FirmaID IN (SELECT FirmaID From [FirmaKodeRegistreringer]
      WHERE [FirmaKodeRegistreringer].FirmaKodeID IN (SELECT FirmaKodeID FROM [FirmaKoder]
      WHERE [FirmaKoder].FirmaKodeTypeNavn = 'NACE'
      AND [FirmaKoder].FirmaKode LIKE '"+ nace +"'))";
    }
    if(ramme != "")
    {
       queryString += " AND [Firma].FirmaID IN (SELECT FirmaID FROM [FirmaAftale]
      WHERE [FirmaAftale].RammeAftale LIKE '%" + ramme + "%')";
    }
    if(storkunde != "")
    {
       queryString += " AND [Firma].FirmaID IN (SELECT FirmaID FROM [FirmaAftale]
      WHERE [FirmaAftale].StorKundeAftale LIKE '%" + storkunde + "%')";
    }
    if(postboks != "")
    {
       queryString += " AND [Firma].Postboks LIKE '%" + postboks + "%'";
    }
    queryString += " ORDER BY "+ sortby;
    System.Data.SqlClient.SqlCommand sqlCommand =
      new System.Data.SqlClient.SqlCommand(queryString, sqlConnection);
    sqlConnection.Open();
    System.Data.SqlClient.SqlDataReader dataReader =
      sqlCommand.ExecuteReader(System.Data.CommandBehavior.CloseConnection);
    return dataReader;
}
```

```
System.Data.SqlClient.SqlDataReader GetPersonInfo(string firma, string firmaID, string navn,
     string stilling, string telefonnr, string email, string sortby)
{
   if(firmaID == "")
   {
      firmaID = "%";
   }
   string connectionStringPerson;
   System.Data.SqlClient.SqlConnection sqlConnectionPerson =
     new System.Data.SqlClient.SqlConnection(connectionStringPerson);
   string personAlt = findaa(navn);
   string queryStringPerson = "SELECT * FROM [Person]
     WHERE ( [Person].Navn LIKE '%" + navn + "%'
     OR [Person].Navn LIKE '%" + personAlt + "%')" +
     "AND [Person].firmaID IN (SELECT FirmaID FROM [Firma]
     WHERE [Firma].Firmanavn1 LIKE '%" + firma + "%')" +
     "AND [Person].firmaID LIKE '" + firmaID + "'";
   if(email != "")
   {
         queryStringPerson += "AND [Person].email LIKE '%" + email + "%'";
   }
   if(stilling != "")
   {
      queryStringPerson += "AND [Person].stilling LIKE '%" + stilling + "%'";
   }
   if(telefonnr != "")
   {
      queryStringPerson += "AND [Person].telefonnr LIKE '%" + telefonnr + "'";
   }
   queryStringPerson += " ORDER BY "+ sortby;
   System.Data.SqlClient.SqlCommand sqlCommandPerson =
     new System.Data.SqlClient.SqlCommand(queryStringPerson, sqlConnectionPerson);
   sqlConnectionPerson.Open();
   System.Data.SqlClient.SqlDataReader dataReaderPerson =
     sqlCommandPerson.ExecuteReader(System.Data.CommandBehavior.CloseConnection);
   return dataReaderPerson;
}

   System.Data.SqlClient.SqlDataReader GetProjektInfo(string firma, string firmaID,
     string projekt, string projektID, string kontakt, string ordressh, string beskrivelse,
     string sortby)
   {
      if(firmaID == "")
      {
         firmaID = "%";
      }
      if(projektID == "")
      {projektID = "%";}
      string connectionString;
    System.Data.SqlClient.SqlConnection sqlConnection =
     new System.Data.SqlClient.SqlConnection(connectionString);

    string queryString = "SELECT * FROM [FirmaLogBog]
     WHERE [FirmaLogBog].Overskrift LIKE '%" + projekt + "%'" +
     "AND [FirmaLogBog].firmaID IN (SELECT FirmaID FROM [Firma]
     WHERE [Firma].Firmanavn1 LIKE '%" + firma + "%') "+
```

```
  "AND [FirmaLogBog].firmaID LIKE '" + firmaID + "'" +
  "AND [FirmaLogBog].FirmaLogbogID LIKE '" + projektID + "'";
  if(kontakt != "")
  {
      queryString += "AND [FirmaLogBog].KundeKontakt LIKE '%" + kontakt + "%'";
  }
  if(ordressh !="")
  {
     if(ordressh != "50")
     {
         queryString += " AND [FirmaLogBog].FirmaLogbogID IN (SELECT FirmaLogbogID
          FROM [Forecast] WHERE [Forecast].OrdreSSH LIKE '" + ordressh + "')";

     }
     else
     {
        queryString += " AND [FirmaLogBog].FirmaLogbogID IN (SELECT FirmaLogbogID
  FROM [Forecast] WHERE [Forecast].OrdreSSH <> '0' AND [Forecast].OrdreSSH <> '100')";
     }
   }
   if(beskrivelse != "")
   {
     queryString += " AND [FirmaLogBog].LogTekst LIKE '%" + beskrivelse + "%'";
   }
   queryString += "ORDER BY " + sortby;

   System.Data.SqlClient.SqlCommand sqlCommand =
  new System.Data.SqlClient.SqlCommand(queryString, sqlConnection);

   sqlConnection.Open();
   System.Data.SqlClient.SqlDataReader dataReader =
  sqlCommand.ExecuteReader(System.Data.CommandBehavior.CloseConnection);
   return dataReader;
}

#region Web Form Designer generated code
override protected void OnInit(EventArgs e)
{
   //
   // CODEGEN: This call is required by the ASP.NET Web Form Designer.
   //
   InitializeComponent();
   base.OnInit(e);
}

/// <summary>
/// Required method for Designer support - do not modify
/// the contents of this method with the code editor.
/// </summary>
private void InitializeComponent()
{
   this.home.Click += new System.EventHandler(this.home_Click);
   this.FirmaBtn.Click += new System.EventHandler(this.FirmaBtn_Click);
   this.PersonBtn.Click += new System.EventHandler(this.PersonBtn_Click);
   this.ProjektBtn.Click += new System.EventHandler(this.ProjektBtn_Click);
   this.ForecastBtn.Click += new System.EventHandler(this.ForecastBtn_Click);
```

```
   this.SoegFirmaBtn.Click += new System.EventHandler(this.SoegFirmaBtn_Click);
   this.NulstilFirma.Click += new System.EventHandler(this.nulstilFirma_Click);
   this.soegAltBtn.Click += new System.EventHandler(this.soegAltBtn_Click);
   this.FirmaDG.SortCommand +=
  new System.Web.UI.WebControls.DataGridSortCommandEventHandler(this.FirmaDG_sort);
   this.FirmaDG.SelectedIndexChanged +=
   new System.EventHandler(this.FirmaDG_SelectedIndexChanged);
   this.soegPersonBtn.Click += new System.EventHandler(this.soegPersonBtn_Click);
   this.nulstilPerson.Click += new System.EventHandler(this.nulstilPerson_Click);
   this.PersonDG.SortCommand +=
  new System.Web.UI.WebControls.DataGridSortCommandEventHandler(this.PersonDG_sort);
   this.PersonDG.SelectedIndexChanged +=
  new System.EventHandler(this.PersonDG_SelectedIndexChanged);
   this.soegProjektBtn.Click += new System.EventHandler(this.soegProjektBtn_Click);
   this.nulstilProjekt.Click += new System.EventHandler(this.nulstilProjekt_Click);
   this.ProjektDG.SortCommand +=
  new System.Web.UI.WebControls.DataGridSortCommandEventHandler(this.ProjektDG_sort);
   this.ProjektDG.SelectedIndexChanged +=
  new System.EventHandler(this.ProjektDG_SelectedIndexChanged);
   this.Load += new System.EventHandler(this.Page_Load);

}
#endregion


private void SoegFirmaBtn_Click(object sender, System.EventArgs e)
{
   Label1.Text = "";
   Label2.Text = "";
   Label3.Text = "";
   string firma = FirmaTB.Text;
   string telefon = TelefonTB.Text;
   string adresse = AdresseTB.Text;
   string postnr = PostnrTB.Text;
   string fax = FaxTB.Text;
   string by = ByTB.Text;
   string firmaID = FirmaIdTB.Text;
   string nace = NaceTB.Text;
   string postboks = PostboksTB.Text;
   string kontakt = NavnTB.Text;
   string ramme = RammeTB.Text;
   string storkunde = StorKundeTB.Text;

   if(firma+telefon+adresse+postnr+nace+fax+by+firmaID+ramme+storkunde+postboks != "")
   {
      FirmaDG.DataSource = GetFirmaInfo(firma, firmaID, telefon, adresse, postnr,
  by, postboks, fax, nace, ramme, storkunde, "Firmanavn1");
      FirmaDG.DataBind();
      FirmaDG.Visible = true;
      if(FirmaDG.Items.Count.Equals(0))
      {
         Label3.Text = "Søgningen resulterede ikke i nogle resultater.
          Hvis du er usikker på stavemåden, brug da % som wildcard.";
         Label3.ForeColor = Color.Red;
         FirmaDG.Visible = false;
      }
```

```
   }
   else
   {
      Label1.Text = "Skriv venligst en værdi i mindst ét søgefelt";
      Label1.ForeColor = Color.Red;
      FirmaDG.Visible = false;
   }
}

public void FirmaDG_sort(object sender, DataGridSortCommandEventArgs e)
{
   string firma = FirmaTB.Text;
   string telefon = TelefonTB.Text;
   string adresse = AdresseTB.Text;
   string postnr = PostnrTB.Text;
   string fax = FaxTB.Text;
   string by = ByTB.Text;
   string firmaID = FirmaIdTB.Text;
   string nace = NaceTB.Text;
   string ramme = RammeTB.Text;
   string storkunde = StorKundeTB.Text;
   string postboks = PostboksTB.Text;

   FirmaDG.DataSource = GetFirmaInfo(firma, firmaID, telefon, adresse, postnr, by,
   postboks, fax, nace, ramme, storkunde, e.SortExpression);
   FirmaDG.DataBind();
}

private void soegPersonBtn_Click(object sender, System.EventArgs e)
{
   Label1.Text = "";
   Label2.Text = "";
   Label4.Text = "";
   string firma = FirmaTB.Text;
   string navn = NavnTB.Text;
   string stilling = TitelTB.Text;
   string telefon = PersTelefonTB.Text;
   string email = EmailTB.Text;
   string firmaID = FirmaIdTB.Text;
   if(firma+navn+stilling+telefon+email+firmaID != "")
   {
      PersonDG.DataSource =
          GetPersonInfo(firma, firmaID, navn, stilling, telefon, email, "Navn");
      PersonDG.DataBind();
      PersonDG.Visible = true;
      if(PersonDG.Items.Count.Equals(0))
      {
         Label4.Text = "Søgningen resulterede ikke i nogle resultater.
          Hvis du er usikker på stavemåden, brug da % som wildcard.";
         Label4.ForeColor = Color.Red;
         PersonDG.Visible = false;
      }
   }
   else
   {
      Label1.Text = "Skriv venligst en værdi i mindst ét søgefelt";
```

```
      Label1.ForeColor = Color.Red;
      PersonDG.Visible = false;
   }
}

private void PersonDG_sort(object sender, DataGridSortCommandEventArgs e)
{
   string firma = FirmaTB.Text;
   string navn = NavnTB.Text;
   string stilling = TitelTB.Text;
   string telefon = PersTelefonTB.Text;
   string email = EmailTB.Text;
   string firmaID = FirmaIdTB.Text;
   PersonDG.DataSource =
  GetPersonInfo(firma, firmaID, navn, stilling, telefon, email, e.SortExpression);
   PersonDG.DataBind();
}

private void ProjektDG_sort(object sender, DataGridSortCommandEventArgs e)
{
   string firma = FirmaTB.Text;
   string projekt = ProjektTB.Text;
   string kontakt = NavnTB.Text;
   string firmaID = FirmaIdTB.Text;
   string projektid = ProjektidTB.Text;
   string beskrivelse = BeskrivelseTB.Text;
   string ordressh = StatusDropDownList.SelectedItem.Value;
   if(ordressh == "-10")
   {ordressh = "";}
   ProjektDG.DataSource = GetProjektInfo(firma, firmaID, projekt, projektid, kontakt,
  ordressh, beskrivelse, e.SortExpression);
   ProjektDG.DataBind();
}

private string findaa(string tekst)
{
   tekst = tekst.ToLower();
   if(tekst.IndexOf("aa") != -1)
   {
      tekst = tekst.Replace("aa", "å");
   }
   else
   {
      if(tekst.IndexOf("å") != -1)
      {
         tekst = tekst.Replace("å", "aa");
      }
   }
         return tekst;
}

private void FirmaBtn_Click(object sender, System.EventArgs e)
{
   Response.Redirect("Firma.aspx");
}
```

```
private void PersonBtn_Click(object sender, System.EventArgs e)
{
   Response.Redirect("Person.aspx");
}

private void ProjektBtn_Click(object sender, System.EventArgs e)
{
   Response.Redirect("Projekt.aspx");
}

private void ForecastBtn_Click(object sender, System.EventArgs e)
{
   Response.Redirect("Forecast.aspx");
}

private void home_Click(object sender, System.EventArgs e)
{
   Response.Redirect("Forside.aspx");
}


private void nulstilFirma_Click(object sender, System.EventArgs e)
{
   Label1.Text = "";
   Label2.Text = "";
   Label3.Text = "";
   TextBox[] TextBoxFirma = new TextBox[]{FirmaIdTB, PostboksTB ,PostnrTB,ByTB,FirmaTB,
                                    NaceTB,AdresseTB,FaxTB,TelefonTB,
                                    StorKundeTB,RammeTB};
   for(int i = 0; i < TextBoxFirma.Length; i++)
   {
      TextBoxFirma[i].Text = null;
   }
   FirmaDG.Visible = false;
}

private void nulstilPerson_Click(object sender, System.EventArgs e)
{
   Label1.Text = "";
   Label2.Text = "";
   Label4.Text = "";
   TextBox[] TextBoxPerson = new TextBox[]{TitelTB, PersTelefonTB, NavnTB, EmailTB};

   for(int i = 0; i < TextBoxPerson.Length; i++)
   {
      TextBoxPerson[i].Text = null;
   }
   PersonDG.Visible = false;
}

private void nulstilProjekt_Click(object sender, System.EventArgs e)
{
   Label1.Text = "";
   Label2.Text = "";
   Label5.Text = "";
   TextBox[] TextBoxProjekt = new TextBox[]{ProjektTB, SumTB, BeskrivelseTB};
```

```
   for(int i = 0; i < TextBoxProjekt.Length; i++)
   {
      TextBoxProjekt[i].Text = null;
   }
   ProjektDG.Visible = false;
}


private void soegProjektBtn_Click(object sender, System.EventArgs e)
{
   Label1.Text = "";
   Label2.Text = "";
   Label5.Text = "";
   string projekt = ProjektTB.Text;
   string firma = FirmaTB.Text;
   string kontakt = NavnTB.Text;
   string firmaID = FirmaIdTB.Text;
   string projektid = ProjektidTB.Text;
   string ordressh = StatusDropDownList.SelectedItem.Value;
   string beskrivelse = BeskrivelseTB.Text;
   if(ordressh == "-10")
   {ordressh = "";}
   if(projekt+firma+kontakt+firmaID+projektid+ordressh+beskrivelse != "")
   {
      ProjektDG.DataSource = GetProjektInfo(firma, firmaID, projekt, projektid, kontakt,
       ordressh, beskrivelse, "Overskrift");
      ProjektDG.DataBind();
      ProjektDG.Visible = true;
      if(ProjektDG.Items.Count.Equals(0))
      {
         Label5.Text = "Søgningen resulterede ikke i nogle resultater.
          Hvis du er usikker på stavemåden, brug da % som wildcard.";
         Label5.ForeColor = Color.Red;
         ProjektDG.Visible = false;
      }
   }
   else
   {
      Label1.Text = "Skriv venligst en værdi i mindst ét søgefelt";
      Label1.ForeColor = Color.Red;
      ProjektDG.Visible = false;
   }
}

private void FirmaDG_SelectedIndexChanged(object sender, System.EventArgs e)
{
   Session["Firma"] = FirmaDG.SelectedItem.Cells[5].Text;
   Response.Redirect("Firma.aspx");
}

private void PersonDG_SelectedIndexChanged(object sender, System.EventArgs e)
{
   Session["Person"] = PersonDG.SelectedItem.Cells[0].Text;
   Session["PersonFirma"] = PersonDG.SelectedItem.Cells[6].Text;
   Response.Redirect("Person.aspx");
```

```
}

private void ProjektDG_SelectedIndexChanged(object sender, System.EventArgs e)
{
   Session["Projekt"] = ProjektDG.SelectedItem.Cells[0].Text;
   Session["ProjektFirma"] = ProjektDG.SelectedItem.Cells[4].Text;
   Response.Redirect("Projekt.aspx");
}

public void SetDefaultButton(Page page, TextBox textControl, Button defaultButton)
{
   // Sets default buttons.
   // Originally created by Janus Kamp Hansen - http://www.kamp-hansen.dk
   // Extended by Darrell Norton - http://dotnetjunkies.com/weblog/darrell.norton/
   //   -- added Mozilla support, fixed a few issues, improved performance

   string theScript = @"

   <SCRIPT language=""javascript"">

   <!--

   function fnTrapKD(btn, event)
   {
      if (document.all)
      {
         if (event.keyCode == 13)
         {
            event.returnValue=false;
            event.cancel = true;
            btn.click();
         }
      }
      else if (document.getElementById)
      {
         if (event.which == 13)
         {
            event.returnValue=false;
            event.cancel = true;
            btn.click();
         }
      }
      else if(document.layers)
      {
         if(event.which == 13)
         {
            event.returnValue=false;
            event.cancel = true;
            btn.click();
         }
      }
   }
   // -->
   </SCRIPT>";

   Page.RegisterStartupScript("ForceDefaultToScript", theScript);
```

```
            textControl.Attributes.Add("onkeydown", "fnTrapKD(" + defaultButton.ClientID + ",event)");
        }


        private void soegAltBtn_Click(object sender, System.EventArgs e)
        {
            Label1.Text = "";
            Label2.Text = "";
            Label3.Text = "";
            string firma = FirmaTB.Text;
            string telefon = TelefonTB.Text;
            string adresse = AdresseTB.Text;
            string postnr = PostnrTB.Text;
            string fax = FaxTB.Text;
            string by = ByTB.Text;
            string firmaID = FirmaIdTB.Text;
            string nace = NaceTB.Text;
            string postboks = PostboksTB.Text;
            string kontakt = NavnTB.Text;
            string ramme = RammeTB.Text;
            string storkunde = StorKundeTB.Text;

            if(firma+telefon+adresse+postnr+nace+fax+by+firmaID+ramme+storkunde+postboks != "")
            {
             FirmaDG.DataSource = GetFirmaInfo(firma, firmaID, telefon, adresse, postnr, by,
                   postboks, fax, nace, ramme, storkunde, "Firmanavn1");
             FirmaDG.DataBind();
             FirmaDG.Visible = true;
             PersonDG.DataSource = GetPersonInfo(firma, firmaID, "", "", "", "", "Navn");
             PersonDG.DataBind();
             PersonDG.Visible = true;
             ProjektDG.DataSource =
GetProjektInfo(firma, firmaID, "", "", "", "", "", "Overskrift");
             ProjektDG.DataBind();
             ProjektDG.Visible = true;
             if(FirmaDG.Items.Count.Equals(0))
             {
                  Label3.Text = "Søgningen resulterede ikke i nogle resultater.";
                  Label3.ForeColor = Color.Red;
                  FirmaDG.Visible = false;
                  PersonDG.Visible = false;
                  ProjektDG.Visible = false;
             }
            }
            else
            {
               Label1.Text = "Skriv venligst en værdi i mindst ét søgefelt";
               Label1.ForeColor = Color.Red;
               FirmaDG.Visible = false;
               PersonDG.Visible = false;
               ProjektDG.Visible = false;
            }
        }
    }
}
```

I.5 Firma.aspx

```
<%@ Page language="c#" Codebehind="Firma.aspx.cs" AutoEventWireup="false"
Inherits="Prototype.FirmaOpret" %>
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN" >
<HTML>
   <HEAD>
      <title>Firma</title>
      <meta content="False" name="vs_showGrid">
      <META http-equiv="Content-Type" content="text/html; charset=windows-1252">
      <meta content="Microsoft Visual Studio .NET 7.1" name="GENERATOR">
      <meta content="C#" name="CODE_LANGUAGE">
      <meta content="JavaScript" name="vs_defaultClientScript">
      <meta content="http://schemas.microsoft.com/intellisense/ie5" name="vs_targetSchema">
   </HEAD>
   <body bgColor="#83b198">
      <form id="Form1" runat="server">
      <p>
      <!-- navigationsknapper -->
        <asp:button id="home" runat="server" CausesValidation="False" Text="Forside" />
        <asp:button id="FirmaBtn" runat="server" CausesValidation="False" Text="Firma" />
        <asp:button id="PersonBtn" runat="server" CausesValidation="False" Text="Person" />
        <asp:button id="ProjektBtn" runat="server" CausesValidation="False" Text="Projekt" />
        <asp:button id="ForecastBtn" runat="server" CausesValidation="False" Text="Forecast" />
        </p>
 <P>
         <P><asp:validationsummary id="ValidationSummary1" runat="server">
        </asp:validationsummary><asp:label id="Label1" runat="server"></asp:label></P>
         <div align="left"><form id="firmaData" action="_validate.asp"
        method="get" name="frmData">
        <fieldset class="width780px"><legend class="minifont"><b>
        <B>Søg, opret </B>eller rediger et firma:</b>
                  </legend>
                  <table class="minifont">
            <tr>
               <td>NACE-kode:</td>
               <td><asp:textbox id="NaceTB" runat="server"></asp:textbox></td>
               <td>CVR-nr.:</td>
               <td><asp:textbox id="CvrTextBox" runat="server"></asp:textbox></td>
               <td>Firma ID:</td>
               <td><asp:textbox id="FirmaIdTB" runat="server"></asp:textbox></td>
            </tr>
            <TR>
               <TD>Salgsansvarlig<FONT size="2">(initialer)</FONT>:</TD>
               <TD><asp:textbox id="SalgsTB" runat="server"></asp:textbox></TD>
               <TD><asp:requiredfieldvalidator id="SalgsValidator"
        runat="server" ErrorMessage="Indtast en salgsansvarlig." ControlToValidate="SalgsTB">
        salg*</asp:requiredfieldvalidator></TD>
               <TD>Omsætning IT:</TD>
               <TD><asp:textbox id="OmsTITB" runat="server"></asp:textbox></TD>
               <td></td>
               <td>Omsætning Inf:</td>
               <td><asp:textbox id="OmsInfTB" runat="server"></asp:textbox></td>
            </TR>
            <tr>
               <td>Firma:</td>
```

```
       <td><asp:textbox id="FirmaTB" runat="server"></asp:textbox></td>
       <td><asp:requiredfieldvalidator id="FirmaValidator" runat="server"
ErrorMessage="Indtast et firma" ControlToValidate="FirmaTB">Firma*
</asp:requiredfieldvalidator></td>
       <td>Koncern</td>
       <td><asp:textbox id="KoncernTB" runat="server"></asp:textbox></td>
       <td>Afdeling</td>
       <td><asp:textbox id="AfdelingTB" runat="server"></asp:textbox></td>
    </tr>
    <tr>
       <td>Adresse</td>
       <td><asp:textbox id="AdresseTB" runat="server"></asp:textbox></td>
       <td><asp:requiredfieldvalidator id="AdresseValidator" runat="server"
ErrorMessage="Indtast en adresse." ControlToValidate="AdresseTB">Adresse*
</asp:requiredfieldvalidator></td>
       <td>Postnummer</td>
       <td><asp:textbox id="PostnrTB" runat="server"></asp:textbox></td>
       <td><asp:requiredfieldvalidator id="PostnrValidator" runat="server"
ErrorMessage="Indtast et postnr." ControlToValidate="PostnrTB">Postnr*
</asp:requiredfieldvalidator></td>
       <td>By:</td>
       <td><asp:textbox id="ByTextBox" runat="server"></asp:textbox></td>
    </tr>
    <tr>
       <td>Postboks:</td>
       <td><asp:textbox id="PostboksTB" runat="server"></asp:textbox></td>
       <td>Telefon</td>
       <td><asp:textbox id="TelefonTB" runat="server"></asp:textbox></td>
       <td><asp:requiredfieldvalidator id="TelefonValidator" runat="server"
ErrorMessage="Indtast et telefonnr." ControlToValidate="TelefonTB">Telefon*
</asp:requiredfieldvalidator></td>
       <td>Fax</td>
       <td><asp:textbox id="FaxTB" runat="server"></asp:textbox></td>
    </tr>
    <tr>
       <td>Storkundeaftale</td>
       <td><asp:textbox id="StorKundeTextBox" runat="server"></asp:textbox></td>
       <td>Rammeaftale</td>
       <td><asp:textbox id="RammeTextBox" runat="server"></asp:textbox></td>
       <td></td>
       <td>Aftaleholder</td>
       <td><asp:textbox id="KontaktPersonTextBox" runat="server"></asp:textbox></td>
    </tr>
    <tr>
       <td>Note:</td>
       <td><asp:textbox id="NoteTextBox" runat="server" TextMode="MultiLine"></asp:textbox
       <td>Projekter:</td>
       <td><asp:dropdownlist id="ProjektDD" runat="server"
DataValueField="FirmaLogbogID" DataTextField="Overskrift"></asp:dropdownlist></td>
       <td><asp:button id="okBtn" runat="server" CausesValidation="False" Text="OK">
</asp:button></td>
    </tr>
         </table>
       </fieldset>
       <P><asp:button id="SoegFirmaBtn" runat="server" CausesValidation="False"
Text="Søg"></asp:button>      
```

```
                <asp:button id="OpretFirmaBtn" runat="server" Text="Opret"></asp:button>
                <asp:button id="RedigerFirmaBtn" runat="server" Text="Rediger"></asp:button>
                <asp:button id="nulstilFirmaBtn" runat="server" CausesValidation="False"
        Text="Nulstil"></asp:button>
               <P><asp:label id="Label2" runat="server"></asp:label></P>
               <P><asp:datagrid id="FirmaDG" runat="server" AutoGenerateColumns="False"
         AllowSorting="True">
            <SelectedItemStyle BackColor="Teal"></SelectedItemStyle>
            <Columns>
               <asp:BoundColumn Visible="False" DataField="FirmaID"
        HeaderText="FirmaID"></asp:BoundColumn>
               <asp:BoundColumn DataField="firmanavn1" SortExpression="firmanavn"
        HeaderText="Firma"></asp:BoundColumn>
               <asp:BoundColumn DataField="adresse1" SortExpression="adresse1"
        HeaderText="Adresse"></asp:BoundColumn>
               <asp:BoundColumn DataField="postnr" SortExpression="postnr"
        HeaderText="Postnr."></asp:BoundColumn>
               <asp:BoundColumn DataField="telefonnr" SortExpression="telefonnr"
        HeaderText="Telefon"></asp:BoundColumn>
               <asp:ButtonColumn Text="V&#230;lg dette" HeaderText="V&#230;lg firma"
        CommandName="Select"></asp:ButtonColumn>
            </Columns>
                  </asp:datagrid></P>
            </form>
         </div>
      </form>
      <asp:datagrid id="NaceDG" runat="server" AutoGenerateColumns="False">
         <Columns>
           <asp:BoundColumn DataField="FirmaKodeID" HeaderText="FirmaKodeID"></asp:BoundColumn>
           <asp:BoundColumn DataField="FirmaKode" HeaderText="FirmaKode"></asp:BoundColumn>
           <asp:BoundColumn DataField="FirmaKodeTypeNavn" HeaderText="FirmaKodeTypeNavn">
        </asp:BoundColumn>
         </Columns>
      </asp:datagrid>
   </body>
</HTML>
```

I.6 Firma.aspx.cs

```
using System;
using System.Collections;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Web;
using System.Web.SessionState;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Web.UI.HtmlControls;

namespace Prototype
{
   /// <summary>
   /// Summary description for FirmaOpret.
```

```
/// </summary>
public class FirmaOpret : System.Web.UI.Page
{
   protected System.Web.UI.WebControls.Button home;
   protected System.Web.UI.WebControls.ValidationSummary ValidationSummary1;
   protected System.Web.UI.WebControls.Button ProjektBtn;
   protected System.Web.UI.WebControls.Button PersonBtn;
   protected System.Web.UI.WebControls.Button ForecastBtn;
   protected System.Web.UI.WebControls.Button SoegFirmaBtn;
   protected System.Web.UI.WebControls.Button OpretFirmaBtn;
   protected System.Web.UI.WebControls.Button RedigerFirmaBtn;
   protected System.Web.UI.WebControls.Button nulstilFirmaBtn;
   protected System.Web.UI.WebControls.TextBox NaceTB;
   protected System.Web.UI.WebControls.TextBox SalgsTB;
   protected System.Web.UI.WebControls.TextBox KoncernTB;
   protected System.Web.UI.WebControls.TextBox OmsTITB;
   protected System.Web.UI.WebControls.TextBox FirmaTB;
   protected System.Web.UI.WebControls.TextBox OmsInfTB;
   protected System.Web.UI.WebControls.TextBox AfdelingTB;
   protected System.Web.UI.WebControls.TextBox KontaktPersonTextBox;
   protected System.Web.UI.WebControls.TextBox AdresseTB;
   protected System.Web.UI.WebControls.TextBox FaxTB;
   protected System.Web.UI.WebControls.TextBox NoteTextBox;
   protected System.Web.UI.WebControls.TextBox ByTextBox;
   protected System.Web.UI.WebControls.TextBox CvrTextBox;
   protected System.Web.UI.WebControls.TextBox TelefonTB;
   protected System.Web.UI.WebControls.TextBox StorKundeTextBox;
   protected System.Web.UI.WebControls.TextBox RammeTextBox;
   protected System.Web.UI.WebControls.Button FirmaBtn;
   protected System.Web.UI.WebControls.RequiredFieldValidator SalgsValidator;
   protected System.Web.UI.WebControls.RequiredFieldValidator FirmaValidator;
   protected System.Web.UI.WebControls.DataGrid FirmaDG;
   protected System.Web.UI.WebControls.TextBox PostnrTB;
   protected System.Web.UI.WebControls.Label Label1;
   protected System.Web.UI.WebControls.TextBox FirmaIdTB;
   protected System.Web.UI.WebControls.TextBox PostboksTB;
   protected System.Web.UI.WebControls.Label Label2;
   protected System.Web.UI.WebControls.DropDownList ProjektDD;
   protected System.Web.UI.WebControls.Button okBtn;
   protected System.Web.UI.WebControls.RequiredFieldValidator AdresseValidator;
   protected System.Web.UI.WebControls.RequiredFieldValidator PostnrValidator;
   protected System.Web.UI.WebControls.RequiredFieldValidator TelefonValidator;
   protected System.Web.UI.WebControls.DataGrid NaceDG;


   System.Data.SqlClient.SqlDataReader GetInfo(string firmanavn, string firmaid,
     string afdeling, string koncern, string telefon, string adresse, string postnr,
     string by, string postboks, string fax, string nace, string ramme, string storkunde,
     string note, string salgsansvarlig, string sortby)
   {
     string connectionString;
     System.Data.SqlClient.SqlConnection sqlConnection =
     new System.Data.SqlClient.SqlConnection(connectionString);
     string firmanavnAlt = findaa(firmanavn);
     string adresseAlt = findaa(adresse);
     string byAlt = findaa(by);
```

```
if(firmaid == "")
{firmaid = "%";}
string queryString = "SELECT * FROM [Firma] WHERE ( [Firma].Firmanavn1 LIKE
'%" + firmanavn + "%' OR [Firma].Firmanavn1 LIKE '%" + firmanavnAlt + "%')" +
 "AND [Firma].telefonnr LIKE '%" + telefon + "'" +
 "AND ([Firma].Adresse1 LIKE '%" + adresse + "'
  OR [Firma].Adresse1 LIKE '%" + adresseAlt + "')" +
 "AND [Firma].Postnr LIKE '%" + postnr + "'" +
 "AND [Firma].FirmaID LIKE '" + firmaid + "'" +
 "AND [Firma].Postnr IN (SELECT Postnr FROM [Postnr] WHERE ( [Postnr].Bynavn LIKE
  '%" + by + "%' OR [Postnr].Bynavn LIKE '%" + byAlt + "%') )";
if(fax != "")
{
   queryString += "AND [Firma].Telefax LIKE '%" + fax + "'";
}
if(nace != "")
{
  queryString += "AND [Firma].FirmaID IN (SELECT FirmaID From [FirmaKodeRegistreringer]
WHERE [FirmaKodeRegistreringer].FirmaKodeID IN (SELECT FirmaKodeID FROM [FirmaKoder]
WHERE [FirmaKoder].FirmaKodeTypeNavn = 'NACE'
AND [FirmaKoder].FirmaKode = '"+ nace +"'))";
}
if(ramme != "")
{
   queryString += " AND [Firma].FirmaID IN (SELECT FirmaID FROM [FirmaAftale]
WHERE [FirmaAftale].RammeAftale LIKE '%" + ramme + "%')";
}
if(storkunde != "")
{
   queryString += " AND [Firma].FirmaID IN (SELECT FirmaID FROM [FirmaAftale]
WHERE [FirmaAftale].StorKundeAftale LIKE '%" + storkunde + "%')";
}
if(note != "")
{
   queryString += " AND [Firma].FirmaID IN (SELECT FirmaID FROM [FirmaAftale]
WHERE [FirmaAftale].Note LIKE '%" + note + "%')";
}
if(postboks != "")
{
   queryString += " AND [Firma].Postboks LIKE '%" + postboks + "%'";
}
if(salgsansvarlig != "")
{
   queryString += " AND [Firma].salgsKontakt LIKE '%" + salgsansvarlig + "%'";
}
if(koncern != "")
{
   queryString += " AND [Firma].FirmaID IN (SELECT FirmaID FROM [FirmaKoncern]
WHERE [FirmaKoncern].Koncern LIKE '%" + koncern + "%')";
}
if(afdeling != "")
{
   queryString += " AND [Firma].FirmaID IN (SELECT FirmaID FROM [FirmaKoncern]
WHERE [FirmaKoncern].Afdeling LIKE '%" + afdeling + "%')";
}
queryString += " ORDER BY "+ sortby;
```

```
  System.Data.SqlClient.SqlCommand sqlCommand =
  new System.Data.SqlClient.SqlCommand(queryString, sqlConnection);

  sqlConnection.Open();
  System.Data.SqlClient.SqlDataReader dataReader =
  sqlCommand.ExecuteReader(System.Data.CommandBehavior.CloseConnection);
  return dataReader;
}


System.Data.SqlClient.SqlDataReader GetProjekt(string firmaid)
{
  string connectionString;
  System.Data.SqlClient.SqlConnection sqlConnection =
  new System.Data.SqlClient.SqlConnection(connectionString);
  string   queryString = "SELECT * FROM [FirmaLogBog]
  WHERE [FirmaLogBog].FirmaID = '" + firmaid +"'";
  System.Data.SqlClient.SqlCommand sqlCommand =
  new System.Data.SqlClient.SqlCommand(queryString, sqlConnection);
  sqlConnection.Open();
  System.Data.SqlClient.SqlDataReader dataReader =
  sqlCommand.ExecuteReader(System.Data.CommandBehavior.CloseConnection);
  return dataReader;
}

int insertFirma(string firmanavn, string telefonnr, string adresse, string postnr,
  string postboks, string telefax, string ejer)
{
  string connectionString;
  System.Data.SqlClient.SqlConnection sqlConnection =
  new System.Data.SqlClient.SqlConnection(connectionString);
  string queryString = "INSERT INTO [Firma] ([Firmanavn1], [Adresse1], [Telefonnr],
  [Telefax], [Postnr], [Postboks], [salgsKontakt], [OprettetAf]) VALUES (@firmanavn,
  @adresse, @telefonnr, @telefax, @postnr, @postboks, @ejer, @ejer)";
  System.Data.SqlClient.SqlCommand sqlCommand =
  new System.Data.SqlClient.SqlCommand(queryString, sqlConnection);
  sqlCommand.Parameters.Add("@firmanavn", System.Data.SqlDbType.VarChar).Value=firmanavn;
  sqlCommand.Parameters.Add("@telefonnr", System.Data.SqlDbType.VarChar).Value=telefonnr;
  sqlCommand.Parameters.Add("@adresse", System.Data.SqlDbType.VarChar).Value = adresse;
  sqlCommand.Parameters.Add("@postnr", System.Data.SqlDbType.VarChar).Value = postnr;
  sqlCommand.Parameters.Add("@postboks", System.Data.SqlDbType.VarChar).Value = postboks;
  sqlCommand.Parameters.Add("@telefax", System.Data.SqlDbType.VarChar).Value = telefax;
  sqlCommand.Parameters.Add("@ejer", System.Data.SqlDbType.VarChar).Value = ejer;
     int rowsAffected = 0;
     sqlConnection.Open();
     try
     {
        rowsAffected = sqlCommand.ExecuteNonQuery();
     }
     finally
     {
        sqlConnection.Close();
     }
     return rowsAffected;
  }
```

```
      int updateFirma(string firmaid, string firmanavn, string telefonnr, string adresse,
        string postnr, string postboks, string telefax, string ejer)
      {
        string connectionString;
        System.Data.SqlClient.SqlConnection sqlConnection =
        new System.Data.SqlClient.SqlConnection(connectionString);
        string queryString = "UPDATE [Firma] SET [FirmaNavn1] = @Navn, [Adresse1]=@Adresse,
        [Telefonnr]=@Telefonnr, [Postnr]=@Postnr, [Postboks]=@Postboks, [Telefax]=@Telefax,
        [salgsKontakt]=@Ejer, [OpdateretAf]=@Opdateretaf, [Opdateret]=@Opdateret"+
           " WHERE [Firma].FirmaID = '" + firmaid + "'";
        System.Data.SqlClient.SqlCommand sqlCommand =
        new System.Data.SqlClient.SqlCommand(queryString, sqlConnection);
        sqlCommand.Parameters.Add("@Navn", System.Data.SqlDbType.VarChar).Value = firmanavn;
        sqlCommand.Parameters.Add("@Adresse", System.Data.SqlDbType.VarChar).Value = adresse;
        sqlCommand.Parameters.Add("@Telefonnr", System.Data.SqlDbType.VarChar).Value=telefonnr;
        sqlCommand.Parameters.Add("@Postnr", System.Data.SqlDbType.VarChar).Value = postnr;
        sqlCommand.Parameters.Add("@Postboks", System.Data.SqlDbType.VarChar).Value = postboks;
        sqlCommand.Parameters.Add("@Telefax", System.Data.SqlDbType.VarChar).Value = telefax;
        sqlCommand.Parameters.Add("@Ejer", System.Data.SqlDbType.VarChar).Value = ejer;
        sqlCommand.Parameters.Add("@Opdateretaf", System.Data.SqlDbType.VarChar).Value =
                Session["Login"]+"";
        sqlCommand.Parameters.Add("@Opdateret", System.Data.SqlDbType.DateTime).Value =
                DateTime.Now;

        int rowsAffected = 0;
        sqlConnection.Open();
        try
        {
           rowsAffected = sqlCommand.ExecuteNonQuery();
        }
        finally
        {
           sqlConnection.Close();
        }

        return rowsAffected;
      }

      int insertFirmaAftale(string firmaid, string storkunde, string rammeaftale,
        string aftaleholder, string note)
      {
        string connectionString;
        System.Data.SqlClient.SqlConnection sqlConnection =
new System.Data.SqlClient.SqlConnection(connectionString);
        string queryString = "INSERT INTO [FirmaAftale] ([FirmaID], [StorKundeAftale],
        [Ramme], [AftaleHolder], [Note]) VALUES (@Firmaid, @Storkunde,
        @Rammeaftale, @AHolder, @Note)";
        System.Data.SqlClient.SqlCommand sqlCommand =
        new System.Data.SqlClient.SqlCommand(queryString, sqlConnection);
        sqlCommand.Parameters.Add("@Storkunde", System.Data.SqlDbType.VarChar).Value=storkunde;
        sqlCommand.Parameters.Add("@Ramme", System.Data.SqlDbType.VarChar).Value = rammeaftale;
        sqlCommand.Parameters.Add("@Note", System.Data.SqlDbType.VarChar).Value = note;
       sqlCommand.Parameters.Add("@AHolder", System.Data.SqlDbType.VarChar).Value=aftaleholder;
        sqlCommand.Parameters.Add("@Firmaid", System.Data.SqlDbType.Int).Value = firmaid;
        int rowsAffected = 0;
```

```
  sqlConnection.Open();
  try
  {
     rowsAffected = sqlCommand.ExecuteNonQuery();
  }
  finally
  {
     sqlConnection.Close();
  }
  return rowsAffected;
}

int updateFirmaAftale(string firmaid, string storkunde, string rammeaftale,
  string aftaleholder, string note)
{
  string connectionString;
  System.Data.SqlClient.SqlConnection sqlConnection =
  new System.Data.SqlClient.SqlConnection(connectionString);
  string queryString = "UPDATE [FirmaAftale] SET [StorKundeAftale] = @StorKunde,
  [RammeAftale]=@Ramme, [AftaleHolder]=@Aholder, [Note]=@Note "+
  " WHERE [FirmaAftale].FirmaID = '" + firmaid + "'";
  System.Data.SqlClient.SqlCommand sqlCommand =
  new System.Data.SqlClient.SqlCommand(queryString, sqlConnection);
  sqlCommand.Parameters.Add("@StorKunde", System.Data.SqlDbType.VarChar).Value=storkunde;
  sqlCommand.Parameters.Add("@Ramme", System.Data.SqlDbType.VarChar).Value = rammeaftale;
  sqlCommand.Parameters.Add("@Aholder", System.Data.SqlDbType.VarChar).Value = aftaleholder;
  sqlCommand.Parameters.Add("@Note", System.Data.SqlDbType.VarChar).Value = note;
  int rowsAffected = 0;
  sqlConnection.Open();
  try
  {
     rowsAffected = sqlCommand.ExecuteNonQuery();
  }
  finally
  {
     sqlConnection.Close();
  }

  return rowsAffected;
}

int insertFirmaKoncern(string firmaid, string koncern, string afdeling)
{
  string connectionString´;
  System.Data.SqlClient.SqlConnection sqlConnection =
  new System.Data.SqlClient.SqlConnection(connectionString);
  string queryString = "INSERT INTO [FirmaKoncern] ([FirmaID], [Koncern], [Afdeling])
  VALUES (@Firmaid, @Koncern, @Afdeling)";
  System.Data.SqlClient.SqlCommand sqlCommand =
  new System.Data.SqlClient.SqlCommand(queryString, sqlConnection);
  sqlCommand.Parameters.Add("@Koncern", System.Data.SqlDbType.VarChar).Value = koncern;
  sqlCommand.Parameters.Add("@Afdeling", System.Data.SqlDbType.VarChar).Value = afdeling;
  sqlCommand.Parameters.Add("@Firmaid", System.Data.SqlDbType.Int).Value = firmaid;
  int rowsAffected = 0;
  sqlConnection.Open();
  try
```

```
        {
           rowsAffected = sqlCommand.ExecuteNonQuery();
        }
        finally
        {
           sqlConnection.Close();
        }
        return rowsAffected;
      }

      int updateFirmaKoncern(string firmaid, string koncern, string afdeling)
      {
        string connectionString;
        System.Data.SqlClient.SqlConnection sqlConnection =
        new System.Data.SqlClient.SqlConnection(connectionString);
        string queryString = "UPDATE [FirmaKoncern] SET [Koncern] = @Koncern,
[Afdeling]=@Afdeling
        WHERE [FirmaKoncern].FirmaID = '" + firmaid + "'";
        System.Data.SqlClient.SqlCommand sqlCommand =
        new System.Data.SqlClient.SqlCommand(queryString, sqlConnection);
        sqlCommand.Parameters.Add("@Koncern", System.Data.SqlDbType.VarChar).Value = koncern;
        sqlCommand.Parameters.Add("@Afdeling", System.Data.SqlDbType.VarChar).Value = afdeling;
        int rowsAffected = 0;
        sqlConnection.Open();
        try
        {
           rowsAffected = sqlCommand.ExecuteNonQuery();
        }
        finally
        {
           sqlConnection.Close();
        }
        return rowsAffected;
      }

      public string GetSpecificInfo(string info, string id, string tabel, string kolonne)
      {
        string connectionString;
        System.Data.SqlClient.SqlConnection myConnection =
        new System.Data.SqlClient.SqlConnection(connectionString);
        string   myScalarQuery = "SELECT " + info + " FROM " + tabel +
        " WHERE ["+tabel+"]." + kolonne + " = '" + id + "'";
        System.Data.SqlClient.SqlCommand myCommand =
        new System.Data.SqlClient.SqlCommand(myScalarQuery, myConnection);
        myCommand.Connection.Open();
        string svar = myCommand.ExecuteScalar().ToString();
        myConnection.Close();
        return svar;
      }

      System.Data.SqlClient.SqlDataReader GetNace(string firmaid)
      {
        string connectionString;
        System.Data.SqlClient.SqlConnection sqlConnection =
        new System.Data.SqlClient.SqlConnection(connectionString);
        string   queryString = "SELECT * FROM FirmaKoder
```

```
  WHERE [FirmaKoder].FirmaKodeTypeNavn = 'NACE' " +
  "AND [FirmaKoder].FirmaKodeID IN (SELECT FirmaKodeID FROM FirmaKodeRegistreringer
  WHERE [FirmaKodeRegistreringer].FirmaID = '" + firmaid + "')";
  System.Data.SqlClient.SqlCommand sqlCommand =
  new System.Data.SqlClient.SqlCommand(queryString, sqlConnection);
  sqlConnection.Open();
  System.Data.SqlClient.SqlDataReader dataReader =
  sqlCommand.ExecuteReader(System.Data.CommandBehavior.CloseConnection);
  return dataReader;
}

private void Page_Load(object sender, System.EventArgs e)
{
  if(Session["Login"] == null)
  {
     Response.Redirect("Login.aspx");
  }
  else
  {
     if(!Page.IsPostBack)
     {
        if(Session["Firma"] != null)
        {
          string firmaid = ""+Session["Firma"];
          findFirma(firmaid);
        }
     }
     TextBox[] webControls = new TextBox[]{FirmaTB, NaceTB, SalgsTB, KoncernTB, OmsTITB,
                 OmsInfTB,AfdelingTB,KontaktPersonTextBox,AdresseTB,
                 FaxTB,NoteTextBox,ByTextBox,CvrTextBox,TelefonTB,
                 StorKundeTextBox,RammeTextBox, PostnrTB, FirmaIdTB, PostboksTB};

     for(int i = 0; i < webControls.Length; i++)
     {
        SetDefaultButton(this.Page, webControls[i], SoegFirmaBtn);
     }

     TextBox[] ejBrugt = new TextBox[]{OmsTITB,OmsInfTB, CvrTextBox};
     for(int i = 0; i < ejBrugt.Length; i++)
     {
        ejBrugt[i].BackColor = Color.Gainsboro;
     }

     RedigerFirmaBtn.Attributes.Add("onclick", "return confirm(
  'Er du sikker på, du vil ændre oplysningerne om dette firma?');");
  }
}

private void findFirma(string firmaid)
{
  NaceDG.DataSource = GetNace(firmaid);
  NaceDG.DataBind();
  NaceDG.Visible = false;
  NaceTB.Text = "";
  if(!NaceDG.Items.Count.Equals(0))
  {
```

```
            for( int i = 0; i < NaceDG.Items.Count; i++)
            {
               NaceDG.SelectedIndex = i;
               NaceTB.Text += NaceDG.SelectedItem.Cells[1].Text + "; ";
            }
         }
         SalgsTB.Text = GetSpecificInfo("salgsKontakt", firmaid, "Firma", "FirmaID");
         FirmaTB.Text = GetSpecificInfo("FirmaNavn1", firmaid, "Firma", "FirmaID");
         AdresseTB.Text = GetSpecificInfo("Adresse1", firmaid, "Firma", "FirmaID");
         FaxTB.Text = GetSpecificInfo("Telefax", firmaid, "Firma", "FirmaID");
         TelefonTB.Text = GetSpecificInfo("Telefonnr", firmaid, "Firma", "FirmaID");
         try
         {
            KontaktPersonTextBox.Text =
            GetSpecificInfo("AftaleHolder", firmaid, "FirmaAftale", "FirmaID");
            StorKundeTextBox.Text =
            GetSpecificInfo("StorKundeAftale", firmaid, "FirmaAftale", "FirmaID");
            RammeTextBox.Text =
GetSpecificInfo("RammeAftale", firmaid, "FirmaAftale", "FirmaID");
            NoteTextBox.Text = GetSpecificInfo("Note", firmaid, "FirmaAftale", "FirmaID");
         }
         catch
         {
            StorKundeTextBox.Text = "nej";
            RammeTextBox.Text = "nej";
            KontaktPersonTextBox.Text = "";
            NoteTextBox.Text = "";
         }
         try
         {
            KoncernTB.Text = GetSpecificInfo("Koncern", firmaid, "FirmaKoncern", "FirmaID");
            AfdelingTB.Text = GetSpecificInfo("Afdeling", firmaid, "FirmaKoncern", "FirmaID");
         }
         catch
         {
            KoncernTB.Text = "";
            AfdelingTB.Text = "";
         }
         PostnrTB.Text = GetSpecificInfo("Postnr", firmaid, "Firma", "FirmaID");
         ByTextBox.Text = GetSpecificInfo("ByNavn", PostnrTB.Text, "Postnr", "Postnr");
         FirmaIdTB.Text = firmaid;
         PostboksTB.Text = GetSpecificInfo("Postboks", firmaid, "Firma", "FirmaID");
         ProjektDD.DataSource = GetProjekt(firmaid);
         ProjektDD.DataBind();
      }

       private void SoegFirmaBtn_Click(object sender, System.EventArgs e)
      {
         TextBox[] firmaTextBox = new TextBox[]{FirmaTB,TelefonTB, AdresseTB,PostnrTB, FaxTB,
                              ByTextBox, FirmaIdTB, NaceTB, RammeTextBox, StorKundeTextBox,
                              SalgsTB, NoteTextBox, PostboksTB, KoncernTB, AfdelingTB};
         for(int i = 0; i < firmaTextBox.Length; i++)
         {
            firmaTextBox[i].BackColor = Color.White;
         }
         Label1.Text = "";
```

```
  FirmaDG.SelectedIndex = -1;
  Label2.Text = "";
  string firma = FirmaTB.Text;
  string telefon = TelefonTB.Text;
  string adresse = AdresseTB.Text;
  string postnr = PostnrTB.Text;
  string fax = FaxTB.Text;
  string by = ByTextBox.Text;
  string nace = NaceTB.Text;
  string firmaid = FirmaIdTB.Text;
  string storkunde = StorKundeTextBox.Text;
  string ramme = RammeTextBox.Text;
  string note = NoteTextBox.Text;
  string postboks = PostboksTB.Text;
  string salg = SalgsTB.Text;
  string koncern = KoncernTB.Text;
  string afdeling = AfdelingTB.Text;
  if(firma+telefon+adresse+nace+firmaid+postnr+fax+by+ramme+storkunde+note+postboks+salg+
  afdeling+koncern != "")
  {
     FirmaDG.DataSource = GetInfo(firma, firmaid, afdeling, koncern, telefon, adresse,
     postnr, by, postboks, fax, nace, ramme, storkunde, note, salg, "Firmanavn1");
     FirmaDG.DataBind();
     FirmaDG.Visible = true;
     Label2.Text = "";
     if(FirmaDG.Items.Count.Equals(1))
     {
        FirmaDG.SelectedIndex = 0;
        firmaid = FirmaDG.SelectedItem.Cells[0].Text;
        findFirma(firmaid);
        FirmaDG.Visible = false;
     }
     if(FirmaDG.Items.Count.Equals(0))
     {
        Label2.Text = "Søgningen resulterede ikke i nogle resultater.
          Hvis du er usikker på stavemåden, brug da % som wildcard.";
        FirmaDG.Visible = false;
     }
  }
  else
  {
     Label1.Text = "Skriv venligst en værdi i mindst ét søgefelt";
     Label1.ForeColor = Color.Red;
     for(int i = 0; i < firmaTextBox.Length; i++)
     {
        firmaTextBox[i].BackColor = Color.Khaki;
     }
     FirmaDG.Visible = false;
  }
}

public void FirmaDG_sort(object sender, DataGridSortCommandEventArgs e)
{
  string firma = FirmaTB.Text;
  string telefon = TelefonTB.Text;
  string adresse = AdresseTB.Text;
```

```
  string postnr = PostnrTB.Text;
  string fax = FaxTB.Text;
  string by = ByTextBox.Text;
  string nace = NaceTB.Text;
  string firmaid = FirmaIdTB.Text;
  string ramme = RammeTextBox.Text;
  string storkunde = StorKundeTextBox.Text;
  string note = NoteTextBox.Text;
  string postboks = PostboksTB.Text;
  string salg = SalgsTB.Text;
  string koncern = KoncernTB.Text;
  string afdeling = AfdelingTB.Text;
  FirmaDG.DataSource = GetInfo(firma, firmaid, afdeling, koncern, telefon, adresse,
  postnr, by, postboks, fax, nace, ramme, storkunde, note, salg, e.SortExpression);
  FirmaDG.DataBind();
  FirmaDG.Visible = true;
}

private string findaa(string tekst)
{
  tekst = tekst.ToLower();
  if(tekst.IndexOf("aa") != -1)
  {
     tekst = tekst.Replace("aa", "å");
  }
  else
  {
     if(tekst.IndexOf("å") != -1)
     {
        tekst = tekst.Replace("å", "aa");
     }
  }
  return tekst;
}

private void home_Click(object sender, System.EventArgs e)
{
  Response.Redirect("Forside.aspx");
}

private void FirmaBtn_Click(object sender, System.EventArgs e)
{
  Response.Redirect("Firma.aspx");
}

private void PersonBtn_Click(object sender, System.EventArgs e)
{
  Response.Redirect("Person.aspx");
}

private void ProjektBtn_Click(object sender, System.EventArgs e)
{
  Response.Redirect("Projekt.aspx");
}

private void ForecastBtn_Click(object sender, System.EventArgs e)
```

```
{
  Response.Redirect("Forecast.aspx");
}

private void nulstilFirmaBtn_Click(object sender, System.EventArgs e)
{
  TextBox[] webControls = new TextBox[]{FirmaTB, NaceTB, SalgsTB, KoncernTB, OmsTITB,
                      OmsInfTB,AfdelingTB,KontaktPersonTextBox,AdresseTB,
                      FaxTB,NoteTextBox,ByTextBox,CvrTextBox,TelefonTB,
                      StorKundeTextBox,RammeTextBox, PostnrTB, FirmaIdTB, PostboksTB};
  for(int i = 0; i < webControls.Length; i++)
  {
     webControls[i].Text = null;
     webControls[i].BackColor = Color.White;
  }
  FirmaDG.Visible = false;
  ProjektDD.Items.Clear();
  Session["Firma"] = null;
  Label1.Text = "";
  Label2.Text = "";
}

private void OpretFirmaBtn_Click(object sender, System.EventArgs e)
{
  string firma = FirmaTB.Text;
  string telefonnr = TelefonTB.Text;
  string adresse = AdresseTB.Text;
  string postnr = PostnrTB.Text;
  string telefax = FaxTB.Text;
  string ejer = SalgsTB.Text;
  string storkunde = StorKundeTextBox.Text;
  string rammeaftale = RammeTextBox.Text;
  string note = NoteTextBox.Text;
  string postboks = PostboksTB.Text;
  string aftaleholder = KontaktPersonTextBox.Text;
  string afdeling = AfdelingTB.Text;
  string koncern = KoncernTB.Text;
  if(storkunde == "")
  {storkunde = "nej";}
  if(rammeaftale == "")
  {rammeaftale = "nej";}
  FirmaDG.DataSource = GetInfo("", "", "", "", telefonnr, adresse, postnr, "", "",
   "", "", "", "", "", "", "FirmaNavn1");
  FirmaDG.DataBind();
  if(FirmaDG.Items.Count.Equals(0))
  {
     insertFirma(firma, telefonnr, adresse, postnr, postboks, telefax, ejer);
     FirmaDG.DataSource = GetInfo(firma, "", "", "", telefonnr, adresse, postnr, "",
   postboks, telefax, "", "", "", "", "", "Firmanavn1");
     FirmaDG.DataBind();
     FirmaDG.Visible = true;
     FirmaDG.SelectedIndex = 0;
     string firmaid = FirmaDG.SelectedItem.Cells[0].Text;
     insertFirmaAftale(firmaid, storkunde, rammeaftale, aftaleholder, note);
     if(afdeling+koncern != "")
     {
```

```
            insertFirmaKoncern(firmaid, koncern, afdeling);
        }
        Label1.Text = firma + " er oprettet i databasen";
        Label1.ForeColor = Color.Black;
    }
    else
    {
        Label1.Text = firma + " eksisterer allerede i databasen.";
        Label1.ForeColor = Color.Red;
        FirmaDG.Visible = true;
    }
}

private void RedigerFirmaBtn_Click(object sender, System.EventArgs e)
{
    string firma = FirmaTB.Text;
    string firmaid = FirmaIdTB.Text;
    string telefonnr = TelefonTB.Text;
    string adresse = AdresseTB.Text;
    string postnr = PostnrTB.Text;
    string telefax = FaxTB.Text;
    string ejer = SalgsTB.Text;
    string storkunde = StorKundeTextBox.Text;
    string rammeaftale = RammeTextBox.Text;
    string note = NoteTextBox.Text;
    string postboks = PostboksTB.Text;
    string aftaleholder = KontaktPersonTextBox.Text;
    string koncern = KoncernTB.Text;
    string afdeling = AfdelingTB.Text;
    if(ejer == "" || firmaid == "" || firma == "" || adresse=="" || postnr == "" ||
    telefonnr == "")
    {
        //Do nothing
    }
    else
    {
        if(storkunde == "")
        {storkunde = "nej";}
        if(rammeaftale == "")
        {rammeaftale = "nej";}
        updateFirma(firmaid, firma, telefonnr, adresse, postnr, postboks, telefax, ejer);
        try
        {
            string firmaaftale =
              GetSpecificInfo("FirmaAftaleID", firmaid, "FirmaAftale", "FirmaID");
            updateFirmaAftale(firmaid, storkunde, rammeaftale, aftaleholder, note);
        }
        catch
        {
            insertFirmaAftale(firmaid, storkunde, rammeaftale, aftaleholder, note);
        }
        try
        {
            string firmakoncern =
              GetSpecificInfo("FirmaKoncernID", firmaid, "FirmaKoncern", "FirmaID");
            updateFirmaKoncern(firmaid, koncern, afdeling);
```

```
      }
      catch
      {
         if(afdeling+koncern != "")
         {
           insertFirmaKoncern(firmaid, koncern, afdeling);
         }
      }
      Label1.Text = firma+"s oplysninger er rettet.";
      Label1.ForeColor = Color.Black;
   }
}

private void FirmaDG_SelectedIndexChanged(object sender, System.EventArgs e)
{
   string firmaid = FirmaDG.SelectedItem.Cells[0].Text;
   findFirma(firmaid);
}

private void okBtn_Click(object sender, System.EventArgs e)
{
   if(ProjektDD.SelectedItem != null)
   {
      Session["Projekt"] = ProjektDD.SelectedItem.Value;
      Session["ProjektFirma"] = FirmaIdTB.Text;
      Response.Redirect("Projekt.aspx");
   }
}

public void SetDefaultButton(Page page, TextBox textControl, Button defaultButton)
{
   // Sets default buttons.
   // Originally created by Janus Kamp Hansen - http://www.kamp-hansen.dk
   // Extended by Darrell Norton - http://dotnetjunkies.com/weblog/darrell.norton/
   //   -- added Mozilla support, fixed a few issues, improved performance

   string theScript = @"

   <SCRIPT language=""javascript"">

   <!--

   function fnTrapKD(btn, event)
   {
      if (document.all)
      {
         if (event.keyCode == 13)
         {
           event.returnValue=false;
           event.cancel = true;
           btn.click();
         }
      }
      else if (document.getElementById)
      {
         if (event.which == 13)
```

```
            {
              event.returnValue=false;
              event.cancel = true;
              btn.click();
            }
         }
         else if(document.layers)
         {
            if(event.which == 13)
            {
              event.returnValue=false;
              event.cancel = true;
              btn.click();
            }
         }
      }
      // -->
      </SCRIPT>";

      Page.RegisterStartupScript("ForceDefaultToScript", theScript);
      textControl.Attributes.Add("onkeydown", "fnTrapKD(
" + defaultButton.ClientID + ",event)");
    }

      #region Web Form Designer generated code
      override protected void OnInit(EventArgs e)
      {
         //
         // CODEGEN: This call is required by the ASP.NET Web Form Designer.
         //
         InitializeComponent();
         base.OnInit(e);
      }

      /// <summary>
      /// Required method for Designer support - do not modify
      /// the contents of this method with the code editor.
      /// </summary>
      private void InitializeComponent()
      {
         this.home.Click += new System.EventHandler(this.home_Click);
         this.FirmaBtn.Click += new System.EventHandler(this.FirmaBtn_Click);
         this.PersonBtn.Click += new System.EventHandler(this.PersonBtn_Click);
         this.ProjektBtn.Click += new System.EventHandler(this.ProjektBtn_Click);
         this.ForecastBtn.Click += new System.EventHandler(this.ForecastBtn_Click);
         this.okBtn.Click += new System.EventHandler(this.okBtn_Click);
         this.SoegFirmaBtn.Click += new System.EventHandler(this.SoegFirmaBtn_Click);
         this.OpretFirmaBtn.Click += new System.EventHandler(this.OpretFirmaBtn_Click);
         this.RedigerFirmaBtn.Click += new System.EventHandler(this.RedigerFirmaBtn_Click);
         this.nulstilFirmaBtn.Click += new System.EventHandler(this.nulstilFirmaBtn_Click);
         this.FirmaDG.SortCommand +=
      new System.Web.UI.WebControls.DataGridSortCommandEventHandler(this.FirmaDG_sort);
         this.FirmaDG.SelectedIndexChanged +=
      new System.EventHandler(this.FirmaDG_SelectedIndexChanged);
         this.Load += new System.EventHandler(this.Page_Load);
```

```
        }
        #endregion
    }
}
```

I.7 Person.aspx

```
<%@ Page language="c#" Codebehind="Person.aspx.cs" AutoEventWireup="false"
Inherits="Prototype.PersonOpret" %>
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN" >
<HTML>
   <HEAD>
      <title>Person</title>
      <meta content="False" name="vs_showGrid">
      <META http-equiv="Content-Type" content="text/html; charset=windows-1252">
      <meta content="Microsoft Visual Studio .NET 7.1" name="GENERATOR">
      <meta content="C#" name="CODE_LANGUAGE">
      <meta content="JavaScript" name="vs_defaultClientScript">
      <meta content="http://schemas.microsoft.com/intellisense/ie5" name="vs_targetSchema">
   </HEAD>
   <body bgColor="#83b198">
      <form id="Form1" runat="server">
        <p>
           <!-- navigationsknapper -->
        <asp:button id="home" runat="server" CausesValidation="False" Text="Forside" />
        <asp:button id="FirmaBtn" runat="server" CausesValidation="False" Text="Firma" />
        <asp:button id="PersonBtn" runat="server" CausesValidation="False" Text="Person" />
        <asp:button id="ProjektBtn" runat="server" CausesValidation="False" Text="Projekt" />
        <asp:button id="ForecastBtn" runat="server" CausesValidation="False" Text="Forecast" />
        </p>
        <!-- Persontabel--><asp:validationsummary id="ValidationSummary1" runat="server">
        </asp:validationsummary><asp:label id="Label1" runat="server"></asp:label>
        <fieldset class="width780px"><legend class="minifont">'
<b>Søg, opret eller rediger en kontaktperson:</b>
           </legend>
           <table class="minifont">
              <tr>
                <TD>Navn:</TD>
                <TD><asp:textbox id="NavnTextBox" runat="server"></asp:textbox></TD>
                <TD><asp:requiredfieldvalidator id="NavnValidator" runat="server"
        ControlToValidate="NavnTextBox" Display="Dynamic" ErrorMessage="Indtast et navn.">
        navn*</asp:requiredfieldvalidator></TD>
                <td>Titel:</td>
                <td><asp:textbox id="TitelTextBox" runat="server"></asp:textbox></td>
                <td></td>
                <td>Firma/FirmaID:</td>
                <td><asp:textbox id="FirmaTextBox" runat="server"></asp:textbox></td>
                <td>
                   <P><asp:requiredfieldvalidator id="FirmaValidator" runat="server"
        ControlToValidate="FirmaTextBox" Display="Dynamic"
ErrorMessage="Indtast et firmanavn.">
        firma*</asp:requiredfieldvalidator></P>
                </td>
              </tr>
              <TR>
```

```
                <TD>Adresse:</TD>
                <TD><asp:textbox id="adresseTextBox" runat="server"></asp:textbox></TD>
                <td>By:</td>
                <td><asp:textbox id="ByTextBox" runat="server"></asp:textbox></td>
                <td>Postnr.:</td>
                <td><asp:textbox id="PostTextBox" runat="server"></asp:textbox></td>
              </TR>
              <TR>
                <TD>Mobil:</TD>
                <TD><asp:textbox id="MobilTextBox" runat="server"></asp:textbox></TD>
                <td>Telefon direkte:</td>
                <td><asp:textbox id="TlfArbejdeTextBox" runat="server"></asp:textbox></td>
                <td><asp:requiredfieldvalidator id="emailValidator" runat="server"
ControlToValidate="TlfArbejdeTextBox"
ErrorMessage="Indtast et telefonnr.">Telefon*</asp:requiredfieldvalidator></td>
                <td>Fax:</td>
                <td><asp:textbox id="FaxTextBox" runat="server"></asp:textbox></td>
              </TR>
              <TR>
                <TD>Afdeling:</TD>
                <TD><asp:textbox id="AfdelingTextBox" runat="server"></asp:textbox></TD>
                <td>Person ID:</td>
                <td><asp:textbox id="PersonIDTextBox" runat="server"></asp:textbox></td>
                <td>E-mail:</td>
                <td><asp:textbox id="EmailTextBox" runat="server"></asp:textbox></td>
              </TR>
              <TR>
                <TD>Eventkoder:</TD>
                <TD><asp:dropdownlist id="eventDropDown" runat="server" Visible="False"
        DataTextField="KortNavn" DataValueField="KontaktKodeID"></asp:dropdownlist>
        <asp:textbox id="personEventTB" runat="server"></asp:textbox></TD>
                <td><asp:button id="nyBtn" runat="server" CausesValidation="False"
        Text="Ny"></asp:button></td>
                <td>Projekter:</td>
                <td><asp:dropdownlist id="ProjektDropDownList" runat="server"
        DataTextField="Overskrift" DataValueField="FirmaLogbogID"></asp:dropdownlist></td>
                <td><asp:button id="okBtn" runat="server" CausesValidation="False"
        Text="OK"></asp:button></td>
                <td></td>
                <td><asp:hyperlink id="emailHyperLink" runat="server" NavigateUrl=
        "<a href=mailto:{0}>"></asp:hyperlink></td>
              </TR>
              <TR>
                <TD>Note</TD>
                <TD><asp:textbox id="NoteTextBox" runat="server" TextMode="MultiLine">
        </asp:textbox></TD>
              </TR>
           </table>
        </fieldset>
        <P>
           <!-- Søg og Nulstil knapper --></P>
        <P><asp:button id="SoegPersonBtn" runat="server" CausesValidation="False" Text="Søg">
        </asp:button>
           <asp:button id="OpretPersonBtn" runat="server" Text="Opret"></asp:button>
           <asp:button id="RedigerPersonBtn" runat="server" Text="Rediger"></asp:button>
           <asp:button id="nulstilPersonBtn" runat="server" CausesValidation="False"
```

```
         Text="Nulstil"></asp:button>
         <P><asp:label id="Label2" runat="server"></asp:label></P>
         <asp:datagrid id="PersonDG" runat="server" AllowSorting="True"
         AutoGenerateColumns="False">
            <SelectedItemStyle BackColor="Teal"></SelectedItemStyle>
            <Columns>
               <asp:BoundColumn Visible="False" DataField="PersonID" HeaderText="PersonID" />
               <asp:BoundColumn Visible="False" DataField="FirmaID" HeaderText="FirmaID" />
               <asp:BoundColumn DataField="Navn" SortExpression="Navn" HeaderText="Navn" />
               <asp:BoundColumn DataField="stilling" SortExpression="Stilling"
         HeaderText="Stilling"></asp:BoundColumn>
               <asp:BoundColumn DataField="Afdeling" SortExpression="Afdeling"
         HeaderText="Afdeling"></asp:BoundColumn>
               <asp:BoundColumn DataField="Telefonnr" SortExpression="Telefonnr"
         HeaderText="Telefonnr"></asp:BoundColumn>
               <asp:HyperLinkColumn DataNavigateUrlField="email"
         DataNavigateUrlFormatString="mailto:{0}" DataTextField="email" SortExpression="email"
         HeaderText="E-mail"></asp:HyperLinkColumn>
               <asp:ButtonColumn Text="V&#230;lg denne" HeaderText="V&#230;lg person"
         CommandName="Select"></asp:ButtonColumn>
            </Columns>
         </asp:datagrid>
         <P><asp:label id="event1" runat="server" Visible="False">Opret eller tilføj ny event.
         (Søg først for at se om eventen allerede er oprettet.)</asp:label></P>
         <P><asp:label id="event2" runat="server" Visible="False">Eventtype: </asp:label>
         <asp:textbox id="eventTB" runat="server" Visible="False"></asp:textbox>
            <asp:label id="event3" runat="server" Visible="False">Beskrivelse: </asp:label>
         <asp:textbox id="beskrivelseTB" runat="server" Visible="False"></asp:textbox>
            <asp:label id="event4" runat="server" Visible="False">Ansvarsområde:</asp:label>
         <asp:textbox id="ansvarTB" runat="server" Visible="False"></asp:textbox></P>
         <P><asp:label id="event5" runat="server" Visible="False">Oprettet af:</asp:label>
         <asp:textbox id="oprettetTB" runat="server" Visible="False"></asp:textbox>
            <asp:label id="event6" runat="server" Visible="False"></asp:label></P>
         <P><asp:button id="soegEventBtn" runat="server" CausesValidation="False"
         Text="Søg" Visible="False"></asp:button>
            <asp:button id="opretEventBtn" runat="server" CausesValidation="False"
         Text="Opret" Visible="False"></asp:button>
            <asp:button id="nulstilEventBtn" runat="server" CausesValidation="False"
         Text="Nulstil event" Visible="False"></asp:button></P>
         <P><asp:datagrid id="EventDG" runat="server" Visible="False" AllowSorting="True"
         AutoGenerateColumns="False">
         <Columns>
           <asp:BoundColumn Visible="False" DataField="KontaktKodeID"
         HeaderText="KontaktKodeID"></asp:BoundColumn>
                 <asp:BoundColumn DataField="KontaktKodeTypeNavn"
         SortExpression="KontaktKodeTypeNavn" HeaderText="Event type"></asp:BoundColumn>
                 <asp:BoundColumn DataField="KortNavn" SortExpression="KortNavn"
         HeaderText="Beskrivelse"></asp:BoundColumn>
                 <asp:BoundColumn DataField="AnsvarligAfdelingNavn"
         SortExpression="AnsvarligAfdelingNavn"
HeaderText="Ansvarsomr&#229;de"></asp:BoundColumn>
                 <asp:ButtonColumn Text="Tilf&#248;j" HeaderText="V&#230;lg"
         CommandName="Select"></asp:ButtonColumn>
               </Columns>
            </asp:datagrid></P>
      </form>
```

```
   </body>
</HTML>
```

I.8 Person.aspx.cs

```
using System;
using System.Collections;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Web;
using System.Web.SessionState;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Web.UI.HtmlControls;

namespace Prototype
{
   /// <summary>
   /// Summary description for PersonOpret.
   /// </summary>
   public class PersonOpret : System.Web.UI.Page
   {
      protected System.Web.UI.WebControls.Button home;
      protected System.Web.UI.WebControls.TextBox NavnTextBox;
      protected System.Web.UI.WebControls.RequiredFieldValidator NavnValidator;
      protected System.Web.UI.WebControls.TextBox FaxTextBox;
      protected System.Web.UI.WebControls.TextBox EmailTextBox;
      protected System.Web.UI.WebControls.TextBox NoteTextBox;
      protected System.Web.UI.WebControls.TextBox TitelTextBox;
      protected System.Web.UI.WebControls.TextBox FirmaTextBox;
      protected System.Web.UI.WebControls.TextBox AfdelingTextBox;
      protected System.Web.UI.WebControls.TextBox TlfArbejdeTextBox;
      protected System.Web.UI.WebControls.RequiredFieldValidator emailValidator;
      protected System.Web.UI.WebControls.DropDownList ProjektDropDownList;
      protected System.Web.UI.WebControls.ValidationSummary ValidationSummary1;
      protected System.Web.UI.WebControls.Button ForecastBtn;
      protected System.Web.UI.WebControls.Button ProjektBtn;
      protected System.Web.UI.WebControls.Button PersonBtn;
      protected System.Web.UI.WebControls.Button OpretPersonBtn;
      protected System.Web.UI.WebControls.Button nulstilPersonBtn;
      protected System.Web.UI.WebControls.Button SoegPersonBtn;
      protected System.Web.UI.WebControls.Button RedigerPersonBtn;
      protected System.Web.UI.WebControls.TextBox adresseTextBox;
      protected System.Web.UI.WebControls.TextBox PostTextBox;
      protected System.Web.UI.WebControls.TextBox ByTextBox;
      protected System.Web.UI.WebControls.TextBox MobilTextBox;
      protected System.Web.UI.WebControls.RequiredFieldValidator FirmaValidator;
      protected System.Web.UI.WebControls.DataGrid PersonDG;
      protected System.Web.UI.WebControls.Label Label1;
      protected System.Web.UI.WebControls.Label Label2;
      protected System.Web.UI.WebControls.HyperLink emailHyperLink;
      protected System.Web.UI.WebControls.Button okBtn;
      protected System.Web.UI.WebControls.TextBox PersonIDTextBox;
```

```
protected System.Web.UI.WebControls.DataGrid EventDG;
protected System.Web.UI.WebControls.DropDownList eventDropDown;
protected System.Web.UI.WebControls.Button nyBtn;
protected System.Web.UI.WebControls.Label event1;
protected System.Web.UI.WebControls.TextBox eventTB;
protected System.Web.UI.WebControls.Label event2;
protected System.Web.UI.WebControls.TextBox beskrivelseTB;
protected System.Web.UI.WebControls.Label event3;
protected System.Web.UI.WebControls.TextBox ansvarTB;
protected System.Web.UI.WebControls.Label event4;
protected System.Web.UI.WebControls.Button soegEventBtn;
protected System.Web.UI.WebControls.Button opretEventBtn;
protected System.Web.UI.WebControls.TextBox oprettetTB;
protected System.Web.UI.WebControls.Label event5;
protected System.Web.UI.WebControls.Label event6;
protected System.Web.UI.WebControls.TextBox personEventTB;
protected System.Web.UI.WebControls.Button nulstilEventBtn;
protected System.Web.UI.WebControls.Button FirmaBtn;

System.Data.SqlClient.SqlDataReader GetPersonInfo(string personid, string firma,
  string navn, string afdeling, string telefonnr, string mobil, string email,
  string stilling, string personevent, string note, string sortby)
{
  string connectionStringPerson;
  System.Data.SqlClient.SqlConnection sqlConnectionPerson =
  new System.Data.SqlClient.SqlConnection(connectionStringPerson);

  if(personid == "")
  { personid = "%";}
  string navnAlt = findaa(navn);

  string queryStringPerson = "SELECT * FROM [Person] WHERE ( [Person].Navn LIKE
  '%" + navn + "%' OR [Person].Navn LIKE '%" + navnAlt + "%')" +
  "AND [Person].firmaID IN (SELECT FirmaID FROM [Firma]
  WHERE [Firma].Firmanavn1 LIKE '%" + firma + "%')" +
  "AND [Person].PersonId LIKE '" + personid + "'";
  if(telefonnr != "")
  {
     queryStringPerson += "AND [Person].telefonnr LIKE '%" + telefonnr + "'";
  }
  if(stilling != "")
  {
     queryStringPerson += "AND [Person].stilling LIKE '%" + stilling + "%'";
  }
  if(afdeling != "")
  {
     queryStringPerson += "AND [Person].Afdeling LIKE '%" + afdeling + "%'";
  }
  if(personevent != "")
  {
     queryStringPerson += "AND [Person].PersonID IN (SELECT PersonID
  FROM [KontaktRegistreringer] WHERE [KontaktRegistreringer].KontaktKodeID
  IN (SELECT KontaktKodeID FROM [KontaktKoder] WHERE [KontaktKoder].KortNavn
  LIKE '%" + personevent + "%'))";
  }
  if(email != "")
```

```
  {
     queryStringPerson += "AND [Person].email LIKE '%" + email + "%'";
  }
  if(note != "")
  {
     queryStringPerson += "AND [Person].PersonID IN (SELECT PersonID FROM [PersonNote]
  WHERE [PersonNote].Note LIKE '%" + note + "%')";
  }
  if(mobil != "")
  {
     queryStringPerson += "AND [Person].Gentelefonnr LIKE '" + mobil + "%'";
  }
  queryStringPerson += " ORDER BY "+ sortby;
  System.Data.SqlClient.SqlCommand sqlCommandPerson =
  new System.Data.SqlClient.SqlCommand(queryStringPerson, sqlConnectionPerson);

  sqlConnectionPerson.Open();
  System.Data.SqlClient.SqlDataReader dataReaderPerson =
  sqlCommandPerson.ExecuteReader(System.Data.CommandBehavior.CloseConnection);
  return dataReaderPerson;
}

System.Data.SqlClient.SqlDataReader GetProjekt(string navn)
{
  string connectionString;
  System.Data.SqlClient.SqlConnection sqlConnection =
  new System.Data.SqlClient.SqlConnection(connectionString);

  string   queryString = "SELECT * FROM [FirmaLogBog]
  WHERE [FirmaLogBog].KundeKontakt LIKE '" + navn + "%'";
  System.Data.SqlClient.SqlCommand sqlCommand =
  new System.Data.SqlClient.SqlCommand(queryString, sqlConnection);
  sqlConnection.Open();
  System.Data.SqlClient.SqlDataReader dataReader =
         sqlCommand.ExecuteReader(System.Data.CommandBehavior.CloseConnection);
  return dataReader;
}

System.Data.SqlClient.SqlDataReader GetEvent(string personid, string kontakttype,
  string ansvar, string beskrivelse, string oprettet, string sortby)
{
  string connectionString;
  System.Data.SqlClient.SqlConnection sqlConnection =
  new System.Data.SqlClient.SqlConnection(connectionString);
  string queryString;
  if(personid != "")
  {
     queryString = "SELECT * FROM [KontaktKoder] WHERE [KontaktKoder].KontaktKodeID
  IN (SELECT KontaktKodeID FROM [KontaktRegistreringer]
  WHERE [KontaktRegistreringer].PersonID = '" + personid + "')";
  }
  else
  {
     queryString = "SELECT * FROM [KontaktKoder] WHERE [KontaktKoder].KontaktKodeTypeNavn
  LIKE '%" + kontakttype + "%'" +
        " AND [KontaktKoder].AnsvarligAfdelingNavn LIKE '%" + ansvar + "%'" +
```

```
        " AND [KontaktKoder].OprettetAf LIKE '%" + oprettet + "%'" +
        " AND [KontaktKoder].KortNavn LIKE '%" + beskrivelse + "%' ORDER BY "+ sortby;
  }
  System.Data.SqlClient.SqlCommand sqlCommand =
  new System.Data.SqlClient.SqlCommand(queryString, sqlConnection);
  sqlConnection.Open();
  System.Data.SqlClient.SqlDataReader dataReader =
  sqlCommand.ExecuteReader(System.Data.CommandBehavior.CloseConnection);
  return dataReader;
}

int insertPerson(string navn, string firma, string stilling, string telefonnr,
  string mobil, string afdeling, string email)
{
  string connectionString;
  System.Data.SqlClient.SqlConnection sqlConnection =
  new System.Data.SqlClient.SqlConnection(connectionString);
  string queryString = "INSERT INTO [Person] ([Navn], [Stilling], [Telefonnr],
  [Gentelefonnr], [Afdeling], [Email], [FirmaID], [OprettetAf]) VALUES (@Navn,
  @Stilling, @Telefonnr, @Mobil, @Afdeling, @Email, @FirmaID, @Oprettetaf)";
  System.Data.SqlClient.SqlCommand sqlCommand =
  new System.Data.SqlClient.SqlCommand(queryString, sqlConnection);
  sqlCommand.Parameters.Add("@Navn", System.Data.SqlDbType.VarChar).Value = navn;
  sqlCommand.Parameters.Add("@Stilling", System.Data.SqlDbType.VarChar).Value = stilling;
  sqlCommand.Parameters.Add("@Telefonnr", System.Data.SqlDbType.VarChar).Value=telefonnr;
  sqlCommand.Parameters.Add("@Mobil", System.Data.SqlDbType.VarChar).Value = mobil;
  sqlCommand.Parameters.Add("@Afdeling", System.Data.SqlDbType.VarChar).Value = afdeling;
  sqlCommand.Parameters.Add("@Email", System.Data.SqlDbType.VarChar).Value = email;
  sqlCommand.Parameters.Add("@FirmaID", System.Data.SqlDbType.Int).Value = firma;
  sqlCommand.Parameters.Add("@Oprettetaf", System.Data.SqlDbType.VarChar).Value =
          Session["Login"]+"";
  int rowsAffected = 0;
  sqlConnection.Open();
  try
  {
     rowsAffected = sqlCommand.ExecuteNonQuery();
  }
  finally
  {
     sqlConnection.Close();
  }
  return rowsAffected;
}

int insertPersonEvent(string personid, string kodeid, string oprettetaf)
{
  string connectionString;
  System.Data.SqlClient.SqlConnection sqlConnection =
  new System.Data.SqlClient.SqlConnection(connectionString);
  string queryString = "INSERT INTO [KontaktRegistreringer] ([PersonID], [KontaktKodeID],
   [OprettetAf]) VALUES (@PersonID, @KontaktKodeID, @Opretaf)";
  System.Data.SqlClient.SqlCommand sqlCommand =
  new System.Data.SqlClient.SqlCommand(queryString, sqlConnection);
  sqlCommand.Parameters.Add("@PersonID", System.Data.SqlDbType.Int).Value = personid;
  sqlCommand.Parameters.Add("@KontaktKodeID", System.Data.SqlDbType.Int).Value = kodeid;
  sqlCommand.Parameters.Add("@Opretaf", System.Data.SqlDbType.VarChar).Value = oprettetaf;
```

```
  int rowsAffected = 0;
  sqlConnection.Open();
  try
  {
     rowsAffected = sqlCommand.ExecuteNonQuery();
  }
  finally
  {
     sqlConnection.Close();
  }
  return rowsAffected;
}

int insertEvent(string kontakt, string ansvar, string beskrivelse, string oprettet)
{
  string connectionString;
  System.Data.SqlClient.SqlConnection sqlConnection =
  new System.Data.SqlClient.SqlConnection(connectionString);
  string queryString = "INSERT INTO [KontaktKoder] ([KontaktKodeTypeNavn],
  [AnsvarligAfdelingNavn], [KortNavn], [OprettetAf]) VALUES (@Kontaktkode,
  @Ansvarlig, @Kortnavn, @Oprettet)";
  System.Data.SqlClient.SqlCommand sqlCommand =
  new System.Data.SqlClient.SqlCommand(queryString, sqlConnection);
  sqlCommand.Parameters.Add("@Kontaktkode", System.Data.SqlDbType.VarChar).Value = kontakt;
  sqlCommand.Parameters.Add("@Ansvarlig", System.Data.SqlDbType.VarChar).Value = ansvar;
  sqlCommand.Parameters.Add("@Kortnavn", System.Data.SqlDbType.VarChar).Value=beskrivelse;
  sqlCommand.Parameters.Add("@Oprettet", System.Data.SqlDbType.VarChar).Value = oprettet;
  int rowsAffected = 0;
  sqlConnection.Open();
  try
  {
     rowsAffected = sqlCommand.ExecuteNonQuery();
  }
  finally
  {
     sqlConnection.Close();
  }
  return rowsAffected;
}

int insertNote(string personid, string note)
{
  string connectionString;
  System.Data.SqlClient.SqlConnection sqlConnection =
  new System.Data.SqlClient.SqlConnection(connectionString);
  string queryString = "INSERT INTO [PersonNote] ([PersonID], [Note])
  VALUES (@Personid, @Note)";
  System.Data.SqlClient.SqlCommand sqlCommand =
  new System.Data.SqlClient.SqlCommand(queryString, sqlConnection);
  sqlCommand.Parameters.Add("@Personid", System.Data.SqlDbType.Int).Value = personid;
  sqlCommand.Parameters.Add("@Note", System.Data.SqlDbType.VarChar).Value = note;

  int rowsAffected = 0;
  sqlConnection.Open();
  try
  {
```

```
      rowsAffected = sqlCommand.ExecuteNonQuery();
    }
    finally
    {
      sqlConnection.Close();
    }
    return rowsAffected;
}

int updatePerson(string navn, string personid, string firma, string stilling,
    string telefonnr, string mobil, string afdeling, string email)
{
    string connectionString;
    System.Data.SqlClient.SqlConnection sqlConnection =
    new System.Data.SqlClient.SqlConnection(connectionString);
    string queryString = "UPDATE [Person] SET [Navn] = @Navn, [Stilling]=@Stilling,
    [Telefonnr]=@Telefonnr, [Afdeling]=@Afdeling, [Email]=@Email, [Gentelefonnr]=@Mobil,
    [FirmaID]=@Firmaid, [Opdateret]=@Opdateret, [OpdateretAf]=@Opdateretaf" +
      " WHERE [Person].PersonID = '" + personid + "'";
    System.Data.SqlClient.SqlCommand sqlCommand =
    new System.Data.SqlClient.SqlCommand(queryString, sqlConnection);

    sqlCommand.Parameters.Add("@Navn", System.Data.SqlDbType.VarChar).Value = navn;
    sqlCommand.Parameters.Add("@Stilling", System.Data.SqlDbType.VarChar).Value = stilling;
    sqlCommand.Parameters.Add("@Telefonnr", System.Data.SqlDbType.VarChar).Value = telefonnr;
    sqlCommand.Parameters.Add("@Mobil", System.Data.SqlDbType.VarChar).Value = mobil;
    sqlCommand.Parameters.Add("@Afdeling", System.Data.SqlDbType.VarChar).Value = afdeling;
    sqlCommand.Parameters.Add("@Email", System.Data.SqlDbType.VarChar).Value = email;
    sqlCommand.Parameters.Add("@Firmaid", System.Data.SqlDbType.Int).Value = firma;
    sqlCommand.Parameters.Add("@Opdateret", System.Data.SqlDbType.DateTime).Value =
    DateTime.Now;
    sqlCommand.Parameters.Add("@Opdateretaf", System.Data.SqlDbType.VarChar).Value =
    Session["Login"]+"";
      int rowsAffected = 0;
      sqlConnection.Open();
      try
      {
        rowsAffected = sqlCommand.ExecuteNonQuery();
      }
      finally
      {
        sqlConnection.Close();
      }
      return rowsAffected;
    }

int updateNote(string personid, string note)
{
    string connectionString;
    System.Data.SqlClient.SqlConnection sqlConnection =
    new System.Data.SqlClient.SqlConnection(connectionString);

    string queryString = "UPDATE [PersonNote] SET [Note] = @Note
    WHERE [PersonNote].PersonID = '" + personid + "'";
    System.Data.SqlClient.SqlCommand sqlCommand =
    new System.Data.SqlClient.SqlCommand(queryString, sqlConnection);
```

```
        sqlCommand.Parameters.Add("@Note", System.Data.SqlDbType.VarChar).Value = note;
        int rowsAffected = 0;
        sqlConnection.Open();
        try
        {
           rowsAffected = sqlCommand.ExecuteNonQuery();
        }
        finally
        {
           sqlConnection.Close();
        }
        return rowsAffected;
      }

      public string GetSpecificInfo(string info, string id, string tabel, string kolonne)
      {
        string connectionString;
        System.Data.SqlClient.SqlConnection myConnection =
new System.Data.SqlClient.SqlConnection(connectionString);
        string   myScalarQuery = "SELECT " + info + " FROM " + tabel + "
        WHERE ["+tabel+"]." + kolonne + " = '" + id + "'";
        System.Data.SqlClient.SqlCommand myCommand =
new System.Data.SqlClient.SqlCommand(myScalarQuery, myConnection);
        myCommand.Connection.Open();
        string svar = myCommand.ExecuteScalar().ToString();
        myConnection.Close();
        return svar;
      }

      public string GetFirmaNavn(string firmaid)
      {
        string connectionString;
        System.Data.SqlClient.SqlConnection myConnection =
        new System.Data.SqlClient.SqlConnection(connectionString);
        string   myScalarQuery = "SELECT Firmanavn1 FROM [Firma]
        WHERE [Firma].FirmaID LIKE '" + firmaid + "'";
        System.Data.SqlClient.SqlCommand myCommand =
        new System.Data.SqlClient.SqlCommand(myScalarQuery, myConnection);
        myCommand.Connection.Open();

        string svar = myCommand.ExecuteScalar().ToString();
        myConnection.Close();

        return svar;
      }

      private void Page_Load(object sender, System.EventArgs e)
      {
        if(Session["Login"] == null)
        {
           Response.Redirect("Login.aspx");
        }
        else
        {
           if(!Page.IsPostBack)
           {
```

```
        if(Session["Person"] !=null)
        {
          string personid = ""+Session["Person"];
          string firmaid = ""+Session["PersonFirma"];
          findPerson(personid, firmaid);
          PersonDG.Visible = false;
        }
      }
      TextBox[] personTextBox = new TextBox[]{NavnTextBox, FaxTextBox, EmailTextBox,
              NoteTextBox, TitelTextBox, FirmaTextBox, AfdelingTextBox, TlfArbejdeTextBox,
                   personEventTB, adresseTextBox, PostTextBox, ByTextBox, MobilTextBox,
                   PersonIDTextBox};
      TextBox[] eventTextBox = new TextBox[]{ansvarTB, beskrivelseTB, eventTB, oprettetTB};
      for(int i = 0; i < personTextBox.Length; i++)
      {
         SetDefaultButton(this.Page, personTextBox[i], SoegPersonBtn);
      }
      for(int i = 0; i < eventTextBox.Length; i++)
      {
         SetDefaultButton(this.Page, eventTextBox[i], soegEventBtn);
      }
      RedigerPersonBtn.Attributes.Add("onclick", "return confirm(
  'Er du sikker på, du vil ændre oplysningerne om denne person?');");
  }
}

private void findPerson(string personid, string firmaid)
{
  NavnTextBox.Text = GetSpecificInfo("Navn", personid, "Person", "PersonID");
  FirmaTextBox.Text = GetSpecificInfo("FirmaNavn1", firmaid, "Firma", "FirmaID");
  FaxTextBox.Text = GetSpecificInfo("Telefax", firmaid, "Firma", "FirmaID");
  EmailTextBox.Text = GetSpecificInfo("Email", personid, "Person", "PersonID");
  emailHyperLink.NavigateUrl = "mailto:"+EmailTextBox.Text;
  emailHyperLink.Text = EmailTextBox.Text;
  PersonIDTextBox.Text = personid;
  TitelTextBox.Text = GetSpecificInfo("Stilling", personid, "Person", "PersonID");
  AfdelingTextBox.Text = GetSpecificInfo("Afdeling", personid, "Person", "PersonID");
  TlfArbejdeTextBox.Text = GetSpecificInfo("Telefonnr", personid, "Person", "PersonID");
  adresseTextBox.Text = GetSpecificInfo("Adresse1", firmaid, "Firma", "FirmaID");
  PostTextBox.Text = GetSpecificInfo("Postnr", firmaid, "Firma", "FirmaID");
  ByTextBox.Text = GetSpecificInfo("Bynavn", PostTextBox.Text, "Postnr", "Postnr");
  MobilTextBox.Text = GetSpecificInfo("Gentelefonnr", personid, "Person", "PersonID");
  ProjektDropDownList.DataSource = GetProjekt(NavnTextBox.Text);
  ProjektDropDownList.DataBind();
  eventDropDown.DataSource = GetEvent(personid, "", "", "", "", "");
  eventDropDown.DataBind();
  try
  {
     NoteTextBox.Text = GetSpecificInfo("Note", personid, "PersonNote", "PersonID");
  }
  catch
  {
     NoteTextBox.Text = "";
  }
}
```

```
      private void home_Click(object sender, System.EventArgs e)
      {
        Response.Redirect("Forside.aspx");
      }

      private void FirmaBtn_Click(object sender, System.EventArgs e)
      {
        Response.Redirect("Firma.aspx");
      }

      private void PersonBtn_Click(object sender, System.EventArgs e)
      {
        Response.Redirect("Person.aspx");
      }

      private void ProjektBtn_Click(object sender, System.EventArgs e)
      {
        Response.Redirect("Projekt.aspx");
      }

      private void ForecastBtn_Click(object sender, System.EventArgs e)
      {
        Response.Redirect("Forecast.aspx");
      }

      private void nulstilBtn_Click(object sender, System.EventArgs e)
      {
        TextBox[] personTextBox = new TextBox[]{NavnTextBox, FaxTextBox, EmailTextBox,
NoteTextBox, TitelTextBox, FirmaTextBox, AfdelingTextBox, TlfArbejdeTextBox,
                          personEventTB, adresseTextBox, PostTextBox, ByTextBox, MobilTextBox,
                          PersonIDTextBox};
        for(int i = 0; i < personTextBox.Length; i++)
        {
           personTextBox[i].Text = null;
           personTextBox[i].BackColor = Color.White;
        }
        WebControl[] eventControl = new WebControl[]
        {event1, event2, event3, event4, event5, event6, soegEventBtn, opretEventBtn,
        nulstilEventBtn, EventDG, eventTB, beskrivelseTB, ansvarTB, oprettetTB};
        for(int i = 0; i < eventControl.Length; i++)
        {
           eventControl[i].Visible = false;
        }
        beskrivelseTB.Text = null;
        ansvarTB.Text = null;
        eventTB.Text = null;
        oprettetTB.Text = null;
        ProjektDropDownList.Items.Clear();
        eventDropDown.BackColor = Color.White;
        eventDropDown.Items.Clear();
        PersonDG.Visible = false;
        emailHyperLink.Text = "";
        Session["Person"] = null;
        Label1.Text = "";
        Label2.Text = "";
        personEventTB.Visible = true;
```

```
      eventDropDown.Visible = false;
      EventDG.Visible = false;
}

   private string findaa(string tekst)
   {
     tekst = tekst.ToLower();

     if(tekst.IndexOf("aa") != -1)
     {
        tekst = tekst.Replace("aa", "å");
     }
     else
     {
        if(tekst.IndexOf("å") != -1)
        {
           tekst = tekst.Replace("å", "aa");
        }
     }
     return tekst;
   }

   private void SoegPersonBtn_Click(object sender, System.EventArgs e)
   {
     TextBox[] personTextBox = new TextBox[]{NavnTextBox,FirmaTextBox,TitelTextBox,
                 TlfArbejdeTextBox,AfdelingTextBox,EmailTextBox, PersonIDTextBox,
                 personEventTB, MobilTextBox, NoteTextBox};
     for(int i = 0; i< personTextBox.Length; i++)
     {
        personTextBox[i].BackColor = Color.White;
     }
     PersonDG.SelectedIndex = -1;
     Label1.Text = "";
     string navn = NavnTextBox.Text;
     string firma = FirmaTextBox.Text;
     if(firma != "")
     {
        if(Char.IsNumber(firma, 1))
        { firma = GetFirmaNavn(firma); }
     }
     string stilling = TitelTextBox.Text;
     string telefonnr = TlfArbejdeTextBox.Text;
     string afdeling = AfdelingTextBox.Text;
     string email = EmailTextBox.Text;
     string personid = PersonIDTextBox.Text;
     string note = NoteTextBox.Text;
     string mobil = MobilTextBox.Text;
     string personevent = personEventTB.Text;
     if(navn+firma+stilling+telefonnr+afdeling+email+mobil+personid+personevent+note != "")
     {
        PersonDG.DataSource = GetPersonInfo(personid, firma, navn, afdeling, telefonnr,
             mobil, email, stilling, personevent, note, "Navn");
        PersonDG.DataBind();
        PersonDG.Visible = true;
        Label2.Text = "";
        personEventTB.Visible = false;
```

```
      eventDropDown.Visible = true;
      if(PersonDG.Items.Count.Equals(1))
      {
         PersonDG.SelectedIndex = 0;
         personid = PersonDG.SelectedItem.Cells[0].Text;
         string firmaid = PersonDG.SelectedItem.Cells[1].Text;
         findPerson(personid, firmaid);
         PersonDG.Visible = false;
      }
      if(PersonDG.Items.Count.Equals(0))
      {
         Label2.Text = "Søgningen resulterede ikke i nogle resultater.
  Hvis du er usikker på stavemåden, brug da % som wildcard.";
         PersonDG.Visible = false;
         personEventTB.Visible = true;
         eventDropDown.Visible = false;
      }
   }
   else
   {
      for(int i = 0; i< personTextBox.Length; i++)
      {
         personTextBox[i].BackColor = Color.Khaki;
      }
      Label1.Text = "Skriv venligst en værdi i mindst ét søgefelt";
      Label1.ForeColor = Color.Red;
      PersonDG.Visible = false;
   }
}

private void Person_sort(object sender, DataGridSortCommandEventArgs e)
{
   Label1.Text = "";
   string navn = NavnTextBox.Text;
   string firma = FirmaTextBox.Text;
   if(firma != "")
   {
      if(Char.IsNumber(firma, 1))
      { firma = GetFirmaNavn(firma); }
   }
   string stilling = TitelTextBox.Text;
   string telefonnr = TlfArbejdeTextBox.Text;
   string afdeling = AfdelingTextBox.Text;
   string email = EmailTextBox.Text;
   string personid = PersonIDTextBox.Text;
   string note = NoteTextBox.Text;
   string personevent = "";
   string mobil = MobilTextBox.Text;
   PersonDG.DataSource = GetPersonInfo(personid, firma, navn, afdeling, telefonnr,
    mobil, email, stilling, personevent, note, e.SortExpression);
   PersonDG.DataBind();
   PersonDG.Visible = true;
}

private void OpretPersonBtn_Click(object sender, System.EventArgs e)
{
```

```
  PersonDG.SelectedIndex = -1;
  Label1.Text = "";
  string navn = NavnTextBox.Text;
  string firmaid = FirmaTextBox.Text;
  if(!Char.IsNumber(firmaid, 1))
  {
     Label1.Text = "Skriv et firmaID istedet for et firmanavn.";
     Label1.ForeColor = Color.Red;
  }
  else
  {
     string stilling = TitelTextBox.Text;
     string telefonnr = TlfArbejdeTextBox.Text;
     string afdeling = AfdelingTextBox.Text;
     string email = EmailTextBox.Text;
     string mobil = MobilTextBox.Text;
     string firma = GetFirmaNavn(firmaid);
     PersonDG.DataSource = GetPersonInfo("", firma, "", "", "", "", email, "", "",
  "", "Navn");
     PersonDG.DataBind();
     if(!PersonDG.Items.Count.Equals(0))
     {
        Label1.Text = "Denne person eksisterer allerede i databasen";
        Label1.ForeColor = Color.Red;
        PersonDG.Visible = true;
     }
     else
     {
        insertPerson(navn, firmaid, stilling, telefonnr, mobil, afdeling, email);
        PersonDG.DataSource = GetPersonInfo("", firma, navn, "", "", "", email, "",
  "", "", "Navn");
        PersonDG.DataBind();
        PersonDG.SelectedIndex = 0;
        string personid = PersonDG.SelectedItem.Cells[0].Text;
        string note = NoteTextBox.Text;
        insertNote(personid, note);
        Label1.Text = navn +" er oprettet i databasen.";
        Label1.ForeColor = Color.Black;

     }
  }
}

private void RedigerPersonBtn_Click(object sender, System.EventArgs e)
{
  Label1.Text = "";
  string personid = PersonIDTextBox.Text;
  string navn = NavnTextBox.Text;
  string firma = FirmaTextBox.Text;
  string stilling = TitelTextBox.Text;
  string telefonnr = TlfArbejdeTextBox.Text;
  string afdeling = AfdelingTextBox.Text;
  string email = EmailTextBox.Text;
  string note = NoteTextBox.Text;
  string mobil = MobilTextBox.Text;
  if(!Char.IsNumber(firma, 1))
```

```
  {
     Label1.Text = "Skriv et firmaID istedet for et firmanavn.";
     Label1.ForeColor = Color.Red;
  }
  else
  {
     if(navn == "" || personid == "" || firma == "" || telefonnr == "")
     {
        //Do nothing
     }
     else
     {
        updatePerson(navn, personid, firma, stilling, telefonnr, mobil, afdeling, email);
        try
        {
          string personnote=GetSpecificInfo("Note", personid, "PersonNote", "PersonID");
          updateNote(personid, note);
        }
        catch
        {
          if(note != "")
          {
             insertNote(personid, note);
          }
        }
        Label1.Text = navn+"s oplysninger er rettet.";
        Label1.ForeColor = Color.Black;
     }
  }
}

private void PersonDG_SelectedIndexChanged(object sender, System.EventArgs e)
{
  Label1.Text = "";
  string personid = PersonDG.SelectedItem.Cells[0].Text;
  string firmaid = PersonDG.SelectedItem.Cells[1].Text;
  findPerson(personid, firmaid);
}

private void okBtn_Click(object sender, System.EventArgs e)
{
  if(ProjektDropDownList.SelectedItem != null)
  {
     Session["Projekt"] = ProjektDropDownList.SelectedItem.Value;
     if(!PersonDG.Items.Count.Equals(0))
     {
        Session["ProjektFirma"] = PersonDG.SelectedItem.Cells[1].Text;
     }
     else
     {
        Session["ProjektFirma"] = Session["PersonFirma"];
     }
     Response.Redirect("Projekt.aspx");
  }
}
```

```
private void nyBtn_Click(object sender, System.EventArgs e)
{
  PersonDG.Visible = false;
  personEventTB.Visible = false;
  eventDropDown.Visible = true;
  WebControl[] eventControl = new WebControl[]{event1, event2, event3, event4, event5,
  event6, soegEventBtn, opretEventBtn, nulstilEventBtn, EventDG, eventTB,
  beskrivelseTB, ansvarTB, oprettetTB};
  for(int i = 0; i < eventControl.Length; i++)
  {
     eventControl[i].Visible = true;
  }
}

private void soegEventBtn_Click(object sender, System.EventArgs e)
{
  string kontakttype = eventTB.Text;
  string ansvar = ansvarTB.Text;
  string beskrivelse = beskrivelseTB.Text;
  string oprettet = oprettetTB.Text;
  if(kontakttype+ansvar+beskrivelse+oprettet == "")
  {
     event6.Text = "Udfyld mindst ét af felterne";
     event6.ForeColor = Color.Red;
     EventDG.Visible = false;
  }
  else
  {
     EventDG.DataSource = GetEvent("", kontakttype, ansvar, beskrivelse, oprettet,
  "KontaktKodeTypeNavn");
     EventDG.DataBind();
     EventDG.Visible = true;
  }
}

private void Event_sort(object sender, DataGridSortCommandEventArgs e)
{
  string beskrivelse = beskrivelseTB.Text;
  string ansvar = ansvarTB.Text;
  string kontakttype = eventTB.Text;
  string oprettet = oprettetTB.Text;
  EventDG.DataSource = GetEvent("", kontakttype, ansvar, beskrivelse, oprettet,
  e.SortExpression);
  EventDG.DataBind();
}

private void opretEventBtn_Click(object sender, System.EventArgs e)
{
  string beskrivelse = beskrivelseTB.Text;
  string ansvar = ansvarTB.Text;
  string kontakttype = eventTB.Text;
  string oprettet = oprettetTB.Text;
  if(beskrivelse == "" || ansvar == "" || kontakttype == "" || oprettet == "")
  {
     event6.Text = "Udfyld venligst alle fire felter.";
     event6.ForeColor = Color.Red;
```

```
  }
  else
  {
     event6.Text = "";
     insertEvent(kontakttype, ansvar, beskrivelse, oprettet);
     EventDG.DataSource = GetEvent("", kontakttype, ansvar, beskrivelse, oprettet,
  "KontaktKodeTypeNavn");
     EventDG.DataBind();
  }
}

private void EventDG_SelectedIndexChanged(object sender, System.EventArgs e)
{
  string kontaktkodeid = EventDG.SelectedItem.Cells[0].Text;
  string personid = PersonIDTextBox.Text;
  string oprettet = oprettetTB.Text;
  insertPersonEvent(personid, kontaktkodeid, oprettet);
  eventDropDown.DataSource = GetEvent(personid, "", "", "", "", "");
  eventDropDown.DataBind();
}

private void nulstilEventBtn_Click(object sender, System.EventArgs e)
{
  beskrivelseTB.Text = null;
  ansvarTB.Text = null;
  eventTB.Text = null;
  oprettetTB.Text = null;
  EventDG.Visible = false;
  event6.Text = "";
}
public void SetDefaultButton(Page page, TextBox textControl, Button defaultButton)
{
  // Sets default buttons.
  // Originally created by Janus Kamp Hansen - http://www.kamp-hansen.dk
  // Extended by Darrell Norton - http://dotnetjunkies.com/weblog/darrell.norton/
  //   -- added Mozilla support, fixed a few issues, improved performance

  string theScript = @"

  <SCRIPT language=""javascript"">

  <!--

  function fnTrapKD(btn, event)
  {
     if (document.all)
     {
        if (event.keyCode == 13)
        {
          event.returnValue=false;
          event.cancel = true;
          btn.click();
        }
     }
     else if (document.getElementById)
     {
```

```
            if (event.which == 13)
            {
              event.returnValue=false;
              event.cancel = true;
              btn.click();
            }
          }
          else if(document.layers)
          {
            if(event.which == 13)
            {
              event.returnValue=false;
              event.cancel = true;
              btn.click();
            }
          }
        }
        // -->
        </SCRIPT>";

        Page.RegisterStartupScript("ForceDefaultToScript", theScript);
        textControl.Attributes.Add("onkeydown", "fnTrapKD(
" + defaultButton.ClientID + ",event)");
      }

      #region Web Form Designer generated code
      override protected void OnInit(EventArgs e)
      {
        //
        // CODEGEN: This call is required by the ASP.NET Web Form Designer.
        //
        InitializeComponent();
        base.OnInit(e);
      }

      /// <summary>
      /// Required method for Designer support - do not modify
      /// the contents of this method with the code editor.
      /// </summary>
      private void InitializeComponent()
      {
        this.home.Click += new System.EventHandler(this.home_Click);
        this.FirmaBtn.Click += new System.EventHandler(this.FirmaBtn_Click);
        this.PersonBtn.Click += new System.EventHandler(this.PersonBtn_Click);
        this.ProjektBtn.Click += new System.EventHandler(this.ProjektBtn_Click);
        this.ForecastBtn.Click += new System.EventHandler(this.ForecastBtn_Click);
        this.nyBtn.Click += new System.EventHandler(this.nyBtn_Click);
        this.okBtn.Click += new System.EventHandler(this.okBtn_Click);
        this.SoegPersonBtn.Click += new System.EventHandler(this.SoegPersonBtn_Click);
        this.OpretPersonBtn.Click += new System.EventHandler(this.OpretPersonBtn_Click);
        this.RedigerPersonBtn.Click += new System.EventHandler(this.RedigerPersonBtn_Click);
        this.nulstilPersonBtn.Click += new System.EventHandler(this.nulstilBtn_Click);
        this.PersonDG.SortCommand +=
        new System.Web.UI.WebControls.DataGridSortCommandEventHandler(this.Person_sort);
        this.PersonDG.SelectedIndexChanged +=
        new System.EventHandler(this.PersonDG_SelectedIndexChanged);
```

```
        this.soegEventBtn.Click += new System.EventHandler(this.soegEventBtn_Click);
        this.opretEventBtn.Click += new System.EventHandler(this.opretEventBtn_Click);
        this.EventDG.SelectedIndexChanged +=
        new System.EventHandler(this.EventDG_SelectedIndexChanged);
        this.nulstilEventBtn.Click += new System.EventHandler(this.nulstilEventBtn_Click);
        this.Load += new System.EventHandler(this.Page_Load);
      }
      #endregion
   }
}
```

I.9 Projekt.aspx

```
<%@ Page language="c#" Codebehind="Projekt.aspx.cs" AutoEventWireup="false"
Inherits="Prototype.AktivitetOpret" %>
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN" >
<HTML>
   <HEAD>
      <title>Projekt</title>
      <META http-equiv="Content-Type" content="text/html; charset=windows-1252">
      <meta content="Microsoft Visual Studio .NET 7.1" name="GENERATOR">
      <meta content="C#" name="CODE_LANGUAGE">
      <meta content="JavaScript" name="vs_defaultClientScript">
      <meta content="http://schemas.microsoft.com/intellisense/ie5" name="vs_targetSchema">
   </HEAD>
   <body bgColor="#83b198">
      <form id="Form1" runat="server">
        <p>
           <!-- navigationsknapper -->
        <asp:button id="home" runat="server" CausesValidation="False" Text="Forside" />
        <asp:button id="FirmaBtn" runat="server" CausesValidation="False" Text="Firma" />
        <asp:button id="PersonBtn" runat="server" CausesValidation="False" Text="Person" />
        <asp:button id="ProjektBtn" runat="server" CausesValidation="False" Text="Projekt" />
        <asp:button id="ForecastBtn" runat="server" CausesValidation="False" Text="Forecast" />
        </p>
        <P><asp:label id="Label1" runat="server"></asp:label></P>
        <P><asp:validationsummary id="ValidationSummary1" runat="server">
        </asp:validationsummary></P>
        <!-- tabel med søgefelter -->
        <fieldset class="width780px"><legend class="minifont">
        <b>Søg, opret eller rediger et projekt:</b>
           </legend>
           <table class="minifont">
              <TR>
                <TD>Projektnavn:</TD>
                <TD><asp:textbox id="ProjektTextBox" runat="server"></asp:textbox></TD>
                <TD><asp:requiredfieldvalidator id="NavnValidator" runat="server"
        Display="Dynamic" ControlToValidate="ProjektTextBox" ErrorMessage="Indtast et navn
        på projektet.">Navn*</asp:requiredfieldvalidator></TD>
                <TD>Firma:</TD>
                <TD><asp:textbox id="firmaTextBox" runat="server"></asp:textbox></TD>
                <TD></TD>
                <TD>Firma ID</TD>
                <TD><asp:textbox id="FirmaIDTextBox" runat="server"></asp:textbox></TD>
                <TD><asp:requiredfieldvalidator id="IDValidator" runat="server"
```

```
Display="Dynamic" ControlToValidate="FirmaIDTextBox" ErrorMessage="Indtast firmaets
id-nummer.">ID*</asp:requiredfieldvalidator>
<asp:regularexpressionvalidator id="FrimaRegExp" runat="server" Display="Dynamic"
ControlToValidate="FirmaIDTextBox" ErrorMessage="FirmaID skal være et tal."
ValidationExpression="\d*"></asp:regularexpressionvalidator></TD>
      </TR>
      <TR>
        <TD>Salgsansvarlig<FONT size="2">(initialer)</FONT>:</TD>
        <TD><asp:textbox id="ejerTextBox" runat="server"></asp:textbox></TD>
        <TD><asp:requiredfieldvalidator id="salgsValidator" runat="server"
Display="Dynamic" ControlToValidate="ejerTextBox"
ErrorMessage="Indtast en salgsansvarlig">Salg*</asp:requiredfieldvalidator></TD>
        <TD>Beskrivelse:</TD>
        <TD><asp:textbox id="beskrivelseTextBox" runat="server" TextMode="MultiLine">
</asp:textbox></TD>
        <TD></TD>
        <TD>Oprettet:</TD>
        <TD><asp:textbox id="OprettetTB" runat="server"></asp:textbox></TD>
        <TD><asp:requiredfieldvalidator id="Requiredfieldvalidator1" runat="server"
Display="Dynamic" ControlToValidate="OprettetTB"
ErrorMessage="Indtast en dato.">Dato*</asp:requiredfieldvalidator>
<asp:regularexpressionvalidator id="DateRegVal" runat="server" Display="Dynamic"
ControlToValidate="OprettetTB" ErrorMessage="Datoen skal stå som DD-MM-YYYY"
ValidationExpression="\d{2}-\d{2}-\d{4}( \d{2}:\d{2}:\d{2})*">
</asp:regularexpressionvalidator></TD>
      </TR>
      <TR>
        <TD>Projektansvarlig hos kunden:</TD>
        <TD><asp:textbox id="kontaktTextBox" runat="server"></asp:textbox></TD>
        <TD>Projektansvarlig hos TI: <FONT size="2">(hvis anden end sælger)</FONT></TD>
        <TD><asp:textbox id="projektAnsTB" runat="server"></asp:textbox></TD>
        <TD>Projekt ID</TD>
        <TD><asp:textbox id="ProjektIDTextBox" runat="server"></asp:textbox></TD>
      </TR>
      <TR>
        <TD>Øvrige kunde-projektdeltagere:</TD>
      </TR>
      <TR>
        <TD>Indflydelse: <FONT size="2">(hvis anden end projektansvarlig)</FONT></TD>
        <TD><asp:textbox id="IndflydelseTextBox" runat="server"></asp:textbox></TD>
        <TD></TD>
        <TD>Økonomiansvarlig: <FONT size="2">(hvis anden end projektansvarlig)</FONT></TD>
        <TD><asp:textbox id="OkonomiTextBox" runat="server"></asp:textbox></TD>
      </TR>
      <TR>
        <TD>Øvrige TI-projektdeltagere:</TD>
        <TD><asp:textbox id="TIdeltagereTextBox" runat="server" TextMode="MultiLine">
</asp:textbox></TD>
        <TD></TD>
        <TD>Opfølgningsdato:</TD>
        <TD><asp:textbox id="opfoelgningTextBox" runat="server"></asp:textbox></TD>
        <TD><asp:regularexpressionvalidator id="OpRegVal" runat="server"
Display="Dynamic" ControlToValidate="opfoelgningTextBox" ErrorMessage="Datoen skal stå
 som DD-MM-YYYY" ValidationExpression="\d{2}-\d{2}-\d{4}( \d{2}:\d{2}:\d{2})*">
</asp:regularexpressionvalidator></TD>
        <TD>Aftaledato:</TD>
```

```
        <TD><asp:textbox id="aftaledatoTextBox" runat="server"></asp:textbox></TD>
        <TD><asp:regularexpressionvalidator id="AftaleRegVal" runat="server"
Display="Dynamic" ControlToValidate="aftaledatoTextBox" ErrorMessage="Datoen skal stå
 som DD-MM-YYYY" ValidationExpression="\d{2}-\d{2}-\d{4}( \d{2}:\d{2}:\d{2})*">
</asp:regularexpressionvalidator></TD>
      </TR>
      <TR>
        <TD>Forecast worst case:</TD>
        <TD><asp:textbox id="worstCaseTextBox" runat="server"></asp:textbox></TD>
        <TD>Forecast best case:</TD>
        <TD><asp:textbox id="BestCaseTextBox" runat="server"></asp:textbox></TD>
        <TD>Forecast:</TD>
        <TD><asp:textbox id="estimatTextBox" runat="server"></asp:textbox></TD>
      </TR>
      <TR>
        <TD>Sum:</TD>
        <TD><asp:textbox id="SumTB" runat="server"></asp:textbox></TD>
        <TD><asp:regularexpressionvalidator id="SumRegExp" runat="server"
Display="Dynamic" ControlToValidate="SumTB" ErrorMessage="Summen skal være et tal."
ValidationExpression="\d*"></asp:regularexpressionvalidator><asp:requiredfieldvalidator
id="SumValidator" runat="server" Display="Dynamic" ControlToValidate="SumTB"
ErrorMessage="Indtast et beløb.">Sum*</asp:requiredfieldvalidator></TD>
        <TD>Ordressh:</TD>
        <TD><asp:dropdownlist id="sandsynlighedDropDownList" runat="server">
              <asp:ListItem Value="Ingen angivet">Ingen angivet</asp:ListItem>
              <asp:ListItem Value="0">0% - sagen tabt</asp:ListItem>
              <asp:ListItem Value="10">10% - interessehenvendelse</asp:ListItem>
              <asp:ListItem Value="25">25% - tilbud samles</asp:ListItem>
              <asp:ListItem Value="33">33% - to andre tilbud fundet</asp:ListItem>
              <asp:ListItem Value="50">50% - et andet tilbud fundet</asp:ListItem>
              <asp:ListItem Value="75">75% - kontraktforhandlinger igang</asp:ListItem>
              <asp:ListItem Value="100">100% - kontrakt underskrevet</asp:ListItem>
           </asp:dropdownlist></TD>
        <TD>Note:</TD>
        <TD><asp:textbox id="NoteTextBox" runat="server" TextMode="MultiLine">
</asp:textbox></TD>
      </TR>
   </table>
</fieldset>
<P>
   <!-- Søg og Nulstil knapper --></P>
<P><asp:button id="SoegProjektBtn" runat="server" CausesValidation="False" Text="Søg">
</asp:button>
   <asp:button id="OpretProjektBtn" runat="server" Text="Opret"></asp:button>
   <asp:button id="RedigerProjektBtn" runat="server" Text="Rediger"></asp:button>
   <asp:button id="nulstilProjektBtn" runat="server" CausesValidation="False"
Text="Nulstil"></asp:button></P>
<P><asp:label id="Label2" runat="server"></asp:label></P>
<P><asp:datagrid id="ProjektDG" runat="server" AutoGenerateColumns="False"
AllowSorting="True">
      <SelectedItemStyle BackColor="Teal"></SelectedItemStyle>
      <Columns>
        <asp:BoundColumn Visible="False" DataField="FirmaLogbogID"
HeaderText="FirmaLogbogID"></asp:BoundColumn>
        <asp:BoundColumn DataField="firmaID" SortExpression="firmaID"
HeaderText="Firma ID"></asp:BoundColumn>
```

```
            <asp:BoundColumn DataField="Overskrift" SortExpression="Overskrift"
        HeaderText="Projektnavn"></asp:BoundColumn>
            <asp:BoundColumn DataField="LogTekst" SortExpression="LogTekst"
        HeaderText="Beskrivelse"></asp:BoundColumn>
            <asp:BoundColumn DataField="KundeKontakt" SortExpression="KundeKontakt"
        HeaderText="Kontaktperson"></asp:BoundColumn>
            <asp:ButtonColumn Text="V&#230;lg dette" HeaderText="V&#230;lg projekt"
        CommandName="Select"></asp:ButtonColumn>
          </Columns>
       </asp:datagrid></P>
        <asp:datagrid id="ForecastDG" runat="server"></asp:datagrid></form>
   </body>
</HTML>
```

I.10 Projekt.aspx.cs

```
using System;
using System.Collections;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Web;
using System.Web.SessionState;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Web.UI.HtmlControls;

namespace Prototype
{
   /// <summary>
   /// Summary description for AktivitetOpret.
   /// </summary>
   public class AktivitetOpret : System.Web.UI.Page
   {
      protected System.Web.UI.WebControls.TextBox firmaTextBox;
      protected System.Web.UI.WebControls.TextBox kontaktTextBox;
      protected System.Web.UI.WebControls.TextBox ejerTextBox;
      protected System.Web.UI.WebControls.TextBox beskrivelseTextBox;
      protected System.Web.UI.WebControls.TextBox opfoelgningTextBox;
      protected System.Web.UI.WebControls.TextBox SumTB;
      protected System.Web.UI.WebControls.TextBox OprettetTB;
      protected System.Web.UI.WebControls.TextBox ProjektTextBox;
      protected System.Web.UI.WebControls.TextBox projektAnsTB;
      protected System.Web.UI.WebControls.TextBox TIdeltagereTextBox;
      protected System.Web.UI.WebControls.TextBox worstCaseTextBox;
      protected System.Web.UI.WebControls.TextBox BestCaseTextBox;
      protected System.Web.UI.WebControls.TextBox aftaledatoTextBox;
      protected System.Web.UI.WebControls.TextBox IndflydelseTextBox;
      protected System.Web.UI.WebControls.TextBox OkonomiTextBox;
      protected System.Web.UI.WebControls.TextBox estimatTextBox;
      protected System.Web.UI.WebControls.DropDownList sandsynlighedDropDownList;
      protected System.Web.UI.WebControls.Button ForecastBtn;
      protected System.Web.UI.WebControls.Button ProjektBtn;
      protected System.Web.UI.WebControls.Button PersonBtn;
      protected System.Web.UI.WebControls.Button FirmaBtn;
```

```
protected System.Web.UI.WebControls.Button home;
protected System.Web.UI.WebControls.Button OpretProjektBtn;
protected System.Web.UI.WebControls.Button RedigerProjektBtn;
protected System.Web.UI.WebControls.Button SoegProjektBtn;
protected System.Web.UI.WebControls.DataGrid ProjektDG;
protected System.Web.UI.WebControls.TextBox FirmaIDTextBox;
protected System.Web.UI.WebControls.Label Label1;
protected System.Web.UI.WebControls.ValidationSummary ValidationSummary1;
protected System.Web.UI.WebControls.RequiredFieldValidator NavnValidator;
protected System.Web.UI.WebControls.RequiredFieldValidator IDValidator;
protected System.Web.UI.WebControls.RequiredFieldValidator SumValidator;
protected System.Web.UI.WebControls.RegularExpressionValidator FrimaRegExp;
protected System.Web.UI.WebControls.RegularExpressionValidator SumRegExp;
protected System.Web.UI.WebControls.TextBox ProjektIDTextBox;
protected System.Web.UI.WebControls.Label Label2;
protected System.Web.UI.WebControls.DataGrid ForecastDG;
protected System.Web.UI.WebControls.RequiredFieldValidator salgsValidator;
protected System.Web.UI.WebControls.RequiredFieldValidator Requiredfieldvalidator1;
protected System.Web.UI.WebControls.RegularExpressionValidator DateRegVal;
protected System.Web.UI.WebControls.RegularExpressionValidator OpRegVal;
protected System.Web.UI.WebControls.RegularExpressionValidator AftaleRegVal;
protected System.Web.UI.WebControls.TextBox NoteTextBox;
protected System.Web.UI.WebControls.Button nulstilProjektBtn;

System.Data.SqlClient.SqlDataReader GetProjektInfo(string firma, string firmaID,
  string projekt, string projektid, string kontakt, string ejer, string opfoelgning,
  string aftale, string beskrivelse, string ordressh, string ansvarTI, string projektTI,
  string note, string sortby)
{
  if(firmaID == "")
  { firmaID = "%"; }
  if(projektid == "")
  { projektid = "%"; }
  string connectionString;
  System.Data.SqlClient.SqlConnection sqlConnection =
  new System.Data.SqlClient.SqlConnection(connectionString);
  string queryString = "SELECT * FROM [FirmaLogBog] WHERE [FirmaLogBog].Overskrift
  LIKE '%" + projekt + "%'" +
  "AND [FirmaLogBog].firmaID IN (SELECT FirmaID FROM [Firma] WHERE [Firma].Firmanavn1
  LIKE '%" + firma + "%') "+
  "AND [FirmaLogBog].firmaID LIKE '" + firmaID + "'" +
  "AND [FirmaLogBog].FirmaLogbogID LIKE '" + projektid + "'";
  if(ordressh != "")
  {
     queryString += " AND [FirmaLogBog].FirmaLogbogID IN (SELECT FirmaLogbogID
  FROM [Forecast] WHERE [Forecast].OrdreSSH LIKE '" + ordressh + "')";
  }
  if(kontakt != "")
  {
     queryString += " AND [FirmaLogBog].KundeKontakt LIKE '%" + kontakt + "%'";
  }
  if(beskrivelse != "")
  {
     queryString += " AND [FirmaLogBog].LogTekst LIKE '%" + beskrivelse + "%'";
  }
  if(ejer != "")
```

```
  {
     queryString += " AND [FirmaLogBog].OprettetAf LIKE '%" + ejer + "%'";
  }
  if(opfoelgning != "")
  {
     queryString += " AND [FirmaLogBog].Opfoelgning LIKE '%" + opfoelgning + "%'";
  }
  if(aftale != "")
  {
     queryString += " AND [FirmaLogBog].TilbudsUdloeb LIKE '%" + aftale + "%'";
  }
  if(ansvarTI != "")
  {
     queryString += " AND [FirmaLogBog].FirmaLogbogID IN (SELECT FirmaLogbogID
  FROM [Forecast] WHERE [Forecast].AnsvarTI LIKE '%" + ansvarTI + "%')";
  }
  if(projektTI != "")
  {
     queryString += " AND [FirmaLogBog].FirmaLogbogID IN (SELECT FirmaLogbogID
  FROM [Forecast] WHERE [Forecast].ProjektDeltagerTI LIKE '%" + projektTI + "%')";
  }
  if(note != "")
  {
     queryString += " AND [FirmaLogBog].FirmaLogbogID IN (SELECT FirmaLogbogID
  FROM [Forecast] WHERE [Forecast].Note LIKE '%" + note + "%')";
  }
  queryString += " ORDER BY " + sortby;
  System.Data.SqlClient.SqlCommand sqlCommand =
  new System.Data.SqlClient.SqlCommand(queryString, sqlConnection);
  sqlConnection.Open();
  System.Data.SqlClient.SqlDataReader dataReader =
  sqlCommand.ExecuteReader(System.Data.CommandBehavior.CloseConnection);
  return dataReader;
}

System.Data.SqlClient.SqlDataReader GetForecast(string projektid)
{
  string connectionString;
  System.Data.SqlClient.SqlConnection sqlConnection =
  new System.Data.SqlClient.SqlConnection(connectionString);
  string queryString = "SELECT * FROM [Forecast]
  WHERE [Forecast].FirmaLogbogID = '" + projektid + "'";
  System.Data.SqlClient.SqlCommand sqlCommand =
  new System.Data.SqlClient.SqlCommand(queryString, sqlConnection);
  sqlConnection.Open();
  System.Data.SqlClient.SqlDataReader dataReader =
  sqlCommand.ExecuteReader(System.Data.CommandBehavior.CloseConnection);
  return dataReader;
}

int insertProjekt(string overskrift, string firmaid, string dato, string opfoelgning,
   string aftale, string sum, string kontakt, string salgsansvarlig, string beskrivelse)
{
  string connectionString;
  System.Data.SqlClient.SqlConnection sqlConnection =
  new System.Data.SqlClient.SqlConnection(connectionString);
```

```
string queryString;
if(opfoelgning == "" && aftale == "")
{
   queryString= "INSERT INTO [FirmaLogBog] ([Overskrift], [FirmaID], [LogDato],
[TilbudsBeloeb], [KundeKontakt], [OprettetAf], [LogTekst]) VALUES (@Overskrift,
@FirmaID, @LogDato, @TilbudsBeloeb, @Kundekontakt, @Oprettetaf, @Logtekst)";
}
else
{
   if(opfoelgning == "")
   {
      queryString = "INSERT INTO [FirmaLogBog] ([Overskrift], [FirmaID], [LogDato],
[TilbudsUdloeb], [TilbudsBeloeb], [KundeKontakt], [OprettetAf], [LogTekst]) VALUES
(@Overskrift, @FirmaID, @LogDato, @TilbudsUdloeb, @TilbudsBeloeb, @Kundekontakt,
@Oprettetaf, @Logtekst)";
   }
   else
   {
      if(aftale =="")
      {
        queryString = "INSERT INTO [FirmaLogBog] ([Overskrift], [FirmaID], [LogDato],
[Opfoelgning], [TilbudsBeloeb], [KundeKontakt], [OprettetAf], [LogTekst]) VALUES
(@Overskrift, @FirmaID, @LogDato, @Opfoelgning, @TilbudsBeloeb, @Kundekontakt,
@Oprettetaf, @Logtekst)";
      }
      else
      {
        queryString = "INSERT INTO [FirmaLogBog] ([Overskrift], [FirmaID], [LogDato],
 [Opfoelgning], [TilbudsUdloeb], [TilbudsBeloeb], [KundeKontakt], [OprettetAf],
[LogTekst]) VALUES (@Overskrift, @FirmaID, @LogDato, @Opfoelgning, @TilbudsUdloeb,
@TilbudsBeloeb, @Kundekontakt, @Oprettetaf, @Logtekst)";
      }
   }
}
System.Data.SqlClient.SqlCommand sqlCommand =
new System.Data.SqlClient.SqlCommand(queryString, sqlConnection);
sqlCommand.Parameters.Add("@Overskrift", System.Data.SqlDbType.VarChar).Value =
 overskrift;
sqlCommand.Parameters.Add("@Kundekontakt", System.Data.SqlDbType.VarChar).Value =
kontakt;
sqlCommand.Parameters.Add("@Oprettetaf", System.Data.SqlDbType.VarChar).Value =
salgsansvarlig;
sqlCommand.Parameters.Add("@Logtekst", System.Data.SqlDbType.VarChar).Value =
beskrivelse;
sqlCommand.Parameters.Add("@TilbudsBeloeb", System.Data.SqlDbType.Int).Value = sum;
sqlCommand.Parameters.Add("@LogDato", System.Data.SqlDbType.DateTime).Value = dato;
if(opfoelgning != "")
{
   sqlCommand.Parameters.Add("@Opfoelgning", System.Data.SqlDbType.DateTime).Value =
opfoelgning;
}
if(aftale != "")
{
   sqlCommand.Parameters.Add("@TilbudsUdloeb", System.Data.SqlDbType.DateTime).Value =
aftale;
}
```

```
    sqlCommand.Parameters.Add("@FirmaID", System.Data.SqlDbType.Int).Value = firmaid;
    int rowsAffected = 0;
    sqlConnection.Open();
    try
    {
       rowsAffected = sqlCommand.ExecuteNonQuery();
    }
    finally
    {
       sqlConnection.Close();
    }
    return rowsAffected;
  }

  int insertForecast(string projektid, string firmaid, string forecast, string bestcase,
    string worstcase, string ssh, string ansvarTI, string projektTI, string okonomi,
    string indflydelse, string note)
  {
    string connectionString;
    System.Data.SqlClient.SqlConnection sqlConnection =
    new System.Data.SqlClient.SqlConnection(connectionString);
    string queryString   = "INSERT INTO [Forecast] ([FirmaLogbogID], [FirmaID],
    [ForecastBeloeb], [BestCase], [WorstCase], [OrdreSSH], [AnsvarTI], [ProjektDeltagerTI],
    [Okonomi], [Indflydelse], [Note]) VALUES (@FirmaLogbogID, @FirmaID, @ForecastBeloeb,
    @BestCase,@WorstCase,@OrdreSSH, @AnsvarTI, @ProjektTI, @Okonomi, @Indflydelse, @Note)";
    System.Data.SqlClient.SqlCommand sqlCommand =
    new System.Data.SqlClient.SqlCommand(queryString, sqlConnection);
    sqlCommand.Parameters.Add("@FirmaLogbogID", System.Data.SqlDbType.Int).Value=projektid;
    sqlCommand.Parameters.Add("@ForecastBeloeb", System.Data.SqlDbType.Int).Value=forecast;
    sqlCommand.Parameters.Add("@BestCase", System.Data.SqlDbType.Int).Value = bestcase;
    sqlCommand.Parameters.Add("@WorstCase", System.Data.SqlDbType.Int).Value = worstcase;
    sqlCommand.Parameters.Add("@OrdreSSH", System.Data.SqlDbType.VarChar).Value = ssh;
    sqlCommand.Parameters.Add("@FirmaID", System.Data.SqlDbType.Int).Value = firmaid;
    sqlCommand.Parameters.Add("@AnsvarTI", System.Data.SqlDbType.VarChar).Value = ansvarTI;
    sqlCommand.Parameters.Add("@ProjektTI", System.Data.SqlDbType.VarChar).Value = projektTI;
    sqlCommand.Parameters.Add("@Okonomi", System.Data.SqlDbType.VarChar).Value = okonomi;
sqlCommand.Parameters.Add("@Indflydelse", System.Data.SqlDbType.VarChar).Value=indflydelse;
    sqlCommand.Parameters.Add("@Note", System.Data.SqlDbType.VarChar).Value = note;
    int rowsAffected = 0;
    sqlConnection.Open();
    try
    {
       rowsAffected = sqlCommand.ExecuteNonQuery();
    }
    finally
    {
       sqlConnection.Close();
    }
    return rowsAffected;
  }

  int updateProjekt(string projektid, string overskrift, string sum, string salgsansvarlig,
     string opfoelgning, string aftale, string beskrivelse, string kontakt)
  {
    string connectionString;
    System.Data.SqlClient.SqlConnection sqlConnection =
```

```
  new System.Data.SqlClient.SqlConnection(connectionString);
  string queryString;
  if(opfoelgning == "" && aftale == "")
  {
     queryString = "UPDATE [FirmaLogBog] SET [TilbudsBeloeb] = @Sum,
  [Overskrift] = @Overskrift, [OprettetAf] = @OprettetAf, [LogTekst]=@Logtekst,
  [KundeKontakt]=@Kundekontakt WHERE [FirmaLogBog].FirmaLogBogID = '" + projektid + "'";
  }
  else
  {
     if(opfoelgning == "")
     {
        queryString = "UPDATE [FirmaLogBog] SET [TilbudsBeloeb] = @Sum,
  [Overskrift] = @Overskrift, [OprettetAf] = @OprettetAf, [TilbudsUdloeb]=@TilbudsUdloeb,
  [LogTekst]=@Logtekst, [KundeKontakt]=@Kundekontakt " +
   "WHERE [FirmaLogBog].FirmaLogBogID = '" + projektid + "'";
     }
     else
     {
        if(aftale == "")
        {
          queryString = "UPDATE [FirmaLogBog] SET [TilbudsBeloeb] = @Sum,
  [Overskrift] = @Overskrift, [OprettetAf] = @OprettetAf, [Opfoelgning]=@Opfoelgning,
  [LogTekst]=@Logtekst, [KundeKontakt]=@Kundekontakt " +
   "WHERE [FirmaLogBog].FirmaLogBogID = '" + projektid + "'";
        }
        else
        {
          queryString = "UPDATE [FirmaLogBog] SET [TilbudsBeloeb] = @Sum,
  [Overskrift] = @Overskrift, [OprettetAf] = @OprettetAf, [Opfoelgning]=@Opfoelgning,
  [TilbudsUdloeb]=@TilbudsUdloeb, [LogTekst]=@Logtekst, [KundeKontakt]=@Kundekontakt " +
              "WHERE [FirmaLogBog].FirmaLogBogID = '" + projektid + "'";
        }
     }
  }
  System.Data.SqlClient.SqlCommand sqlCommand =
  new System.Data.SqlClient.SqlCommand(queryString, sqlConnection);
  sqlCommand.Parameters.Add("@Sum", System.Data.SqlDbType.Int).Value = sum;
sqlCommand.Parameters.Add("@Overskrift", System.Data.SqlDbType.VarChar).Value=overskrift;
 sqlCommand.Parameters.Add("@Kundekontakt", System.Data.SqlDbType.VarChar).Value=kontakt;
  sqlCommand.Parameters.Add("@OprettetAf", System.Data.SqlDbType.VarChar).Value =
  salgsansvarlig;
 sqlCommand.Parameters.Add("@Logtekst", System.Data.SqlDbType.VarChar).Value=beskrivelse;
  if(opfoelgning != "")
  {
     sqlCommand.Parameters.Add("@Opfoelgning", System.Data.SqlDbType.DateTime).Value
   = opfoelgning;
  }
  if(aftale != "")
  {
     sqlCommand.Parameters.Add("@TilbudsUdloeb", System.Data.SqlDbType.DateTime).Value
   = aftale;
  }
  int rowsAffected = 0;
  sqlConnection.Open();
  try
```

```
    {
       rowsAffected = sqlCommand.ExecuteNonQuery();
    }
    finally
    {
       sqlConnection.Close();
    }
    return rowsAffected;
  }

  int updateForecast(string projektid, string bestcase, string worstcase, string forecast,
    string ordressh, string ansvarTI, string projektTI, string okonomi, string indflydelse,
    string note)
  {
    string connectionString;
    System.Data.SqlClient.SqlConnection sqlConnection =
    new System.Data.SqlClient.SqlConnection(connectionString);
    string queryString = "UPDATE [Forecast] SET [BestCase] = @BestCase,
    [WorstCase] = @WorstCase, [ForecastBeloeb] = @ForecastBeloeb, [OrdreSSH] = @OrdreSSH,
    [AnsvarTI]=@AnsvarTI, [ProjektDeltagerTI]=@ProjektTI, [Okonomi]=@Okonomi,
    [Indflydelse]=@Indflydelse, [Note]=@Note, [OpdateretAf]=@Opdateretaf,
    [Opdateret]=@Opdateret WHERE [Forecast].FirmaLogBogID = '" + projektid + "'";
    System.Data.SqlClient.SqlCommand sqlCommand =
    new System.Data.SqlClient.SqlCommand(queryString, sqlConnection);
    sqlCommand.Parameters.Add("@BestCase", System.Data.SqlDbType.Int).Value = bestcase;
    sqlCommand.Parameters.Add("@WorstCase", System.Data.SqlDbType.Int).Value = worstcase;
    sqlCommand.Parameters.Add("@ForecastBeloeb", System.Data.SqlDbType.Int).Value = forecast;
    sqlCommand.Parameters.Add("@OrdreSSH", System.Data.SqlDbType.VarChar).Value = ordressh;
    sqlCommand.Parameters.Add("@AnsvarTI", System.Data.SqlDbType.VarChar).Value = ansvarTI;
    sqlCommand.Parameters.Add("@ProjektTI", System.Data.SqlDbType.VarChar).Value=projektTI;
    sqlCommand.Parameters.Add("@Okonomi", System.Data.SqlDbType.VarChar).Value = okonomi;
sqlCommand.Parameters.Add("@Indflydelse", System.Data.SqlDbType.VarChar).Value=indflydelse;
    sqlCommand.Parameters.Add("@Note", System.Data.SqlDbType.VarChar).Value = note;
    sqlCommand.Parameters.Add("@OpdateretAf", System.Data.SqlDbType.VarChar).Value =
    Session["Login"]+"";
    sqlCommand.Parameters.Add("@Opdateret", System.Data.SqlDbType.DateTime).Value =
    DateTime.Now;
  int rowsAffected = 0;
    sqlConnection.Open();
    try
    {
       rowsAffected = sqlCommand.ExecuteNonQuery();
    }
    finally
    {
       sqlConnection.Close();
    }
    return rowsAffected;
  }

  public string GetSpecificInfo(string info, string id, string tabel, string kolonne)
  {
    string connectionString;
    System.Data.SqlClient.SqlConnection myConnection =
    new System.Data.SqlClient.SqlConnection(connectionString);
    string   myScalarQuery = "SELECT " + info + " FROM " + tabel +
```

```
  " WHERE ["+tabel+"]." + kolonne + " = '" + id + "'";
  System.Data.SqlClient.SqlCommand myCommand =
  new System.Data.SqlClient.SqlCommand(myScalarQuery, myConnection);
  myCommand.Connection.Open();
  string svar = myCommand.ExecuteScalar().ToString();
  myConnection.Close();
  return svar;
}

private void Page_Load(object sender, System.EventArgs e)
{
  if(Session["Login"] == null)
  {
     Response.Redirect("Login.aspx");
  }
  else
  {
     if(!Page.IsPostBack)
     {
        if(Session["Projekt"] != null)
        {
          string projektid = "" + Session["Projekt"];
          string firmaid = "" + Session["ProjektFirma"];
          findProjekt(projektid, firmaid);
          ProjektDG.Visible = false;
        }
     }
     TextBox[] projektTextBox = new TextBox[]{firmaTextBox,kontaktTextBox,ejerTextBox,
                beskrivelseTextBox,ProjektTextBox,opfoelgningTextBox,SumTB,OprettetTB,
                projektAnsTB,TIdeltagereTextBox,worstCaseTextBox,BestCaseTextBox,
                aftaledatoTextBox,NoteTextBox, IndflydelseTextBox,OkonomiTextBox,
                estimatTextBox,FirmaIDTextBox, ProjektIDTextBox};
     for(int i = 0; i < projektTextBox.Length; i++)
     {
        SetDefaultButton(this.Page, projektTextBox[i], SoegProjektBtn);
     }

     // Put user code to initialize the page here
     RedigerProjektBtn.Attributes.Add("onclick", "return confirm(
  'Er du sikker på, du vil ændre oplysningerne om dette projekt?');");
  }
}

private void findProjekt(string projektid, string firmaid)
{
  FirmaIDTextBox.Text = firmaid;
  firmaTextBox.Text = GetSpecificInfo("FirmaNavn1", firmaid, "Firma", "FirmaID");
  kontaktTextBox.Text =
  GetSpecificInfo("KundeKontakt", projektid, "FirmaLogBog", "FirmaLogbogID");
  ejerTextBox.Text =
  GetSpecificInfo("OprettetAf", projektid, "FirmaLogBog", "FirmaLogbogID");
  beskrivelseTextBox.Text =
  GetSpecificInfo("LogTekst", projektid, "FirmaLogBog", "FirmaLogbogID");
  opfoelgningTextBox.Text =
  GetSpecificInfo("Opfoelgning", projektid, "FirmaLogBog", "FirmaLogbogID");
  SumTB.Text=GetSpecificInfo("TilbudsBeloeb", projektid, "FirmaLogBog", "FirmaLogbogID");
```

```
  OprettetTB.Text=GetSpecificInfo("Oprettet", projektid, "FirmaLogBog", "FirmaLogbogID");
  ProjektTextBox.Text =
  GetSpecificInfo("Overskrift", projektid, "FirmaLogBog", "FirmaLogbogID");
  ProjektIDTextBox.Text = projektid;
  aftaledatoTextBox.Text =
  GetSpecificInfo("TilbudsUdloeb", projektid, "FirmaLogBog", "FirmaLogbogID");
  ForecastDG.DataSource = GetForecast(projektid);
  ForecastDG.DataBind();
  ForecastDG.Visible = false;
  try
  {
     estimatTextBox.Text =
  GetSpecificInfo("ForecastBeloeb", projektid, "Forecast", "FirmaLogbogID");
     sandsynlighedDropDownList.SelectedValue =
  GetSpecificInfo("OrdreSSH", projektid, "Forecast", "FirmaLogbogID");
     worstCaseTextBox.Text =
  GetSpecificInfo("WorstCase", projektid, "Forecast", "FirmaLogbogID");
     BestCaseTextBox.Text =
  GetSpecificInfo("BestCase", projektid, "Forecast", "FirmaLogbogID");
     projektAnsTB.Text =
  GetSpecificInfo("AnsvarTI", projektid, "Forecast", "FirmaLogbogID");
     TIdeltagereTextBox.Text =
  GetSpecificInfo("ProjektDeltagerTI", projektid, "Forecast", "FirmaLogbogID");
     OkonomiTextBox.Text =
  GetSpecificInfo("Okonomi", projektid, "Forecast", "FirmaLogbogID");
     IndflydelseTextBox.Text =
  GetSpecificInfo("Indflydelse", projektid, "Forecast", "FirmaLogbogID");
     NoteTextBox.Text = GetSpecificInfo("Note", projektid, "Forecast", "FirmaLogbogID");
  }
  catch
  {
     estimatTextBox.Text = "";
     sandsynlighedDropDownList.SelectedValue = "Ingen angivet";
     worstCaseTextBox.Text = "";
     BestCaseTextBox.Text = "";
     projektAnsTB.Text = "";
     TIdeltagereTextBox.Text = "";
     OkonomiTextBox.Text = "";
     IndflydelseTextBox.Text = "";
     NoteTextBox.Text = "";
  }
}

private void home_Click(object sender, System.EventArgs e)
{
  Response.Redirect("Forside.aspx");
}

private void FirmaBtn_Click(object sender, System.EventArgs e)
{
  Response.Redirect("Firma.aspx");
}

private void PersonBtn_Click(object sender, System.EventArgs e)
{
  Response.Redirect("Person.aspx");
```

```
}

private void ProjektBtn_Click(object sender, System.EventArgs e)
{
  Response.Redirect("Projekt.aspx");
}

private void ForecastBtn_Click(object sender, System.EventArgs e)
{
  Response.Redirect("Forecast.aspx");
}

private void nulstilProjektBtn_Click(object sender, System.EventArgs e)
{
  TextBox[] projektTextBox = new TextBox[]{firmaTextBox,kontaktTextBox,ejerTextBox,
                beskrivelseTextBox,ProjektTextBox,opfoelgningTextBox,SumTB,OprettetTB,
                projektAnsTB,TIdeltagereTextBox,worstCaseTextBox,BestCaseTextBox,
                aftaledatoTextBox,NoteTextBox, IndflydelseTextBox,OkonomiTextBox,
                estimatTextBox,FirmaIDTextBox, ProjektIDTextBox};
  for(int i = 0; i < projektTextBox.Length; i++)
  {
     projektTextBox[i].Text = null;
     projektTextBox[i].BackColor = Color.White;
  }
  Label1.Text = "";
  sandsynlighedDropDownList.SelectedValue = "Ingen angivet";
  sandsynlighedDropDownList.BackColor = Color.White;
  ProjektDG.Visible = false;
  Session["Projekt"] = null;
}

private void SoegProjektBtn_Click(object sender, System.EventArgs e)
{
  TextBox[] projektTextBoxArray=new TextBox[]{firmaTextBox,FirmaIDTextBox,ProjektTextBox,
                ejerTextBox, kontaktTextBox ,beskrivelseTextBox,ProjektIDTextBox,
                opfoelgningTextBox, aftaledatoTextBox, NoteTextBox, projektAnsTB,
                TIdeltagereTextBox};
  for(int i = 0; i < projektTextBoxArray.Length; i++)
  {
     projektTextBoxArray[i].BackColor = Color.White;
  }
  sandsynlighedDropDownList.BackColor = Color.White;
  ProjektDG.SelectedIndex = -1;
  Label1.Text = "";
  string firma = firmaTextBox.Text;
  string firmaID = FirmaIDTextBox.Text;
  string projekt = ProjektTextBox.Text;
  string kontakt = kontaktTextBox.Text;
  string ejer = ejerTextBox.Text;
  string beskrivelse = beskrivelseTextBox.Text;
  string projektid = ProjektIDTextBox.Text;
  string ordressh = sandsynlighedDropDownList.SelectedItem.Value;
  string opfoelgning = opfoelgningTextBox.Text;
  string aftale = aftaledatoTextBox.Text;
  string ansvarTI = projektAnsTB.Text;
  string projektTI = TIdeltagereTextBox.Text;
```

```
  string note = NoteTextBox.Text;
  if(ordressh == "Ingen angivet")
  {ordressh = "";}
  if(firma+firmaID+projekt+kontakt+ejer+beskrivelse+projektid+ordressh+opfoelgning+aftale
  +note+ansvarTI+projektTI != "")
  {
     ProjektDG.DataSource = GetProjektInfo(firma, firmaID, projekt, projektid, kontakt,
   ejer, opfoelgning, aftale, beskrivelse, ordressh, ansvarTI, projektTI, note,
  "Overskrift");
     ProjektDG.DataBind();
     ProjektDG.Visible = true;
     Label2.Text = "";
     if(ProjektDG.Items.Count.Equals(1))
     {
        ProjektDG.SelectedIndex = 0;
        projektid = ProjektDG.SelectedItem.Cells[0].Text;
        string firmaid = ProjektDG.SelectedItem.Cells[1].Text;
        findProjekt(projektid, firmaid);
        ProjektDG.Visible = false;
     }
     if(ProjektDG.Items.Count.Equals(0))
     {
        Label2.Text = "Søgningen resulterede ikke i nogle resultater.
  Hvis du er usikker på stavemåden, brug da % som wildcard.";
        ProjektDG.Visible = false;
     }
  }
  else
  {
     for(int i = 0; i < projektTextBoxArray.Length; i++)
     {
        projektTextBoxArray[i].BackColor = Color.Khaki;
     }
     sandsynlighedDropDownList.BackColor = Color.Khaki;
     Label1.Text = "Skriv venligst en værdi i mindst ét søgefelt";
     Label1.ForeColor = Color.Red;
     ProjektDG.Visible = false;
  }
}

private void ProjektDG_sort(object sender, DataGridSortCommandEventArgs e)
{
  Label1.Text = "";
  string firma = firmaTextBox.Text;
  string firmaID = FirmaIDTextBox.Text;
  string projekt = ProjektTextBox.Text;
  string kontakt = kontaktTextBox.Text;
  string ejer = ejerTextBox.Text;
  string beskrivelse = beskrivelseTextBox.Text;
  string projektid = ProjektIDTextBox.Text;
  string opfoelgning = opfoelgningTextBox.Text;
  string aftale = aftaledatoTextBox.Text;
  string ansvarTI = projektAnsTB.Text;
  string projektTI = TIdeltagereTextBox.Text;
  string note = NoteTextBox.Text;
  string ordressh = sandsynlighedDropDownList.SelectedItem.Value;
```

```
  if(ordressh == "Ingen angivet")
  {ordressh = "";}
  ProjektDG.DataSource = GetProjektInfo(firma, firmaID, projekt, projektid,
  kontakt, ejer, opfoelgning, aftale, beskrivelse, ordressh, ansvarTI, projektTI,
  note, e.SortExpression);
  ProjektDG.DataBind();
  ProjektDG.Visible = true;
}

private void OpretProjektBtn_Click(object sender, System.EventArgs e)
{
  Label1.Text = "";
  string overskrift = ProjektTextBox.Text;
  string firma = firmaTextBox.Text;
  string firmaid = FirmaIDTextBox.Text;
  string sum = SumTB.Text;
  string oprettet = OprettetTB.Text;
  string worstcase = worstCaseTextBox.Text;
  if(worstcase == "")
  {
     worstcase = "0";
  }
  string bestcase = BestCaseTextBox.Text;
  if(bestcase == "")
  {
     bestcase = "0";
  }
  string forecast = estimatTextBox.Text;
  if(forecast == "")
  {
     forecast = "0";
  }
  string ssh = sandsynlighedDropDownList.SelectedItem.Value;
  string salgsansvarlig = ejerTextBox.Text;
  string opfoelgning = opfoelgningTextBox.Text;
  string aftale = aftaledatoTextBox.Text;
  string beskrivelse = beskrivelseTextBox.Text;
  string kontakt = kontaktTextBox.Text;
  string ansvarTI = projektAnsTB.Text;
  string projektTI = TIdeltagereTextBox.Text;
  string note = NoteTextBox.Text;
  string okonomi = OkonomiTextBox.Text;
  string indflydelse = IndflydelseTextBox.Text;
  ProjektDG.DataSource = GetProjektInfo(firma, firmaid, overskrift, "", "", "",
  "", "", "", "", "", "", "", "Overskrift");
  ProjektDG.DataBind();
  if(!ProjektDG.Items.Count.Equals(0))
  {
     Label1.Text = "Projektet eksisterer allerede i databasen.";
     Label1.ForeColor = Color.Red;
     ProjektDG.Visible = true;
  }
  else
  {
     insertProjekt(overskrift, firmaid, oprettet, opfoelgning, aftale, sum,
  kontakt, salgsansvarlig, beskrivelse);
```

```
      ProjektDG.DataSource = GetProjektInfo(firma, firmaid, overskrift, "", "",
  "", "", "", "", "", "", "", "", "Overskrift");
      ProjektDG.DataBind();
      ProjektDG.SelectedIndex = 0;
      string projektid = ProjektDG.SelectedItem.Cells[0].Text;
      insertForecast(projektid, firmaid, forecast, bestcase, worstcase, ssh,
  ansvarTI, projektTI, okonomi, indflydelse, note);
      Label1.Text = overskrift+" er oprettet i databasen.";
      Label1.ForeColor = Color.Black;
  }
}

private void RedigerProjektBtn_Click(object sender, System.EventArgs e)
{
  Label1.Text = "";
  Label1.ForeColor = Color.Black;
  string projektid = ProjektIDTextBox.Text;
  string firmaid = FirmaIDTextBox.Text;
  string sum = SumTB.Text;
  string overskrift = ProjektTextBox.Text;
  string worstcase = worstCaseTextBox.Text;
  if(worstcase == "")
  {
     worstcase = "0";
  }
  string bestcase = BestCaseTextBox.Text;
  if(bestcase == "")
  {
     bestcase = "0";
  }
  string forecast = estimatTextBox.Text;
  if(forecast == "")
  {
     forecast = "0";
  }
  string salgsansvarlig = ejerTextBox.Text;
  string opfoelgning = opfoelgningTextBox.Text;
  string aftale = aftaledatoTextBox.Text;
  string ordressh = sandsynlighedDropDownList.SelectedItem.Value;
  string beskrivelse = beskrivelseTextBox.Text;
  string kontakt = kontaktTextBox.Text;
  string ansvarTI = projektAnsTB.Text;
  string projektTI = TIdeltagereTextBox.Text;
  string note = NoteTextBox.Text;
  string okonomi = OkonomiTextBox.Text;
  string indflydelse = IndflydelseTextBox.Text;
  if(projektid == "" || firmaid == "" || overskrift == "" || sum =="" ||
   salgsansvarlig == "")
  {
     //Do nothing
  }
  else
  {
     updateProjekt(projektid, overskrift, sum, salgsansvarlig, opfoelgning, aftale,
  beskrivelse, kontakt);
     ForecastDG.DataSource = GetForecast(projektid);
```

```
     ForecastDG.DataBind();
     if(ForecastDG.Items.Count.Equals(0))
        {
          insertForecast(projektid, firmaid, forecast, bestcase, worstcase, ordressh,
   ansvarTI, projektTI, okonomi, indflydelse, note);
        }
        else
        {
          updateForecast(projektid, bestcase, worstcase, forecast, ordressh, ansvarTI,
  projektTI, okonomi, indflydelse, note);
        }
     Label1.Text = "Oplysningerne er rettet.";
     Label1.ForeColor = Color.Black;
  }
}

private void ProjektDG_SelectedIndexChanged(object sender, System.EventArgs e)
{
  string projektid = ProjektDG.SelectedItem.Cells[0].Text;
  string firmaid = ProjektDG.SelectedItem.Cells[1].Text;
  findProjekt(projektid, firmaid);
}

public void SetDefaultButton(Page page, TextBox textControl, Button defaultButton)
{
  // Sets default buttons.
  // Originally created by Janus Kamp Hansen - http://www.kamp-hansen.dk
  // Extended by Darrell Norton - http://dotnetjunkies.com/weblog/darrell.norton/
  //   -- added Mozilla support, fixed a few issues, improved performance

  string theScript = @"

  <SCRIPT language=""javascript"">

  <!--

  function fnTrapKD(btn, event)
  {
     if (document.all)
     {
        if (event.keyCode == 13)
        {
          event.returnValue=false;
          event.cancel = true;
          btn.click();
        }
     }
     else if (document.getElementById)
     {
        if (event.which == 13)
        {
          event.returnValue=false;
          event.cancel = true;
          btn.click();
        }
     }
```

```
            else if(document.layers)
            {
               if(event.which == 13)
               {
                 event.returnValue=false;
                 event.cancel = true;
                 btn.click();
               }
            }
          }
          // -->
          </SCRIPT>";

          Page.RegisterStartupScript("ForceDefaultToScript", theScript);
          textControl.Attributes.Add("onkeydown", "fnTrapKD(" + defaultButton.ClientID + ",event)");
        }

        #region Web Form Designer generated code
        override protected void OnInit(EventArgs e)
        {
          //
          // CODEGEN: This call is required by the ASP.NET Web Form Designer.
          //
          InitializeComponent();
          base.OnInit(e);
        }

        /// <summary>
        /// Required method for Designer support - do not modify
        /// the contents of this method with the code editor.
        /// </summary>
        private void InitializeComponent()
        {
          this.home.Click += new System.EventHandler(this.home_Click);
          this.FirmaBtn.Click += new System.EventHandler(this.FirmaBtn_Click);
          this.PersonBtn.Click += new System.EventHandler(this.PersonBtn_Click);
          this.ProjektBtn.Click += new System.EventHandler(this.ProjektBtn_Click);
          this.ForecastBtn.Click += new System.EventHandler(this.ForecastBtn_Click);
          this.SoegProjektBtn.Click += new System.EventHandler(this.SoegProjektBtn_Click);
          this.OpretProjektBtn.Click += new System.EventHandler(this.OpretProjektBtn_Click);
          this.RedigerProjektBtn.Click += new System.EventHandler(this.RedigerProjektBtn_Click);
          this.nulstilProjektBtn.Click += new System.EventHandler(this.nulstilProjektBtn_Click);
          this.ProjektDG.SortCommand +=
          new System.Web.UI.WebControls.DataGridSortCommandEventHandler(this.ProjektDG_sort);
          this.ProjektDG.SelectedIndexChanged +=
          new System.EventHandler(this.ProjektDG_SelectedIndexChanged);
          this.Load += new System.EventHandler(this.Page_Load);
        }
        #endregion
    }
}
```

I.11 Forecast.aspx

```
<%@ Page language="c#" Codebehind="Forecast.aspx.cs" AutoEventWireup="false"
```

```
Inherits="Prototype.Forecast" %>
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN" >
<HTML>
   <HEAD>
      <title>Forecast</title>
      <META http-equiv="Content-Type" content="text/html; charset=windows-1252">
      <meta content="Microsoft Visual Studio .NET 7.1" name="GENERATOR">
      <meta content="C#" name="CODE_LANGUAGE">
      <meta content="JavaScript" name="vs_defaultClientScript">
      <meta content="http://schemas.microsoft.com/intellisense/ie5" name="vs_targetSchema">
   </HEAD>
   <body bgColor="#83b198">
      <form id="Form2" runat="server">
        <p>
           <!-- navigationsknapper -->
        <asp:button id="home" runat="server" CausesValidation="False" Text="Forside" />
        <asp:button id="FirmaBtn" runat="server" CausesValidation="False" Text="Firma" />
        <asp:button id="PersonBtn" runat="server" CausesValidation="False" Text="Person" />
        <asp:button id="ProjektBtn" runat="server" CausesValidation="False" Text="Projekt" />
        <asp:button id="ForecastBtn" runat="server" CausesValidation="False" Text="Forecast" />
        </p>
        <P><STRONG>Forecast:</STRONG></P>
        <P>Salgsansvarlig <FONT size="2">(initialer)</FONT>:
<asp:TextBox id="salgTextBox" runat="server"></asp:TextBox>
           Årstal: <asp:TextBox id="aarTextBox" runat="server"></asp:TextBox></P>
        <P>
           <asp:Button id="soegBtn" runat="server" Text="Find forecast"></asp:Button>
           <asp:Button id="NulstilBtn" runat="server" Text="Nulstil"></asp:Button></P>
        <P>
           <asp:Label id="Label1" runat="server" Visible="False" ForeColor="Red">
Indtast en salgsansvarlig.</asp:Label></P>
        <P>
           <asp:Label id="forecastLabel" runat="server" Visible="False">Forecast for</asp:Label>
           <asp:Label id="salgLabel" runat="server"></asp:Label></P>
        <P>
           <asp:DataGrid id="ForecastDataGrid" runat="server" AutoGenerateColumns="False"
AllowSorting="True">
              <Columns>
                <asp:BoundColumn Visible="False" DataField="FirmaLogbogID"
HeaderText="ProjektID"></asp:BoundColumn>
                <asp:BoundColumn Visible="False" DataField="FirmaID" HeaderText="FirmaID">
</asp:BoundColumn>
                <asp:BoundColumn DataField="Overskrift" SortExpression="Overskrift"
HeaderText="Overskrift"></asp:BoundColumn>
                <asp:BoundColumn DataField="TilbudsUdloeb" SortExpression="TilbudsUdloeb"
HeaderText="Aftaledato"></asp:BoundColumn>
                <asp:BoundColumn DataField="TilbudsBeloeb" SortExpression="TilbudsBeloeb"
HeaderText="Tilbudsbel&#248;b"></asp:BoundColumn>
                <asp:BoundColumn DataField="OrdreSSH" SortExpression="OrdreSSH"
HeaderText="Ordresandsynlighed"></asp:BoundColumn>
                <asp:BoundColumn DataField="ForecastBeloeb" SortExpression="ForecastBeloeb"
HeaderText="Forecast"></asp:BoundColumn>
                <asp:ButtonColumn Text="V&#230;lg dette" HeaderText="V&#230;lg projekt"
CommandName="Select"></asp:ButtonColumn>
              </Columns>
           </asp:DataGrid></P>
```

```
        <P>
           <asp:Label id="TilbudsBeloebLabel" runat="server" Visible="False">
Sum af Tilbudsbeløb:</asp:Label>
           <asp:Label id="sumBeloebLabel" runat="server"></asp:Label></P>
        <P>
           <asp:Label id="ForecastSumLabel" runat="server" Visible="False">
Sum af Forecast:</asp:Label>
           <asp:Label id="sumForecastLabel" runat="server"></asp:Label></P>
        <asp:DataGrid id="DataGrid1" runat="server" AutoGenerateColumns="False">
           <Columns>
              <asp:BoundColumn DataField="FirmaLogbogID" HeaderText="FirmaLogbogID" />
              <asp:BoundColumn DataField="Overskrift" HeaderText="Overskrift" />
           </Columns>
        </asp:DataGrid>
      </form>
   </body>
</HTML>
```

I.12 Forecast.aspx.cs

```
using System;
using System.Collections;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Web;
using System.Web.SessionState;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Web.UI.HtmlControls;

namespace Prototype
{
   /// <summary>
   /// Summary description for Forecast.
   /// </summary>
   public class Forecast : System.Web.UI.Page
   {
      protected System.Web.UI.WebControls.Button home;
      protected System.Web.UI.WebControls.Button FirmaBtn;
      protected System.Web.UI.WebControls.Button PersonBtn;
      protected System.Web.UI.WebControls.Button ProjektBtn;
      protected System.Web.UI.WebControls.Button soegBtn;
      protected System.Web.UI.WebControls.TextBox salgTextBox;
      protected System.Web.UI.WebControls.DataGrid ForecastDataGrid;
      protected System.Web.UI.WebControls.Label salgLabel;
      protected System.Web.UI.WebControls.Label forecastLabel;
      protected System.Web.UI.WebControls.DataGrid DataGrid1;
      protected System.Web.UI.WebControls.Button NulstilBtn;
      protected System.Web.UI.WebControls.TextBox aarTextBox;
      protected System.Web.UI.WebControls.Label sumForecastLabel;
      protected System.Web.UI.WebControls.Label sumBeloebLabel;
      protected System.Web.UI.WebControls.Label TilbudsBeloebLabel;
      protected System.Web.UI.WebControls.Label ForecastSumLabel;
```

```
      protected System.Web.UI.WebControls.Label Label1;
      protected System.Web.UI.WebControls.Button ForecastBtn;

      private void Page_Load(object sender, System.EventArgs e)
      {
        if(Session["Login"] == null)
        {
           Response.Redirect("Login.aspx");
        }
        else
        {
           SetDefaultButton(this.Page, salgTextBox, soegBtn);
           SetDefaultButton(this.Page, aarTextBox, soegBtn);
        }
      }

      System.Data.SqlClient.SqlDataReader GetProjekt(string salgsansvarlig, string aar,
string sortby)
      {
        string connectionString;
        System.Data.SqlClient.SqlConnection sqlConnection =
new System.Data.SqlClient.SqlConnection(connectionString);
        string queryString = "SELECT * FROM [FirmaLogBog] LEFT JOIN
[Forecast] ON [FirmaLogBog].FirmaLogbogID = [Forecast].FirmaLogbogID" +
           " WHERE [FirmaLogBog].OprettetAf LIKE '" + salgsansvarlig + "'";
        if(aar != "")
        {
           queryString += " AND [FirmaLogBog].TilbudsUdloeb LIKE '%" + aar + "%'";
        }
        queryString += " ORDER BY "+sortby;
        System.Data.SqlClient.SqlCommand sqlCommand =
new System.Data.SqlClient.SqlCommand(queryString, sqlConnection);
        sqlConnection.Open();
        System.Data.SqlClient.SqlDataReader dataReader =
sqlCommand.ExecuteReader(System.Data.CommandBehavior.CloseConnection);
        return dataReader;
      }

      private void soegBtn_Click(object sender, System.EventArgs e)
      {
        Label1.Visible = false;
        string salgsansvarlig = salgTextBox.Text;
        string aar = aarTextBox.Text;
        if(salgsansvarlig=="")
        {
           Label1.Visible = true;
           ForecastDataGrid.Visible = false;
           forecastLabel.Visible = false;
           ForecastSumLabel.Visible = false;
           TilbudsBeloebLabel.Visible = false;
           salgLabel.Visible = false;
           sumBeloebLabel.Visible = false;
           sumForecastLabel.Visible = false;
        }
        else
        {
```

```
      ForecastDataGrid.DataSource = GetProjekt(salgsansvarlig, aar, "TilbudsUdloeb DESC");
      ForecastDataGrid.DataBind();
      ForecastDataGrid.Visible = true;
      forecastLabel.Visible = true;
      ForecastSumLabel.Visible = true;
      TilbudsBeloebLabel.Visible = true;
      int sum = 0;
      int forecast = 0;
      for(int i = 0; i < ForecastDataGrid.Items.Count; i++)
      {
         ForecastDataGrid.SelectedIndex = i;
         if(Char.IsNumber(ForecastDataGrid.SelectedItem.Cells[4].Text, 0))
         {
           sum += Int32.Parse(ForecastDataGrid.SelectedItem.Cells[4].Text);
         }
         if(Char.IsNumber(ForecastDataGrid.SelectedItem.Cells[6].Text, 0))
         {
           forecast += Int32.Parse(ForecastDataGrid.SelectedItem.Cells[6].Text);
         }
      }
      salgLabel.Text = salgsansvarlig;
      sumBeloebLabel.Text = sum.ToString();
      sumForecastLabel.Text = forecast.ToString();
  }
}

private void home_Click(object sender, System.EventArgs e)
{
  Response.Redirect("Forside.aspx");
}

private void FirmaBtn_Click(object sender, System.EventArgs e)
{
  Response.Redirect("Firma.aspx");
}

private void PersonBtn_Click(object sender, System.EventArgs e)
{
  Response.Redirect("Person.aspx");
}

private void ProjektBtn_Click(object sender, System.EventArgs e)
{
  Response.Redirect("Projekt.aspx");
}

private void ForecastBtn_Click(object sender, System.EventArgs e)
{
  Response.Redirect("Forecast.aspx");
}

private void NulstilBtn_Click(object sender, System.EventArgs e)
{
  Label1.Visible = false;
  salgLabel.Text = "";
  forecastLabel.Visible = false;
```

```
        sumForecastLabel.Text = "";
        sumBeloebLabel.Text = "";
        TilbudsBeloebLabel.Visible = false;
        ForecastSumLabel.Visible = false;
        salgTextBox.Text = "";
        aarTextBox.Text = "";
        ForecastDataGrid.Visible = false;
      }

      private void ForecastDataGrid_SortCommand(object source,
System.Web.UI.WebControls.DataGridSortCommandEventArgs e)
      {
        string salgsansvarlig = salgTextBox.Text;
        string aar = aarTextBox.Text;
        ForecastDataGrid.DataSource = GetProjekt(salgsansvarlig, aar, e.SortExpression);
        ForecastDataGrid.DataBind();
        ForecastDataGrid.Visible = true;
      }

      private void ForecastDataGrid_SelectedIndexChanged(object sender, System.EventArgs e)
      {
        Session["Projekt"] = ForecastDataGrid.SelectedItem.Cells[0].Text;
        Session["ProjektFirma"] = ForecastDataGrid.SelectedItem.Cells[1].Text;
        Response.Redirect("Projekt.aspx");
      }
      public void SetDefaultButton(Page page, TextBox textControl, Button defaultButton)
      {
        // Sets default buttons.
        // Originally created by Janus Kamp Hansen - http://www.kamp-hansen.dk
        // Extended by Darrell Norton - http://dotnetjunkies.com/weblog/darrell.norton/
        //   -- added Mozilla support, fixed a few issues, improved performance

        string theScript = @"

        <SCRIPT language=""javascript"">

        <!--

        function fnTrapKD(btn, event)
        {
           if (document.all)
           {
              if (event.keyCode == 13)
              {
                event.returnValue=false;
                event.cancel = true;
                btn.click();
              }
           }
           else if (document.getElementById)
           {
              if (event.which == 13)
              {
                event.returnValue=false;
                event.cancel = true;
                btn.click();
```

```
            }
          }
          else if(document.layers)
          {
             if(event.which == 13)
             {
               event.returnValue=false;
               event.cancel = true;
               btn.click();
             }
          }
        }
        // -->
        </SCRIPT>";

        Page.RegisterStartupScript("ForceDefaultToScript", theScript);
        textControl.Attributes.Add("onkeydown", "fnTrapKD(
" + defaultButton.ClientID + ",event)");
      }

      #region Web Form Designer generated code
      override protected void OnInit(EventArgs e)
      {
        //
        // CODEGEN: This call is required by the ASP.NET Web Form Designer.
        //
        InitializeComponent();
        base.OnInit(e);
      }

      /// <summary>
      /// Required method for Designer support - do not modify
      /// the contents of this method with the code editor.
      /// </summary>
      private void InitializeComponent()
      {
        this.home.Click += new System.EventHandler(this.home_Click);
        this.FirmaBtn.Click += new System.EventHandler(this.FirmaBtn_Click);
        this.PersonBtn.Click += new System.EventHandler(this.PersonBtn_Click);
        this.ProjektBtn.Click += new System.EventHandler(this.ProjektBtn_Click);
        this.ForecastBtn.Click += new System.EventHandler(this.ForecastBtn_Click);
        this.soegBtn.Click += new System.EventHandler(this.soegBtn_Click);
        this.NulstilBtn.Click += new System.EventHandler(this.NulstilBtn_Click);
        this.ForecastDataGrid.SortCommand +=new
System.Web.UI.WebControls.DataGridSortCommandEventHandler
(this.ForecastDataGrid_SortCommand);
        this.ForecastDataGrid.SelectedIndexChanged += new
System.EventHandler(this.ForecastDataGrid_SelectedIndexChanged);
        this.Load += new System.EventHandler(this.Page_Load);
      }
      #endregion
   }
}
```