Avdelning för datavetenskap

David Jonsson & Fredrik Larsson

Webbaserad släktträdsmodul

Med koppling till EmiWeb

Web-Based Family Tree Module

With Link to EmiWeb

Datavetenskap

C-uppsats 15hp

Datum: 08-06-03

Handledare: Donald F. Ross

Examinator: Martin Blom

Löpnummer: C2008:03

Karlstads universitet 651 88 Karlstad

Tfn 054-700 10 00 Fax 054-700 14 60

Information@kau.se www.kau.se

Denna rapport är skriven som en del av det arbete som krävs för att erhålla en kandidatexamen i datavetenskap. Allt material i denna rapport, vilket inte är mitt eget, har blivit tydligt identifierat och inget material är inkluderat som tidigare använts för erhållande av annan examen.

David Jonsson

_____

Fredrik Larsson

_____

Godkänd, 080603

Handledare: Donald F. Ross

_____

Examinator: Martin Blom

_____

Sammanfattning

Att släktforska idag handlar inte bara om att leta runt i gamla arkiv eller kyrkoböcker. Idag erbjuds en rad olika möjligheter för dem som släktforskar. Idag behöver inte en intresserad person vara någon expert på släktforskning utan även personer utan erfarenhet av släktforskning kan prova på detta. Detta görs möjligt tack vare den nya teknik som finns då främst de webbsidor som erbjuder möjligheten att söka i flera stora arkiv på ett enkelt sätt, och direkt framför sin egen dator hemma. Även för dem som fortfarande föredrar att leta på det gamla sättet finns idag program till datorer som underlättar arbetet betydligt. Dock är det inte många som har något stöd för nätverk vilket ger problem då större föreningar som har flera användare som jobbar med t.ex. släktträd samtidigt. Det skulle därför underlätta betydligt om ett sådant program fanns där användarna kunde koppla upp sig mot en gemensam databas och arbeta samtidigt med samma släktträd utan att behöva vänta på någon annan skall bli klar med sitt arbete.

Denna rapport beskriver det arbete som utförts på Emigrantregistret i Karlstad under våren 2008, där uppgiften var att skapa en prototyp för ett webbaserat system för konstruktion av släktträd. Syftet med applikationen är att effektivisera det arbete som utförs på Emigrantregistret, och även i framtiden kunna ge möjligheten att lansera materialet för allmänheten via Internet i någon form av betalningstjänst.

Web-Based Family Tree Module

Abstract

Today's genealogy research reaches beyond the old archives and church records. There are several convenient methods available to those keen on investigating their ancestry. You need not be a genealogy expert with years of experience to start tracing your roots. Modern and often web-based technology lets you search through multiple voluminous archives at once, in your own home. Those who still prefer the old-fashioned method of research will be also be aided by computer software that helps lighten their workload. Unfortunately, many of these programs were intended for a single user, and come without network support, which can be troublesome if you wish to share your tree with others, as in the case with many companies where several people are working on the same family tree. A network-based family tree would allow the users to connect to a database and work in the same application without having to wait for someone else to finish.

This report describes the work done at Emigrantregistret / Kinship Center in Karlstad during the spring of 2008, where we were assigned to create a prototype for web-based family tree construction. The application aims to make Emigrantregistret / Kinship Center's work more efficient and provide a method for introducing the material to the public via the Internet as a subscription service.

Innehållsförteckning

Figurförteckning

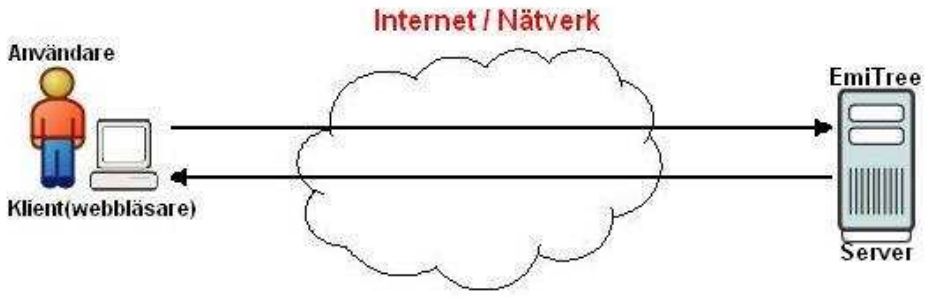
Tabellförteckning

1 Inledning

Dagens släktforskning har mer och mer börjat använda nya tekniker som dataprogram och webbsidor för att kunna erbjuda intresserade olika möjligheter att släktforska. Detta är ett sätt som effektiviserar arbetet en hel del, då det arbete med att leta i kyrkböcker eller liknande inte längre är ett måste för dem som inte har möjlighet till detta. Idag räcker det med att ha tillgång till en tjänst via Internet, och enkelt och snabbt söka upp, och snyggt och prydligt få den informationen som önskas.

Denna rapport behandlar ett projekt som just är inriktat mot en liknande tjänst inom släktforskning, nämligen en webbapplikation för att skapa släktträd. De program som idag finns (2008-05-06) är ofta inriktade på att användas på en dator och saknar ofta möjligheten att använda nätverk att fler användare kan utnyttja systemet samtidigt.

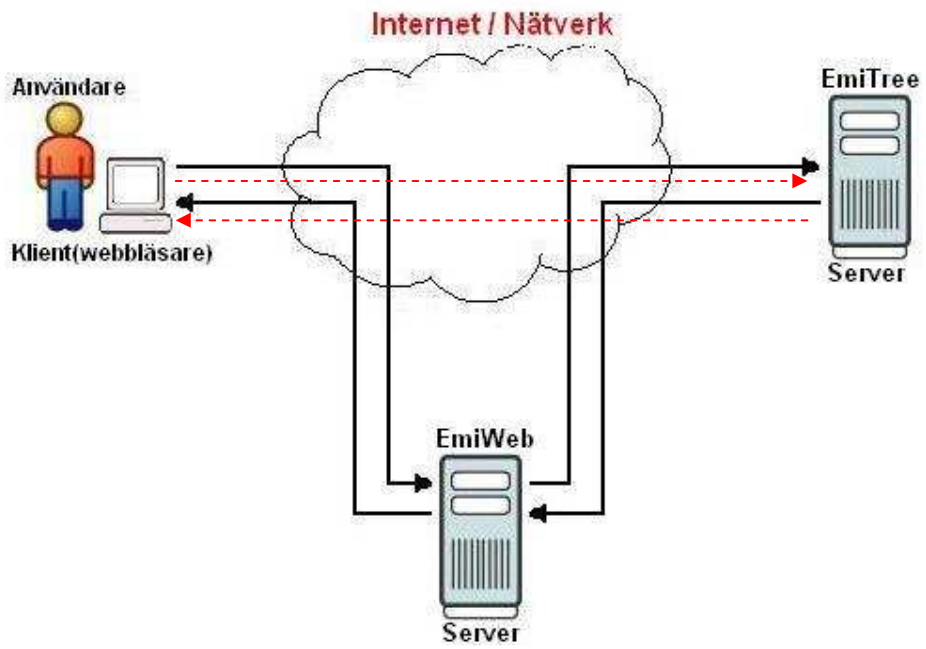
Därför har detta projekt inriktats på att skapa just ett program för att ge flera användare möjlighet att arbeta och utnyttja systemet samtidigt. För att också göra systemet geografiskt oberoende valdes en webbapplikation som lösning. En webbsida kan nås av alla de som har tillgång till en dator och Internet oberoende av vart i världen användaren befinner sig. Figur 1 visar hur en användare av systemet ansluter till applikationen.



Figur 1: Översiktsbild för användning av EmiTree

Rapporten behandlar också hur denna applikation kan utnyttjas av andra system för att tillhandahålla olika släktträd. Vid Emigrantregistret i Karlstad som är uppdragsgivare för detta projekt utvecklas en annan webbapplikation vid namn EmiWeb som är ett sökbart arkiv av emigranter. Här önskades en möjlighet att visa ett släktträd för en emigrant. Figur 2 visar ett

exempel på hur en användare som är inloggad på EmiWeb kan få se ett släktträd som hämtats ifrån EmiTree.



Figur 2: Översiktsbild för användning av EmiWeb med koppling till EmiTree

Kapitel 2 i denna rapport tar upp viktiga begrepp som är bra att läsaren vet innan fortsatt läsning. Här presenteras även de program och filstrukturer som idag används för att skapa och lagra släktträd. En kort presentation ges också av det program som utvecklas vid Emigrantregistret nämligen EmiWeb. Här behandlas också grundinformation för de tekniker som användes under projektet, och då främst ASP.NET.

Den kravspecifikation som togs fram av uppdragsgivaren presenteras närmare i kapitel 3.

I kapitlen 4-6 presenteras de olika delar som detta projekt har omfattat, databasdesign, gränssnittsdesign och programdesign. Allt detta gjordes från grunden då inget tidigare system fanns att vidareutveckla.

För att ta reda på prestandan samt hur användarvänligheten var för systemet gjordes ett antal tester, dessa resultat presenteras i kapitel 7.

I kapitel 8 är rapportens slutsats, där en utvärdering är gjord för arbetet och en kontroll för att se om alla kraven är uppfyllda. Här finns även förslag på andra funktioner som kan bli aktuella i framtida projekt.

2 Bakgrund

2.1 Introduktion

Under åren 1957-1959 diskuterades en rad olika förslag för att stärka sambanden mellan Sverige och ättlingar i Amerika. I samband med en konferens i Karlstad där samtliga av Värmlands primärkommuner och övriga intressenter deltog grundades 1960 Emigrantregistret [2]. Huvuduppgiften var till en början att söka efter och registrera de personer som emigrerat till Amerika, samt att utföra släktutredningar omkring dem och att samla in material i syfte att belysa deras situation i Amerika. Förutom alfabetiska register upprättades emigrantlängder för varje församling i kronologisk ordning. Detta arbete bygger på församlingarnas utflyttnings- och husförhörsböcker. Kungliga Bibliotekets samlingar av svenskamerikanska tidningar, och notiser om emigrationen har gåtts igenom. Nu omfattar institutionen runt 7000 volymer och fortlöpande bevakning av antikvariat och bokauktioner sker.

Detta medför att arkiven blir stora och arbetet med sökning och utförande av släktutredningar är en krävande process, därför har Emigrantregistret sedan några år påbörjat och delvis genomfört en datorisering av allt material för att underlätta detta arbete.

De program som används inom släktforskning idag är oftast föråldrade, behovet av att modernisera dessa är stort. Föreningen arbetar just nu med en webbaserad lösning som skall knyta ihop ett antal databaser med koppling till den stora emigrationen till Nordamerika. Denna tjänst ger personalen möjlighet att snabbare och effektivare söka fram personer ur arkiven och kunna utföra uppdragsforskning fortare utan att personalen behöver hoppa ut och in i olika programvaror för att följa och koppla personer.

De vill nu ha hjälp med att skapa ett webbaserat system för att konstruera släktträd samt att göra sökningar i släktträd via webben. Det nuvarande systemet som används bygger på en lokal lösning, där samtliga forskare måste föra in sina resultat via en och samma dator. Denna lösning är inte en effektiv och bra lösning, då en släktutredning kan vara mycket tidskrävande, då det idag finns stora mängder släktutredningar i arkiven på pappersform. Dessa utredningar har Emigrantregistret ambitioner att överföra till digital form. Med ett webbaserat verktyg för detta kan flera använda och jobba med släktträdet samtidigt. Det ger också föreningen möjlighet att i framtiden om efterfrågan finns, låta utomstående personer abonnera på tjänsten och då få tillgång till att göra sökningar i släktträdet via Internet.

2.2 Definitioner

Här presenteras de definitioner och termer som används i resten av rapporten.

ANSEL – ANSEL är en förkortning av **A**merican **N**ational **S**tandard for **E**xtended **L**atin och är teckensamling som används främst i bibliotekets sammanhang. GEDCOM formatet (se definition nedan) använder ANSEL som standard format [6].

ANSI – **A**merican **N**ational **S**tandards **I**nstitute [1] är en amerikansk organisation som arbetar för att standardisera produkter för den amerikanska och den internationella marknaden. ANSI grundades av ett antal ingenjörsorganisationer såväl som den amerikanska staten.

ASP - ASP är ett server-baserat scriptverktyg för att skapa dynamiska webbsidor och webbapplikationer [8].

ASP.NET – ASP.NET används för att skapa webbapplikationer med hjälp av Microsofts senaste ramverk. Under de senaste åren har ASP.NET utvecklats till ett av de mest konsekventa, stabila och innehållsrika ramverk som finns för hantering av http-anrop [8].

DATABAS – Enligt [17] är en databas är en samling data som hör ihop och kan fungera som en modell av världen för t.ex. ett företag och dess verksamhet. De data som är lagrad är också bestående vilket innebär att de data som finns inte försvinner om programmet stängs ner eller om datorn stängs av.

DISGEN – DISGEN är ett dataprogram för släktforskning som skapades av föreningen DIS. Syfte med dataprogrammet var att underlätta användning av datorer vid släktforskning [3].

EmiTree – Detta är benämningen på detta projekt och antyder på den släktträdsapplikation som skall utvecklas under projektet.

E/R DIAGRAM – När en databas ska skapas som beskriver en del av verkligheten, det första som ofta görs är att beskriva hur verkligheten ser ut och fungerar. Denna beskrivning brukar kallas för en begreppsmässig beskrivning och E/R Diagram är en modell för detta. E/R står för Entity-Relationship.

GEDCOM – GEDCOM är en förkortning av **GE**nealogical **D**ata **COM**unication och är ett flexibelt samt ett konstant format för utbyte mellan olika genealogisystem. Syftet med formatet var att utdela genealogisk information och skapa ett bra verktyg för andra mjukvaror [4].

MD5 - En hash-funktion som skapar en 128 bitar stor kontrollsumma av en angiven text [15].

MySQL – MySQL är databashanterare som använder frågespråket SQL. Fördelen med MySQL är att det är en fri programvara som är licenserad under GNU [25]. Mer information om MySQL finns på [5].

RELATIONSMODELLER – En datamodell som beskriver hur data i tabeller skall organiseras och se ut.

2.3 DISGEN

Detta kapitel kommer handla om släktforsningsprogrammet DISGEN och dess funktionalitet. Kapitlet tar också upp definitioner av ett släkträd, hur ett släktträd representerares, och hur ett släktträd kan se ut.

2.3.1 Översikt

1970 gjorde DIS de första försöken med datorhjälp vid släktforskning och 1982 kom den första versionen av DISGEN, men inte förrän 1997 kom den första versionen till operativsystemet Windows. Det är också endast på denna version som vidareutveckling sker. DISGEN är nu uppe i sin åttonde version och utvecklas hela tiden efter de önskemål och idéer som kommer in till föreningen.

Detta program är det nuvarande som Emigrantregistret använder för att skapa sina släktträd. DISGEN bygger dock på en lokal lösning, vilket innebär att det inte finns något stöd för nätverk och medför att flera personer inte kan arbeta med släktträdet samtidigt. Detta leder till

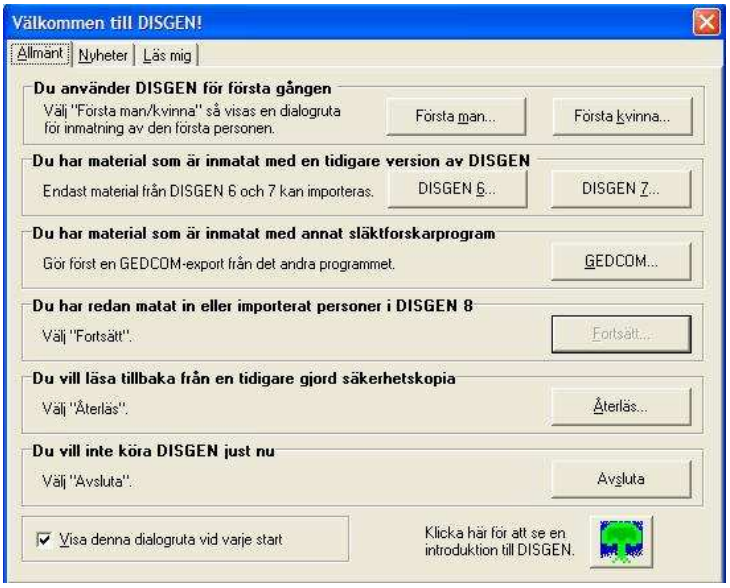
att arbetet med att digitalisera arkiven är en långsam och krävande process, då många personer ibland måste sitta och vänta på att någon annan skall bli klar. Att arbeta på detta sätt är inte effektivt, föreningen efterfrågar nu en lösning på detta problem. I övrigt är DISGEN ett fullt funktionellt program och innehåller många bra funktioner för arbete med släktträd. Målet med detta projekt var inte att skapa en version av DISGEN med nätverksstöd, utan det största målet var att få med de grundläggande funktionera i DISGEN samt att erbjuda möjligheten att jobba med ett och samma släktträd via en webbapplikation.

2.3.2 Funktionalitet

Några av de funktioner som DISGEN erbjuder användaren är:

- Samla och strukturera stora delar av data.
- Välja att visa släkter i olika perspektiv.
    1. Stamtavla
    2. Antavla
    3. Tabellform
    4. Trädform
    5. Släktbok
- Kontrollerar sina uppgifter.
- Presenterar sin forskning.

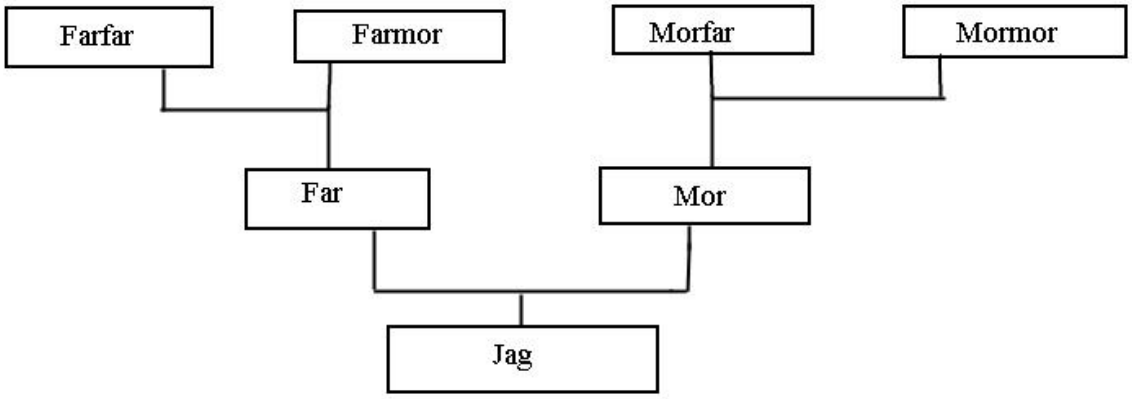
När DISGEN startas får användaren upp ett formulär Figur 3. Detta formulär visar de alternativ användaren har vid start. Det ena är att importera en GEDCOM fil, vilket innebär att det finns ett släktträd på fil som skall läggas in i programmet. Sedan finns möjligheten att lägga till en helt ny person, en så kallad obesläktad, vilket innebär att användaren inte vet vem som är föräldrar. Detta alternativ används också första gången en person skall läggas till i programmet. Det sista är att de går att fortsätta arbeta på ett släktträd genom att först söka upp en befintlig person i släktträdet och välja denna att arbeta vidare på. Dessa alternativ skall även den nya webbapplikationen erbjuda. Dessa skall ha samma funktion men design och upplägg kommer att skilja sig från det nuvarande.



Figur 3: Startruta i DISGEN

2.3.3 Släktträd

Släktträd används inom genealogi och är en karta som representerar familjrelationer i en form av trädstruktur [14]. Vid medeltiden användes ett släktträd för första gången och det var bilder som visade Jesus släkt. Det första ickebibliska släktträdet som var komplett och som visade alla relationer skapades året 1360. Det släktträdet skapades av Giovanni Boccaccio [13] som var en italiensk författare och poet. Han presenterade grekiska och romerska gudar i sina släktträd. Det finns olika sätt att representera ett släkträd, där de normala är att presentera det ifrån toppen till botten Figur 4. I toppen finns de äldsta generationerna och vid botten finns de yngsta generationerna.

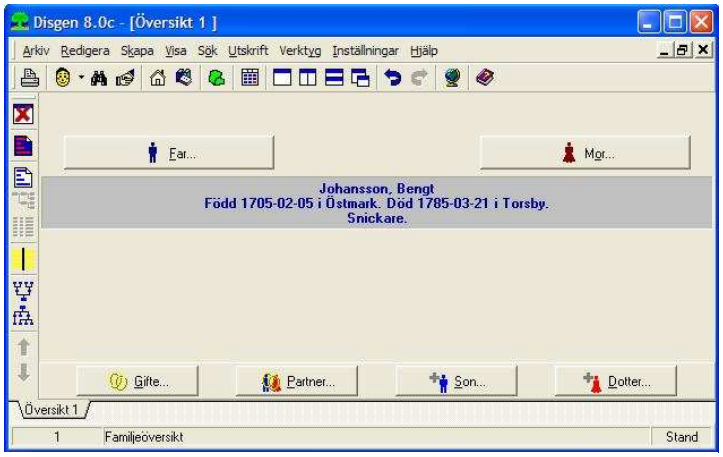


Figur 4: Representation av ett släktträd

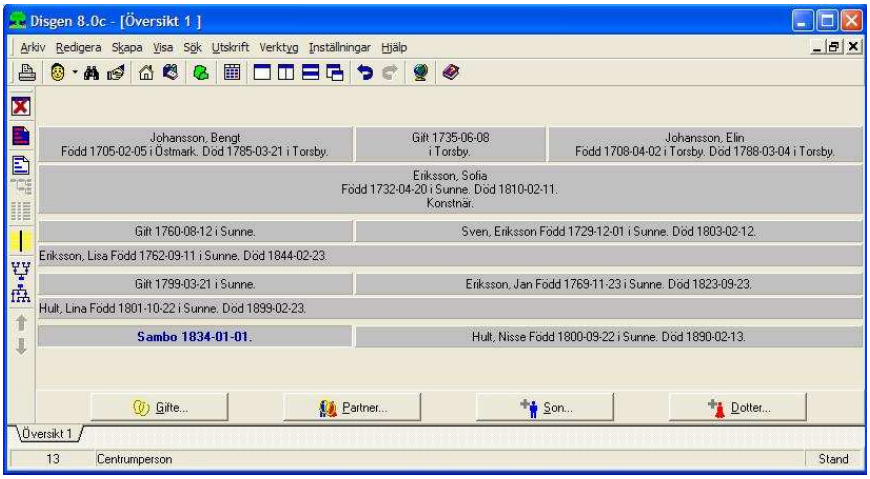
2.3.4 Visning av släkträd

När en person väljs i DISGEN byggs utifrån denna ett släktträd. Den person som trädet utgår ifrån kallas för centrumperson. Ifrån denna visas två generationer bakåt i tiden. Skulle det vara att det inte finns några personer bakåt visas knappar som ger användaren möjlighet att lägga till en mor eller en far [Figur 5].

Då det finns generationer bakåt i tiden listas dessa. Utifrån centrumpersonen visas också om denna har några relationer och i så fall vilken typ av relation det är. Namnet på partnern visas också samt om det finns barn listas dessa. Relationerna visas i kronologisk ordning så det på ett enkelt sätt går att följa [Figur 6].

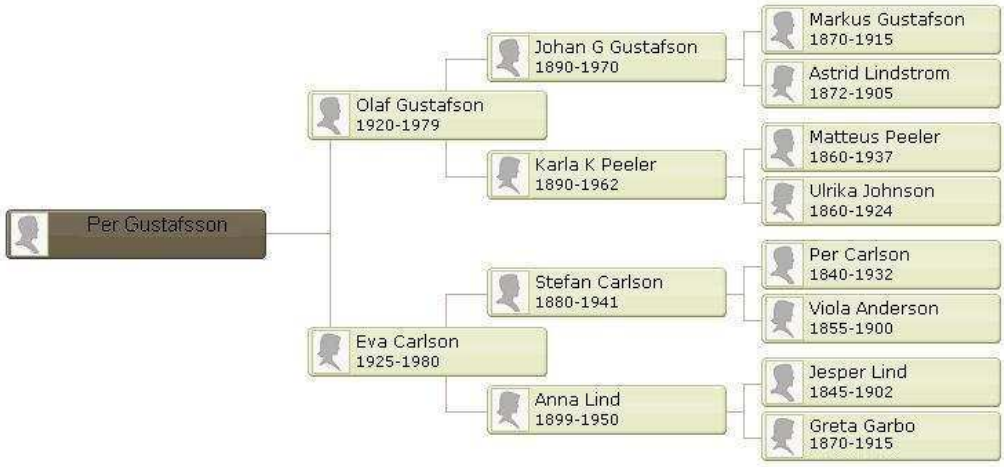


Figur 5: Bild som visar när generationer saknas i ett släktträd.



Figur 6: Ett släktträd med personer bakåt samt relationer och barn.

Även den nya applikationen kommer att erbjuda dessa möjligheter, dock kommer själva trädvisningen se lite annorlunda ut, då själva trädet inte skrivs ut neråt som det gör i DISGEN utan utskriften kommer att ske från vänster till höger för att få en bättre översikt av trädet [Figur 7].



Figur 7: Förslag på layout för släktträdet [9].

Där kommer släktträdet att visas liggande, längst till vänster är den person som har valts. Till höger visas sedan ett förinställt antal generationer. Tanken är att endast två generationer skall representeras i webbapplikationen.

Den information som varje gren skall innehålla är namn på personen och dennes födelsedatum samt dödsdatum om detta finns. Sedan skall födelse ort och dödsort finnas med. För den person som är längst till vänster skall lite mera information skrivas ut. Här skall dennes olika relationer presenteras i kronologisk ordning samt vilka barn som denne har i respektive relation [Figur 8]. Här ska även födelsedatum, dödsdatum och födelseort skrivas ut, samt datum för vigsel.

Figur 8: Exempel över hur informationen för centrumpersonen ska se ut

2.4 GEDCOM

Detta kapitel handlar om hur GEDCOM formatet används i DISGEN och strukturen av formatet. Strukturen av formatet beskriver hur en GEDCOM representeras i en textfil.

2.4.1 Koncept

När en exportering av ett släktträd i DISGEN utförs, skapas en GEDCOM fil som är i ett vanligt textformat med viss struktur. I bilaga [A.1] finns ett exempel över hur den fil kan se ut som innehåller ett släktträd för tre generationer.

Den GEDCOM version som kommer att användas i detta projekt är version 5.5 [4]. Den exporterade filen representerar en databas i formen av en sekventiell ström av relaterade poster. Varje post representeras som en sekvens av olika taggar som är arrangerade i en hierarki. Varje rad i filen innehåller ett nivå nummer i hierarkin, en tagg samt ett valfritt värde som Figur 9 visar. En tagg är det nyckelord som beskriver raden, t.ex. INDI som säger att nu kommer en individ, eller NAME som anger namnet på en individ.

En rad kan även innehålla en korsreferens eller en pekare till en annan post i databasen, detta anges då i formen @korsreferens@. Detta används då relationskopplingar mellan personer skall göras, eller för att visa att en person tillhör en viss familj.

Hierarkin byggs upp av ett nivånummer som står i början på varje rad där underordnade rader har högre nivånummer. Hierarkin tillåter en rad att ha underordnade rader och att de i sin tur kan ha underordnade rader. En serie av en eller flera rader utgör en post i databasen, där varje ny post börjar med det lägsta nivånumret i hierarkin alltså 0 (noll).

2.4.2 Strukturen

GEDCOM strukturen är uppbyggd av fyra stycken huvuddelar [7]. Längst upp skall det finnas ett huvud som innehåller information om det data som finns i filen, som vilken version det rör

sig om eller namn på den person som exporterat ut filen. Sist i filen skall det också finnas en TRLR del som står för TRAILER och innebär att detta är slutet på filen. Dessa två delar är obligatoriska, medan de övriga två inte behöver finnas.

De övriga två är olika typer av datarekords där den viktigaste är den som innehåller själva dataposterna från släktträdet, och som i sin tur är lagrat i ett antal olika delar.

- Familjeposter
- Individposter
- Källposter

I en familjepost finns information om äktenskap eller vilken typ av relation två individer har till varandra. Det står även vem som är make och vem som är maka samt information om dessa har några barn och i så fall vilka dessa är. En familjepost får endast innehålla en make samt en maka. I dagens samhälle där det kan förekomma att en person har flera partners hanterar GEDCOM detta genom att skapa en familj för varje relation, detta även om en individ har två partners samtidigt. GEDCOM standarden utgår också från att maken är en man och makan är en kvinna. Detta kan tyckas vara gammaldags, men det finns ingen begränsning som säger att en relation inte kan vara en homosexuell relation. EmiTree kommer därför ge användaren möjlighet att lägga till homosexuella relationer. Detta görs möjligt då enda relationen mellan två personer görs via korsreferenser som består av identifieringsnummer till individer och ingen kontroll görs om ett ID tillhör en man eller en kvinna.

En individpost innehåller information om de personer som användaren vet eller har upptäckt. Det finns en möjlighet att ange källor till den information som finns i dessa poster, och den informationen lagras i så kallade källposter.

Utöver dessa tre olika poster finns några till, men dessa kommer inte att tas upp i denna rapport då de inte används i detta projekt.

```
0 @1@ INDI
1 SEX M
1 NAME Fredrik/Larsson/
1 BIRT
2 DATE 11 FEB 1984
1 FAMC @2@
```

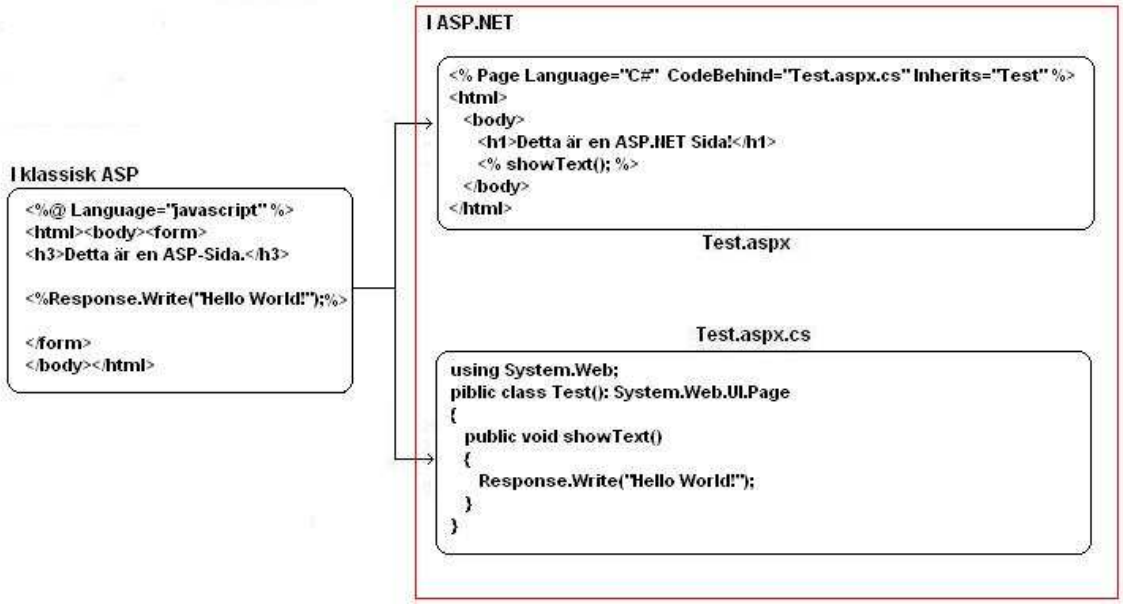
Figur 9: Exempel som visar hur en individ representeras i GEDCOM

2.5 ASP.NET

2.5.1 Bakgrund

Första version av ASP.NET [8] skapades 2002 av dataföretaget Microsoft och var en förbättring av ASP. Hur som helst är programmeringsmodell i ASP.NET helt olik jämfört med ASP. Orsaken varför Microsoft skapade ASP.NET var att uppmuntra fler utvecklare att skapa webbapplikationer. I ASP kod är programkod inbakad i HTML-kod enligt vänstra delen i Figur 10. I ASP.NET delas programkod och html-kod upp i två delar som högra delen i Figur 10 visar. Orsaken varför programkoden och html-koden delades upp var att göra applikationen snabbare och att applikationen skulle ha tillgång till hela .NET biblioteket.

ASP.NET är inte bakåtkompatibelt med ASP. ASP.NET är en stor del i Microsoft.NET Framework. Orsaken att ASP.NET valdes var att det är bra verktyg för att skapa webbapplikationer och är ett populärt verktyg i arbetslivet.



Figur 10: Skillnaden mellan ASP och ASP.NET

2.5.2 Microsoft.NET Framework

Microsoft.NET Framework [10] är en uppsättning av datorprogram som är en mjukvarukomponent av Microsoft operativsystem [10]. De tre största delarna som ingår i Microsoft.NET Framework är programspråk, server/klient teknologier och utvecklingsmiljöerna. Ett exempel på ett programspråk som används i Mircosoft.NET Framework är C# vilket är det som används i detta projekt. Den andra delen är server/klient

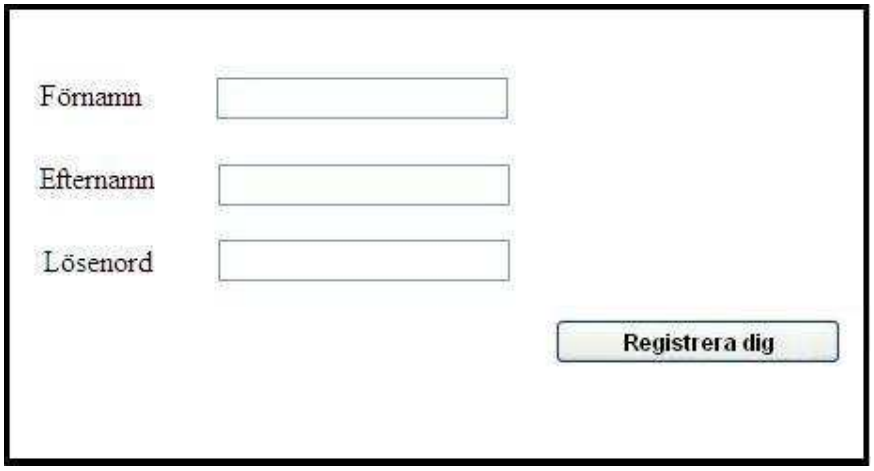
teknologier där ASP.NET finns med, och sista delen är utvecklingsmiljöerna dit Visual Studio.NET tillhör.

2.5.3 ASP.NET sida

Om information skall visas på en webbsida kan en ASP.NET sida skapas. En ASP.NET sida är en dynamisk hemsida där information på hemsidan kan ändras under hela tiden. ASP.NET använder filformatet ”ASPX” när en dynamisk hemsida skapas. ASPX-formatet är ett HTML-format där programkod är placerade i en separat fil eller inom ASP.NET taggar. Exempel på taggar är:

- <% %> , används när programkod skall implementeras i ASPX sidan.
- <%-- --%> , kommentar.
- <%= %> , används om ett uttryck skall adderas.
- <%@ %>, direktiv.

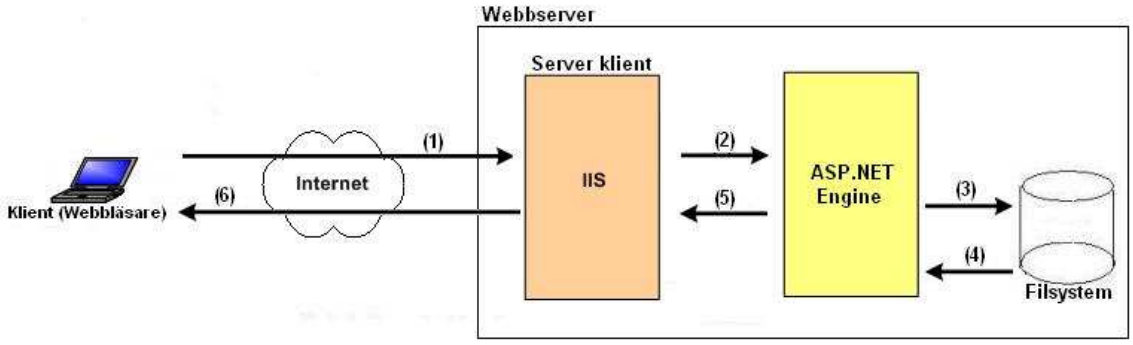
ASPX-formatet används främst för att skapa webbformulär på en ASP.NET sidan och ett webbformulär tillåter en användare att skriva in data till en webbapplikation. Ett webbformulär [Figur 11] innehåller olika delar som t.ex. textboxar, knappar och menyer etc., och kallas för kontroller.



Figur 11: Exempel på webbformulär

Programkoden som finns i en separat datafil eller i taggar utför uppgiften när en händelse sker, exempel på en händelse är när en användare trycker på knappen ”Registrera dig”.

2.5.4 Livscykeln hos en ASP sida



Figur 12: Hur en aspx sida begärs och returneras

När en ASPX sida begärs bearbetas olika steg, dessa olika steg ingår i livscykel hos ASP.NET [12]. I Figur 12 visas hur en ASPX sida begärs och vilka delar som används för att returnera en HTML sida så att användaren kan titta på den i sin webbläsare.

1. Det första steget är att klienten skickar ett http anrop till webbservern. Ett anrop till en viss hemsida görs med ett GET anrop t.ex. GET /sida.aspx HTTP/1.1
2. Webbservern ser att den begäran som klienten gör är till en dynamisk hemsida av typen ASPX. Då skickas begäran över till ASP.NET Engine som sedan tar han om anropet.
3. ASP.NET Engine undersöker om det finns någon kod med filnamnet sida.aspx på filsystemet.
4. Filsystemet returnerar den begärda kodfilen tillbaka till ASP.NET Engine. Denna kod fil består av en ASPX del och en *.cs del som Figur 10 visar.

   Nu utförs den process som är en sidas livscykel. Denna process är uppdelad i ett antal steg och utförs varje gång som en sida laddas.

   i. **Skapande**. Alla sidobjekt som finns på sidan skapas, såsom textboxar eller knappar. Alla dessa objekt placeras i en hierarki på sidan, som sedan utgör ett okjektträd.

   ii. **Initialisering**. En ”init” händelse startas för alla kontroller och för objekten. Händelsen ”init” skapades när ett kontrollobjekt har blivit skapat och har hand om initialiseringen av objekten.

   iii. **Laddning**. Först sparas alla tillstånd av kontrollerna ifrån det gömda fältet ”VIEWSTATE” som finns gömt någonstans på sidan. ”VIEWSTATE” är en

egenskap som anskaffar sidans tillstånd som ärvs ifrån kontrollerna. Efter det laddas alla data som användaren har skrivit in på sidan till kontrollerna. Sist aktiveras ”Load” händelsen för sidobjektet och för alla objekten.

iv. **Händelse behandling**. Sidan körs med alla deras operationer och egenskaper. Alla kontroller på sidan är under hantering. Ex. på en hantering är om texten på textruta har ändrats, en att användaren tryckt på en knapp.

v. **Översättning till HTML.** En ”preRender” händelse aktiveras för alla objekt som finns i ett kontrollträd. Kontrollerna som finns i kontrollträdet bestäms vid skapande av sidan. En ”preRender” händelse hanterar de sista ändringar av innehållet på sidan eller kontrollerna. Sedan är metoden ”SaveViewState” anropade på alla objekt för att spara deras tillstånd i gömda fältet ”VIEWSTATE”. När alla översättnings metoder är anropade på alla objekt generas HTML koden som sedan returneras och kan visas i klientens webbläsare.

vi. **Avallokering.** Detta är sista steget och här anropas ”Dispose” metoden på alla objekt och därpå ”Unload” händelsen aktiveras**. ”**Dispose” metoden avallokera alla objekt. Unload” händelsen tillåter att avslutnings uppgifter att köra som t.ex. stänga filer.

5. ASP.NET Engine returnera den genererade HTML koden till webbservern.

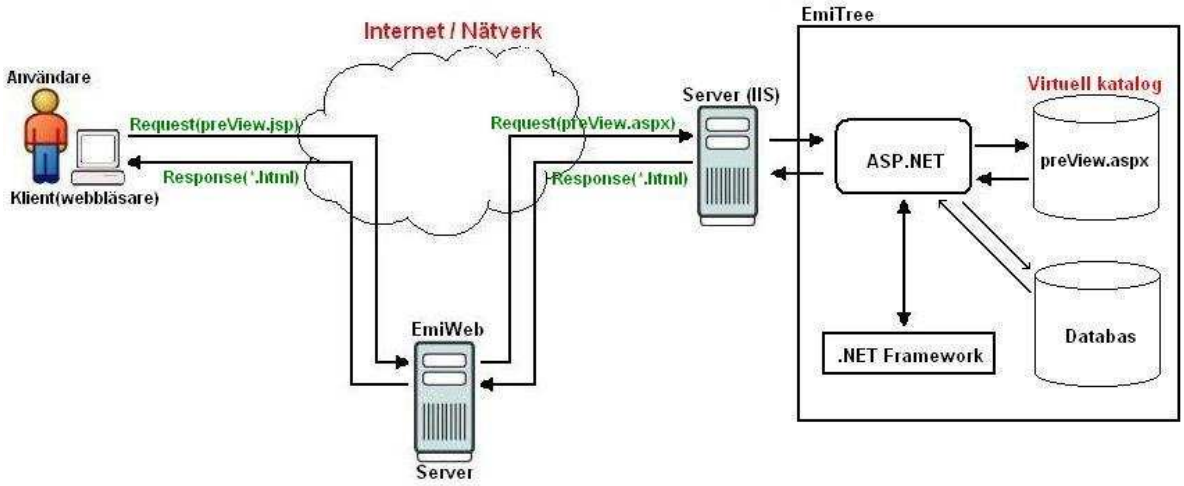
6. Webbservern skickar sedan vidare denna HTML kod till klientens webbläsare som tolkar och visar innehållet på skärmen.

2.6 EmiWeb

EmiWeb är det nu pågående projekt som utvecklas på Emigrantregistret för att sammanställa en rad olika arkiv som är samlade i en gemensam databas. Detta gör det möjligt att söka i flera olika arkiv från samma applikation, och att ge fler möjligheten att kunna ta del av dessa arkiv. EmiWeb är inriktat på den nordiska migrationen till och från Nordamerika. Intentionen är att det i framtiden skall innefatta flertalet arkiv innehållande migrationsrelaterat material. Idag är endast vissa arkiv representerade och omfattar framförallt den svenska emigrationen.

Detta program tillåter arkivägarna, där Emigrantregistret är en av dem att föra in nya emigranter samt att uppdatera och komplettera uppgifter för en redan existerande emigrant. Slutkunderna kan däremot inte ändra något, men de kan kommentera och komplettera uppgifterna. Enligt önskemål från Emigrantregistret skall släktträdsmodulen kopplas till EmiWeb [Figur 14]. Denna koppling skall fungera på det vis att om en emigrant finns i EmiWebs arkiv samt att denna även finns i släktträdet ska en koppling mellan dessa ske så att användaren kan få se denna emigrants släktträd direkt i EmiWeb. För att åstadkomma detta har flera olika metoder diskuterats. EmiWeb är baserat på Java lösningar och släktträdsapplikationen är baserat på ASP.NET lösningar går det inte ta del av varandras sessioner, det första alternativet som diskuterades var en länkning direkt från EmiWeb till EmiTree. Detta skulle innebära att EmiTree skulle startas upp och utnyttja EmiWebs sessioner för att logga in användaren för att sedan visa den valda personens släktträd, men efter diskussioner med Mathias Nilsson som arbetar med utvecklingen av EmiWeb blev lösningen att kopplingen ska ske via en IFRAME[11] i EmiWeb där en ASP.NET sida visas och skriver ut ett släktträd för den angivne emigranten. Funktionaliteten för detta släktträd skall vara begränsad till att bara tillåta visning i ett förinställt antal generationer. Om operationer skall utföras i trädet krävs en inloggning till den ordinarie släktträdsapplikationen, alltså att användaren skall logga in på EmiTrees webbsida direkt som Figur 1 visar. Figur 13 visar ett släktträd som hämtats från EmiTree och visas i EmiWeb för en emigrant.

Figur 13: Representation av släktträd i EmiWeb



Figur 14: Koppling mellan EmiWeb och EmiTree

Figur 14 beskriver hur denna koppling mellan användare, EmiWeb och EmiTree fungerar. När användaren som är inloggad i EmiWeb vill se en förhandsgranskning av en persons släktträd skickas en begäran från användarens webbläsare som begär en sida av formatet JSP [26] från EmiWeb som innehåller förhandsgranskningen. Denna sida i sin tur begär den information den behöver från EmiTree, genom att skicka en begäran med den information som behövs för att få fram ett släktträd. Detta anrop görs till en ASPX sida som finns på

servern som EmiTree körs från. EmiTree behandlar sedan anropet och skapar de objekt som behövs och hämtar den data som skall finnas från databasen, detta sammansätts sedan till en HTML fil [Figur 14] som returneras tillbaka till EmiWeb. I EmiWeb kommer denna HTML fil som returnerats från EmiTree att visas i en IFRAME. Den JSP fil som användaren begärde att få se genererar sedan sin HTML fil som i sista steget returneras tillbaka till användarens webbläsare som då kan se den information som begärdes i form av ett släktträd för en person [Figur 13].

2.7 Sammanfattning

I detta kapitel har olika delar gåtts igenom som skall ge läsaren en bra bakgrundsförståelse för att vidare kunna läsa och förstå rapporten. Här beskrivs bakgrunden och målet med projektet, vilket är att få fram en applikation som ska vara ett hjälpmedel vid konstruktion av släktträd. Denna applikation byggs med ASP.NET lösningar för att kunna erbjuda ett lätt och användarvänligt gränssnitt för webben, samt att de då blir stöd för flera användare som på ett effektivt sätt kan arbeta parallellt med varandra vid olika arbetsstationer via nätverk.
EmiTree ska sedan göras länkbart via den nu befintliga applikationen EmiWeb för att ge användaren möjlighet att snabbt och enkelt få se en persons släktträd. Denna koppling sker via en IFRAME som anropar en ASP.NET sida som visar släktträdet.

En stor del av applikationen är funktionaliteten att kunna importera ett släktträd från en GEDCOM fil, samt sedan kunna exportera tillbaka till en GEDCOM fil. Dessa funktioner krävs då det nuvarande programmet är DISGEN som inte har något nätverkstöd eller lagrar sina data i någon databas som går att koppla upp till via webben. Kravet från Emigrantregistret var att släktträdet skall lagras i en MYSQL databas, då det är detta som finns på deras servrar.
Kravet om att kunna exportera tillbaka från databasen till en GEDCOM fil finns eftersom GEDCOM är ett standardformat vid lagring av genealogisk information, och i framtiden kan det bli aktuellt att nya system skall användas och då måste ett sätt att lätt flytta data finnas tillgänglig. Andra funktioner som skall erbjudas är olika operationer som kan utföras på ett släktträd, och hit hör t.ex. att lägga till nya personer, ändra befintlig information för en person eller ta bort någon från trädet. Trädet skall också bli sökbart via en rad olika parametrar för att ge användaren ett effektivt verktyg för snabba och smarta sökningar som resulterar i att mindre tid behöver läggas på sökningar, som i nuläget kan vara en tidskrävande process.

3 Implementation

3.1 Översikt

Detta projekt innehåller tre stycken steg som skall utföras under implementationen. Det första som fick göras var en databasdesign, då ingen färdig databas fanns från början att jobba med. Nästa steg var att ta fram en programdesign i form av UML diagram där användarfall finns presenterade. Det sista steget var att ta fram en design över gränssnittet som skulle implementeras, och denna har även testats och godkänts av uppdragsgivaren. Samtliga av dessa tre steg är framtagna från den kravspecifikation som togs fram av uppdragsgivaren i början av projektet.

3.2 Kravspecifikation

Projektet innehåller två stycken kravspecifikationer. Först den tekniska specifikationen som tar upp applikationens funktionalitet. Den andra tar upp krav som skall finnas med i släktforsknings sammanhang, som vilken data som ska lagras.
Specifikationens moment är prioriterande efter prioteringsmodellen MoSCoW [20] som står för:

- MUST – Skall finnas med i projektet, annars är projektet misslyckat.
- SHOULD – Borde finnas med om det är möjligt.
- COULD – Kan finnas med om detta inte påverkar det övriga kraven.
- WONT – Skall inte finnas med nu, men kan bli aktuellt i framtiden.

3.2.1 Teknisk kravspecifikation

Applikationen måste ha (MUST):

1. ett webbaserat gränssnitt där data ska sparas i MySQL databas.
2. En sökfunktion där användarna av systemet ska kunna söka efter personer som finns i släktträdet.
3. funktion för importering av GEDCOM 5.5 filer samt för exportering från databas till GEDCOM 5.5 fil.
4. möjlighet att lägga till, ta bort och redigera personer i släktträdet och även kunna lägga till en obesläktad person i trädet.
5. Funktioner för att skapa relationer mellan personer.
6. En inloggnings möjlighet för att kunna tilldela olika tjänster till användarna av systemet.

7. Ett enkelt och användarvänligt gränssnitt.

Applikationen borde ha (SHOULD):

8. funktion för att visa två stycken vyer samtidigt där två personer kan ses samtidigt.
9. olika utskriftsmöjligheter som:
    - ansedel
    - tabellform
    - trädform
    - antavla
    - stamtavla
    - släktbok

10. funktion för att kontrollera släktskap mellan två valda personer.
11. en ortsfunktion.

3.2.2 Släktforskningsmässig kravspecifikation

Data som ska lagras är (MUST):

1. efternamn
2. förnamn
3. yrke
4. födelsedatum
5. födelseort/församling/län
6. källa för födelseuppgifter
7. dödsdatum
8. dödsort/församling/län
9. källa för dödsuppgifter
10. kön
11. fritext
12. relationer: utom äktenskap, gift, sambo och före äktenskap
13. vigsel datum
14. vigsel ort
15. källa för vigseluppgifter

Data som bör lagras (SHOULD):

16. dop datum
17. dop ort/församling/län
18. källa dop uppgifter
19. datum för begravning
20. begravnings ort/församling/län
21. källa för begravningsuppgifter
22. dop faddrar
23. dödsorsak

4 Design av databasen

4.1 Översikt

För att lagra det släktträd som skapas av applikationen har en databaslösning används, vilket ger programmet och användaren möjligheten att arbeta samtidigt med samma släktträd utan att störa varandra via ett nätverk eller till och med över Internet.
Ett krav för detta projekt var att databashantering systemet MySQL skulle användas, då det är denna som finns tillgänglig på Emigrantregistrets server.

4.2 Design

För att få en effektiv databaslösning har designfasen av denna varit en viktigt del under projektet. Kraven på databasen var stora då den skulle klara av att ta emot data från en GEDCOM fil vid importering av ett släktträd, och sedan kunna byggas vidare på utan att några relationer mellan personer förändras. Att även kunna exporteras tillbaka data till en GEDCOM fil när som helst under användandet utan att data går förlorad. Detta medförde då att planeringen av databasen var en viktigt och kritisk del i projektet. Med en dålig databasdesign skulle applikationen inte kunna fungera optimalt.

Genom att dela upp designen i två delar blev det en tydlig överblick över designen. Designen är uppdelad i E/R diagram och relationsmodeller. Genom att arbeta fram en bra design ges möjligheten att testa den på papper innan den skall implementeras och eventuella brister och fel hittas i ett tidigt stadium. Dessa tester visade sig vara mycket viktiga då flera olika problem uppstod, då mest i form av hur data skulle lagras på bästa sätt. Ett av de största problemen var möjligheten att importera flera GEDCOM filer. I en GEDCOM fil har varje post fått ett referensid som identifierar själva posten, detta ID har genererats av de program som exporteringen utfördes ifrån. Problemet med detta är att både familjeposter och individposterna delar på samma uppräkning av dessa referenser.

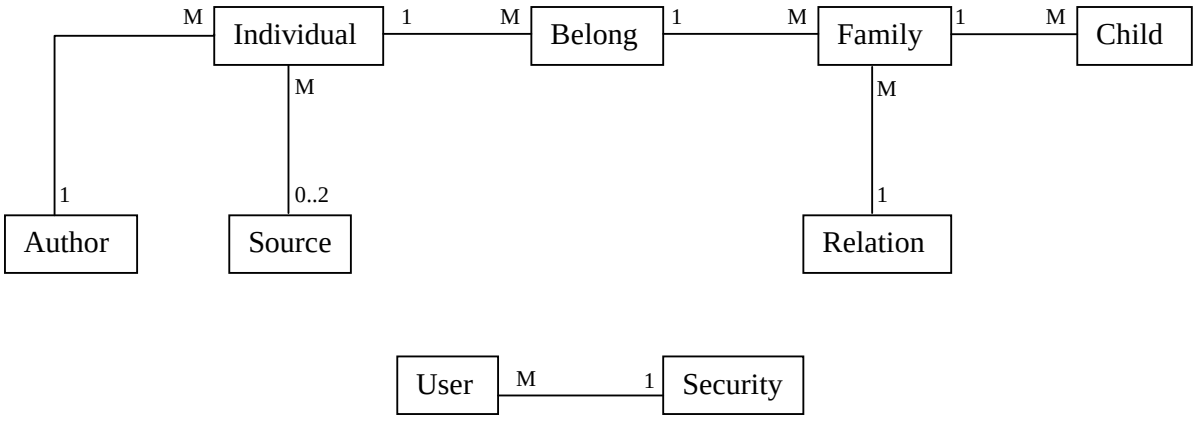
```
0 @7@ INDI
1 SEX F
1 NAME Astrid/Larsson/
0 @8@ FAM
1 CHIL @4@
1 HUSB @9@
1 WIFE @10@
0 @9@ INDI
1 SEX M
1 NAME Göran/Henriksson/
```

Figur 15: Exempel på hur referenser skapas i GEDCOM

I Figur 15 syns detta tydlig då den första individen har fått referensid 7, sedan kommer en familjepost och denna har referensid 8 och sedan återigen en individpost som då har referensid 9. Detta blir ett problem då dessa två ska ligga i olika tabeller i databasen, eftersom individer och familjer är två helt olika saker. Den naturliga lösningen skulle vara att ha en automatisk uppräkning av en primärnyckel, men i detta fall går inte denna lösning att tillämpa eftersom de då skulle bli fel vid exportering tillbaka till GEDCOM som vill ha det i formatet som Figur 15 visar.

När användaren sedan önskar att importera två olika GEDCOM filer uppstår också problemet att samma referensnummer återkommer i olika filer, och detta måste lösas. Att bara ta ett referensnummer direkt från GEDCOM filen och lagra det i databasen skulle vara den mest effektiva lösning, men dock bara om flera importeringar inte skulle tillåtas. Nu när flera importeringar tillåts måste en annan lösning tillämpas för att fortfarande garantera att varje post som lagras i databasen förblir unik och ett referensnummer endast förekommer en gång. Lösningen för båda dessa problem blev att låta databasen själv hålla reda på de referensnummer som lagrats och för varje ny post måste en fråga ske till databasen som svarar med nästa lediga referensnummer.

4.2.1 E/R Diagram



Figur 16: E/R Diagram för EmiTree

4.2.2 Tabeller

4.2.2.1 Individual

Tabell 1: Individtabellen

Kolumnnamn	Datatyp	Constraint	Referens
IID	Int	PK, NOT NULL	
AID	Int	FK	Author.AID
firstName	Varchar		
lastName	Varchar		
sex	Varchar		
birthDate	Varchar		
birthPlace	Varchar		
birthSource	Int	FK	Source.SID
deathDate	Varchar		
deathPlace	Varchar		
deathSource	Int	FK	Source.SID
occupation	Varchar		
changeDate	Varchar		
notes	Text		

Individtabellen innehåller alla personer som finns med i släktträdet, med den information som begärdes i kravspecifikationen för projektet. Primärnyckeln för tabellen är individnumret, och detta används för att unikt kunna identifiera en person i släktträdet. Skälet till att de olika datumposterna har datatypen varchar som en sträng är att vid konstruktion av ett släktträd kan innebära att datumet kan vara okänt. Lösningen har då varit att användaren som registrerade

personen i släktträdet endast har skrivit in årtal eller i värsta fall skrivit med formen (”1870 ??”) som gör att importering och tolkningen blir komplicerad. Detta problem har uppstått i och med att de föregående programmen som använts för att skapa släktträd inte har haft någon kontroll på att det är ett datum som skrivits in utan har tillåtit vad som helst. I denna första version av EmiTree har lösningen på detta problem blivit att endast spara årtal och datum som strängar.

4.2.2.2 Family

Tabell 2: Familjetabellen

Kolumnnamn	Datatyp	Constraint	Referens
FID	Int	PK, NOT NULL	
husband	Int	FK	Individual.IID
wife	Int	FK	Individual.IID
relationType	Int	FK	Relation.RID
relationDate	Varchar		
relationPlace	Varchar		
notes	Text		
source	Int	FK	Source.SID
changeDate	Varchar		

I denna tabell finns alla relationer som är mellan olika personer, samt information om relationen som vigseldatum och typ av relation. Vilken relation två personer har till varandra anges med en siffra som refererar till relationstabellen [Tabell 4]. De olika relationer som kan finnas är:

1. Gift
2. Sambo
3. Före äktenskap
4. Utom äktenskap

4.2.2.3 Child

Tabell 3: Barntabellen

Kolumnnamn	Datatyp	Constraint	Referens
FID	Int	PK, FK, NOT NULL	Family.FID
IID	Int	PK, FK, NOT NULL	Individual.FID

Denna tabell har som uppgift är att ange vilka individer som är barn i vilka familjer. Här är båda fälten unika, vilket innebär att en individ endast kan vara barn i en familj.

4.2.2.4 Relation

Tabell 4: Relationstabellen

Kolumnnamn	Datatyp	Constraint	Referens
RID	Int	PK, NOT NULL	
type	Varchar		

Relationstabellen används för att hämta ut vilken relationstyp som är i en familj. Genom att ha detta i en egen tabell finns möjligheten att i framtiden lägga till flera olika typer av relationer samt ta bort någon om detta skulle behövas.

4.2.2.5 Belong

Tabell 5: Tabell över föräldrar

Kolumnnamn	Datatyp	Constraint	Referens
IID	Int	PK, FK, NOT NULL	Individual.IID
FID	Int	PK, FK, NOT NULL	Family.FID

I denna tabell länkas individer till de familjer där han eller hon är man eller hustru. Anledningen med denna tabell är då en person ofta har haft flera äktenskap, och då kan dessa lätt registreras och hämtas via denna tabell.

4.2.2.6 Source

Tabell 6: Källtabell

Kolumnnamn	Datatyp	Constraint	Referens
SID	Int	PK, NOT NULL	
text	Varchar		

Alla poster i databasen kan ha en eller flera källor. Dessa anges här med ett unik id-nummer som identifierar dem.

4.2.2.7 Author

Tabell 7: Författartabell/Ägartabell

Kolumnnamn	Datatyp	Constraint	Referens
AID	Int	PK, NOT NULL	
name	Varchar		
org	Varchar		

Denna tabells uppgift är att ange vem som äger de olika posterna som finns i databasen. Syftet med detta är att vid importering eller tilläggning av personer finns ett intresse att logga vem som lagt till dessa. Detta är också ett sätt att kunna isolera olika poster från varandra om detta skulle behövas, t.ex. om det i framtiden skall kunna ge rättigheter till andra föreningar att fylla i sina egna resultat, och endast ge dem rättigheter att ändra data som ingår i deras domän.

4.2.2.8 User

Tabell 8: Användartabell

Kolumnnamn	Datatyp	Constraint	Referens
UID	Int	PK, NOT NULL	
SID	Int	FK, NOT NULL	Security.SID
firstName	Varchar	NOT NULL	
lastName	Varchar	NOT NULL	
userName	Varchar	NOT NULL	
password	Varchar	NOT NULL	
email	Varchar		

För att kunna identifiera olika användare som kan använda systemet används denna tabell [Tabell 8]. Den innehåller användarinformation och inloggningsuppgifter. Vilken säkerhetsnivå användaren har anges med en referens till security [Tabell 9].
För att öka säkerheten används MD5 kryptering på lösenordet.

4.2.2.9 Security

Tabell 9: Säkerhetstabell

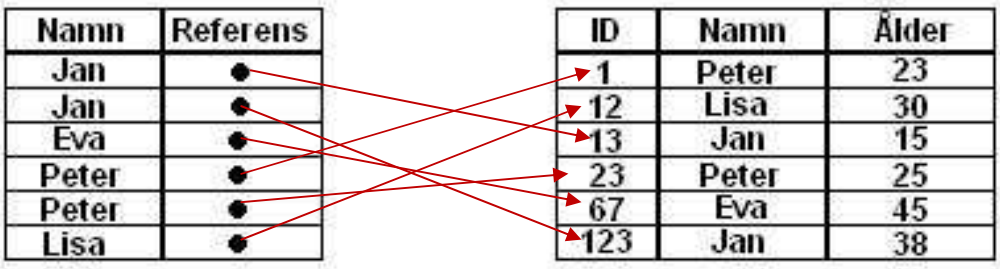
Kolumnnamn	Datatyp	Constraint	Referens
SID	Int	PK, NOT NULL	
type	Varchar		

Tabellen innehåller två fält där det ena är ett ID-nummer som identifierar vilken typ av säkerhetsnivå användaren har. Den sista är ett fält som med text anger vilken säkerhetsnivå som ID-numret representerar.

4.3 Effektivisering

4.3.1 Indexering

För att effektivisera sökningar i databasen har de fält som används oftast vid sökning indexerats. Ett index fungerar som en extra tabell som innehåller pekare till huvudtabellen, ungefär som ett register i en bok [16]. I Figur 17 finns två tabeller, den vänstra är indextabellen som innehåller fältet namn och ett referensfält som pekar på personen i den andra persontabellen (den högra). Indextabellen kan nu användas som ett register och vid sökningar efter t.ex. personer med namn Jan, då kan databasen kolla i indextabellen och få fram de pekare som pekar på rader i persontabellen som har namnet Jan i fältet namn. Detta går mycket fortare att utföra än att databasen måste stega igenom hela persontabellen för att hitta matchningar.



Figur 17: Exempel på indexering

För detta projekt har indexeringar gjorts i individ tabellen [Tabell 1] på några av de fält som skall vara sökbara. Detta gäller följande fält:

- firstname
- lastname
- birthDate
- birthPlace

4.3.2 Normalisering

För att undvika att dubbellagra data i databasen och därmed få en dålig design har normalisering tillämpats. Med normalisering menas att uppdelning sker med det data som riskerar att dubbellagras i egna tabeller. Exempel på detta kan vara om en tabell innehåller personuppgifter samt anger vilken ort som personen bor på. Då är det dålig design att för varje person som bor i Karlstad lagra detta i tabellen. Lösning blir att detta fält lyfts ut till en egen ortstabell, och sedan refereras en person till en ort i orttabellen [19].

4.4 Säkerhet

4.4.1 Transaktioner

För att alltid hålla den information som finns lagrad i databasen konsistent utförs alla databasoperationer inbakade i transaktioner. Med en transaktion menas att databasoperationer som är kritiska och måste genomföras helt eller inte alls grupperas ihop. Om ett fel skulle uppstå när en transaktion utförs återställs all det data som fanns lagrad innan transaktionen började och databasen är därmed fortfarande i ett konsistentt läge [21]. Detta är väldigt vikigt i detta projekt eftersom släktträdet är beroende av att alla kopplingar mellan personer, barn och familjer är intakta för att informationen som visas på skärmen skall vara korrekt.

4.5 Sammanfattning

I detta kapitel har designen av databasen presenteras, och en beskrivning har getts till samtliga tabeller som finns med i systemet. Efter önskemål från kunden har MySQL används för lagring av samtliga data i släktträdet.

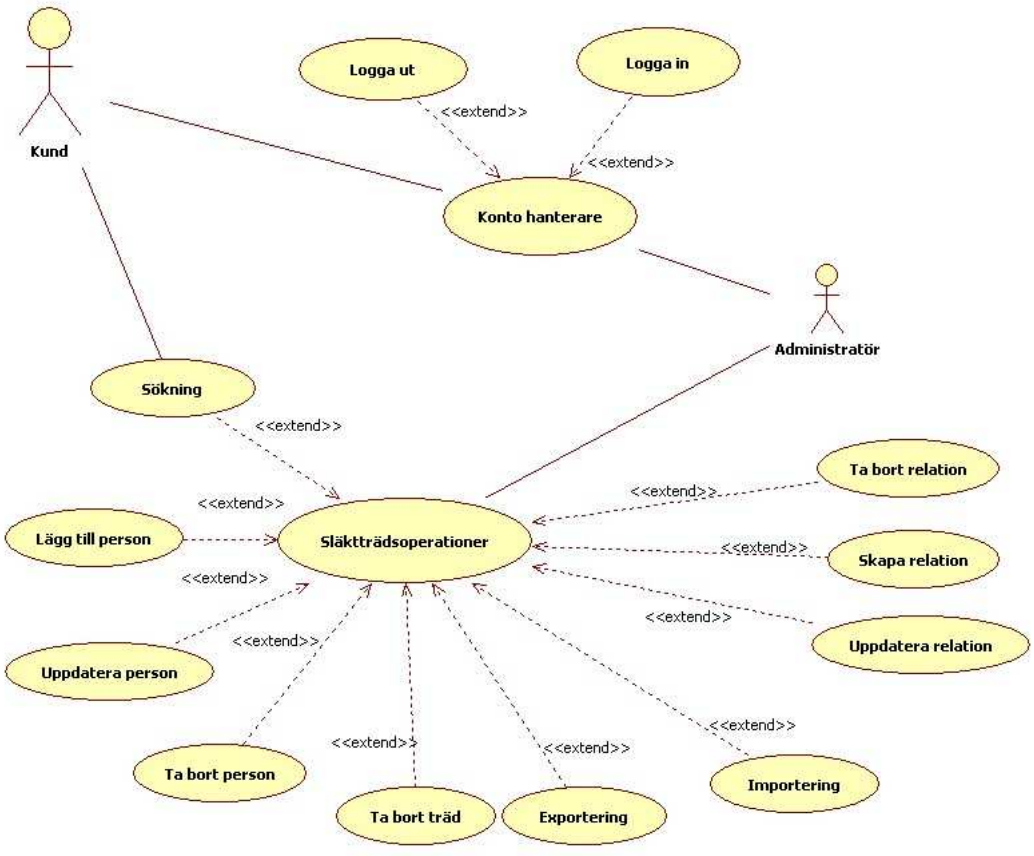
En översikt i form av ett E/R-diagram har också visats, där läsaren kan se hur tabellerna hänger ihop och vilka relationer som finns mellan dem.

Kapitlet tar även upp en del om effektivisering av databasen samt säkerhet för lagring av data till databasen.

5 Design av Programvaran

För att ta fram en bra design för programvaran som ger en bra grund att utgå ifrån i projektet användes de riktlinjer som beskrivs av Larman Craig [22] som tar upp objekt orienterad design och analys inom UP [23]. Ur denna bok har tre områden inom designfasen valts att tas med, användningsfallsdiagram som ger läsaren en visuell bild över de aktörer som finns representerade och vilka uppgifter dessa har inom systemet. Sedan beskrivs varje användningsfall ingående.
Det som inte ryms eller inte kan behandlas som något användarfall har placerats under sektionen för kompletterande specifikation.

5.1 Användningsfall



Figur 18: Användningsfalls diagram

5.1.1 Användningsfall ”Logga in”

Aktör: Användaren av systemet.

Beskrivning: Möjliggör för aktören att få tillträde till programmet.

Förvillkor: Aktören har ett användarkonto med användarnamn och lösenord.

Eftervillkor: Systemet loggade in aktören som nu kan använda systemet.

Normalflöde:

1. Systemet begär aktören att fylla i användarnamn och lösenord
2. Aktören fyller i sina användaruppgifter, och trycker på knappen ”logga in”.
3. Systemet kontrollerar inloggning mot databasen.
4. Systemet loggar in användaren, och flaggar honom som inloggad.

Alternativt flöde:
Om inloggning misslyckas.

1. Systemet meddelar aktören att inloggning misslyckades.
2. Systemet återställer sig i startläge där aktören kan försöka logga in igen.

5.1.2 Användningsfall ”Logga ut”

Aktör: Användaren av systemet.

Beskrivning: Möjliggör en aktör att logga ut ifrån programmet.

Förvillkor: Aktören är inloggad på systemet.

Eftervillkor: Aktören loggades ut ifrån programmet.

Normalflöde:

1. Användarfallet börjar när aktören trycker på knappen ”logga ut”
2. Systemet loggar ut aktören och flaggar honom som utloggad.
3. Systemet öppnar sidan för inloggning.

5.1.3 Användningsfall “Sökning”

Aktör: Användaren av systemet.

Beskrivning: Tillåter en aktör att söka efter personer i släktträdet.

Förvillkor: Aktören är inloggad i programmet.

Eftervillkor: Ett resultat listades med matchande sökresultat

Normalflöde:

1. Användarfallet börjar med att aktören valt att gå till söksidan.
2. Systemet erbjuder aktören ett antal parametrar att söka på.
    a) Förnamn

b) Efternamn
c) Kön
d) Födelsedatum
e) Födelseplats
f) Fritext

3. Aktören fyller i sökformuläret och trycker på knappen ”sök”
4. Systemet hämtar sökresultat från databasen.
5. Systemet visar en lista över sökresultatet för aktören.

Alternativt flöde:

Om inget sökresultat finns:

1. Systemet visar upp meddelandet ”inga träffar” för aktören.
2. Systemet återgår till ursprungsläge så att ny sökning kan genomföras.

Om aktören angett felaktiga tecken i sökfälten:

1. Systemet visar aktören ett felmeddelande ”Felaktig inmatning” och markerar vilket sökfält som felet uppstod i.

5.1.4 Användningsfall ”Ta bort träd”

Aktör: Administratör

Beskrivning: Möjliggör för aktören att ta bort ett släktträd.

Förvillkor: Aktören är inloggad

Eftervillkor: Släktträdet har tagits bort.

Normalflöde:

1. Användarfallet startar när aktören trycker på ”ta bort träd” knappen.
2. Systemet gör en förfrågan om han verkligen vill ta bort släktträdet.
3. Aktören trycker på ”ja” knappen.
4. Systemet tar bort släktträdet från databasen.

Alternativt flöde:

Om aktören väljer att avbryta borttagningen.

1. Systemet gör en förfrågan om han verkligen vill ta bort släktträdet.
2. Aktören trycker på ”nej” knappen.
3. Systemet avbryter processen.

5.1.5 Användningsfall ”Importera träd”

Aktör: Administratör

Beskrivning: Möjliggör för aktören att importera en släkträd från fil.

Förvillkor: Aktören är inloggad.

Eftervillkor: Ett släktträd importerades till databasen.

Normalflöde:

1. Systemet ber aktören att ange vilken fil som skall importeras.
2. Aktören väljer fil, och trycker på knappen ”importera”.
3. Systemet går igenom filen och lägger in släktträdet i databasen.
4. Systemet visar ett meddelande att importeringen är slutförd.

Alternativt flöde:

Ett fel uppstår vid importeringen.

1. Systemet får ett fel vid importeringen.
2. Systemet utför en rollback på databasen så att den lagrade data som finns är konsistent.
3. Systemet meddelar aktören att ett fel uppstod.
4. Systemet återställer sig till grundläget så att aktören kan göra ett nytt försök.

Aktören väljer att avbryta importeringen:

1. Systemet utför en rollback så att inga ändringar har skett till databasen.
2. Systemet meddelar aktören att importeringen är avbruten.
3. Systemet återgår till utgångsläget

5.1.6 Användningsfall ”Lägg till person”

Aktör: Användaren av systemet.

Beskrivning: Möjliggör för aktören att lägga till en person i ett släktträd.

Förvillkor: Aktören är inloggad på systemet med korrekta rättigheter.

Eftervillkor: En ny person har lagts in i släktträdet

Normalflöde:

1. Användarfallet startar med att aktören trycker på knappen ”Lägg till person”.
2. Systemet erbjuder aktören ett antal fält som kan fyllas i.

a) Förnamn
b) Efternamn
c) Kön
d) Yrke
e) Födelsedatum
f) Födelseort
g) Födelsekälla
h) Dödsdatum
i) Dödsort
j) Dödskälla
k) Fritext

3. Aktören fyller i informationen och trycker på knappen ”lägg till”.
4. Systemet lagrar personen i databasen.
5. Systemet omdirigerar aktören till sidan för att visa släktträd.
6. Systemet sätter den nya personen som centrumperson.

Alternativt flöde:

Aktören gör en felaktig inmatning i formuläret:

1. Systemet visar aktören ett felmeddelande ”Felaktig inmatning” och markerar vilket sökfält som felet uppstod i.

Fel uppstår vid lagring till databasen:

1. Systemet utför en rollback på databasen så att den lagrade data som finns är konsistent.
2. Systemet meddelar aktören att ett fel uppstod.
3. Systemet återställer sig till grundläget så att aktören kan göra ett nytt försök.

5.1.7 Användningsfall ”Ta bort person”

Aktör: Användaren av systemet

Beskrivning: Möjliggör för en aktör med rätt rättigheter att ta bort en person i släktträdet.

Förvillkor: Aktören är inloggad i programmet samt att släktträdet inte är tomt och person ifråga får inte ha några relaterande relationer.

Eftervillkor: En person har tagits bort från trädet

Normalflöde:

1. Användarfallet börjar när aktören trycker på knappen ”ta bort” för en vald person.
2. Systemet frågar aktören om han är säker på att personen skall tas bort.
3. Aktören trycker på ”ja”.
4. Systemet kollar efter relaterande personer.
5. Systemet tar bort personen från individtabellen [Tabell 1].
6. Systemet meddelar aktören att personen är borttagen.

Alternativt flöde:

Om personen inte är frikopplad, dvs. det finns relationer mellan personen och en annan person.

1. Systemet meddelar aktören att personen inte kan tas bort då det finns relaterade relationer till denna person.
2. Systemet ber användaren att först ta bort dessa relationer och sedan försöka igen.

5.1.8 Användningsfall ”Uppdatera personinformation”

Aktör: Användaren av systemet.

Beskrivning: Möjliggör en aktör att uppdatera information för en person.

Förvillkor: Aktören är inloggad på systemet med korrekta rättigheter.

Eftervillkor: Den valda personens information har uppdaterats.

Normalflöde:

1. Användarfallet börjar med att aktören trycker på knappen ”ändra” för en vald person.
2. Systemet gör en kontroll så att inmatad information är korrekt.
3. Systemet uppdaterar den valda personens information i databasen.
4. Systemet omdirigerar aktören till släktträdet, och sätter den valda personen som centerperson.

Alternativt flöde:

Aktören har gjort en felaktig inmatning.

1. Systemet visar aktören ett felmeddelande ”Felaktig inmatning” och markerar vilket textfält som felet uppstod i.

5.1.9 Användningsfall ”Exportera träd”

Aktör: Administratören

Beskrivning: Möjliggör för aktören att exportera ett släktträd till en GEDCOM fil.

Förvillkor: Aktören är inloggad.

Eftervillkor: Släktträdet har exporterats till en GEDCOM fil.

Normalflöde:

1. Användarfallet börjar med att aktören går till sidan för exportering av databas.
2. Systemet ber aktören att välja vilken fil som exporteringen skall ske till.
3. Aktören väljer fil, och trycker sedan på ”exportera”.
4. Systemet går igenom databasen och skriver innehållet till en GEDCOM fil.
5. Systemet meddelar aktören att exporteringen är genomförd.

5.1.10 Användningsfall ”Visa träd”

Aktör: Användaren av systemet

Beskrivning: Möjliggör för aktören som är inloggad att se släktträdet.

Förvillkor: Aktören skall vara inloggad.

Eftervillkor: Släktträdet visades.

Normalflöde:

1. Användarfallet börjar när aktören gjort en sökning och tryckt på en person i sökresultatet.
2. Systemet hämtar information om vald person och skriver ut på skärmen.
3. Systemet hämtar de relationer som personen har, och de tillhörande barnen till dessa och visar det i trädet.
4. Systemet hämtar mor och far samt morföräldrar och farföräldrar, och visar detta i trädet.

Alternativt flöde:

Sidan för visning av träd öppnas då ingen person är vald.

1. Systemet omdirigerar aktören till söksidan.

5.1.11 Användningsfall ”Skapa relation”

Aktör: Användaren av systemet.

Beskrivning: Tillåter aktören att skapa en relation för en person, t.ex. att skapa en familj.

Förvillkor: Aktören är inloggad, och har rätt rättigheter.

Eftervillkor: En relation skapades för den valda personen.

Normalflöde:

1. Användarfallet börjar när aktören trycker på knappen ”skapa relation”.
2. Systemet erbjuder aktören ett antal fält att fylla i.
    a) Relations typ
        a. Gift
        b. Sambo
        c. Före äktenskap
        d. Utom äktenskap
    b) Relationsdatum
    c) Ort där giftermål etc. utfördes.
    d) Källa för relationen.
    e) Fritextfält
3. Aktören fyller i formuläret och trycker på ”skapa”.
4. Systemet utför en kontroll på inmatad information.
5. Systemet lagrar informationen i databasen.
6. Systemet omdirigerar aktören till släktträdet och sätter den valda personen som centerperson.

Alternativt flöde:

Aktören har gjort en felaktig inmatning.

1. Systemet visar aktören ett felmeddelande ”Felaktig inmatning” och markerar vilket textfält som felet uppstod i.

5.1.12 Användningsfall ”Uppdatera relation”

Aktör: Användaren av systemet.

Beskrivning: Ger aktören möjlighet att ändra informationen för en relation.

Förvillkor: Aktören är inloggad och har rätt rättigheter.

Eftervillkor: Den valda relationen uppdaterades.

Normalflöde:

1. Användarfallet startar när aktören väljer att uppdatera en relation.
2. Systemet hämtar den information som finns om relationen och visar den på skärmen.
    a. Relations typ
        i. Gift

ii. Sambo
iii. Före äktenskap
iv. Utom äktenskap

b. Relationsdatum
c. Ort där giftermål etc. utfördes.
d. Källa för relationen.
e. Fritextfält

3. Aktören kan nu ändra informationen som finns i textfälten.
4. Aktören trycker på knappen ”ändra”.
5. Systemet genomför en kontroll på den inmatade informationen.
6. Systemet uppdaterar informationen till databasen.
7. Systemet omdirigerar aktören till visning av släktträdet och sätter den person vars relation är vald som centerperson.

Alternativt flöde:

Aktören har gjort en felaktig inmatning.

1. Systemet visar aktören ett felmeddelande ”Felaktig inmatning” och markerar vilket textfält som felet uppstod i.

5.2 Kompletterande specifikation

Denna del behandlar andra krav och önskemål som inte på ett enkelt sätt kan tas med som användningsfall. Det som tas upp här är delar som ingår i URPS+ (usability, realiability, performance, supportability)[22] och andra kvalitetskrav.

5.2.1 Funktionalitet

5.2.1.1 Loggning och felhantering

Alla fel som uppstår vid användandet av systemet skall loggas och bevaras på bestående medium, som fil eller databas. Vid fel skall dessa rapporteras till logfilen med information om det fel som uppstod och om möjligt varför felet uppstod. Detta för att systemadministratören ska få en möjlighet att snabb isolera och åtgärda felet.

5.2.1.2 Säkerhet

Allt användande av systemet kräver att aktören identifierar sig genom att göra en inloggning till systemet. Aktörerna har också olika rättigheter för att förhindra att obehöriga utför operationer som för dem inte är tillåtna. Alla personers vistelse skall också loggas i en fil (en loggfil) så vid eventuella fel eller missbruk går det att gå tillbaka och se vem som gjort vad och vid vilken tidpunkt.

5.2.2 Användarvänlighet

- Systemet skall inte använda färger som är starka och som upplevs som svåra att läsa mot. Även färger som associeras med färgblindhet skall undvikas.
- Den text som visas skall vara tydlig och inte ha en för liten storlek.
- För att kunna använda systemet fullt ut kommer ett krav på skärmupplösning att sättas till minst 1024*768 pixlar. Detta beror på att ett släktträd innehåller mycket information som skall visas för användaren. Vid lägre upplösning riskerar texten bli mycket liten och svårläst för personer med sämre syn. Även överblicken över arbetsytan blir svårjobbad.
- Systemet skall alltid meddela användaren vad som sker, och vid långa processer skall information om hur långt gången denna process är och en beräknad återstående tid skall visas.

5.2.3 Pålitlighet

5.2.3.1 Återhämtningsförmåga

De fel som uppstår skall systemet alltid göra ett försök att rädda sig ifrån för att förhindra att aktören upplever problemen vid användandet. Vid fel som inte kan repareras kommer systemet att meddela aktören detta och om nödvändigt kräva att systemet startas om. En felrapport skrivs först till logfilen med information om det fel som uppstod och om möjligt varför felet uppstod.

5.2.3.2 Prestanda

Användaren av systemet skall inte behöva vänta längre än några sekunder för att få någon respons på den förfrågan de gör. Problemet med systemet är att det skall arbetas över ett nätverk och av flera samtidiga användare. Det kommer inte att finnas något krav på specifik

internetuppkoppling eftersom tjänsten skall kunna köras från så många uppkopplade datorer som möjligt där alla kanske inte har tillgång till bredband. Däremot kan krav komma att sättas på den server som skall tillhandahålla tjänsten så att flera användare kan utnyttja systemet samtidigt. Framtida tester på prestandan kommer att visa om eventuell gräns för antal samtidiga användare skall sättas.

5.2.4 Underhåll

För att underhåll skall kunna göras på systemet skall detta vara väldokumenterat. Alla klasser skall vara kommenterade, och alla metoder skall beskrivas. Då detta följs kan framtida projekt utföras snabbare och effektivare när tid och pengar inte behöver spenderas på att först förstå vad koden betyder och har för uppgift.

5.2.5 Implementation

Systemet skall vara implementerat med ASP.NET och använda programspråket C#. ASP.NET valdes eftersom detta är en plattform som används mycket för just webbaserade system fast detta var inget krav ifrån uppdragsgivaren. En annan fördel med detta är att det kan användas av alla webbläsare då det enda som skickas till användarens webbläsare är vanlig HTML, den övriga koden hanteras på serversidan.

5.3 Sammanfattning

Detta kapitel har tagit upp alla de användarfall som togs fram utifrån den kravspecifikation som fanns att tillgå. Varje användarfall är beskrivet i detalj i form av ett normalflöde som beskriver hur systemet fungerar, när allt fungerar som de ska, samt alternativa flöden som beskriver eventuella fel och felhanteringar.

Kapitlet tar också upp en kompletterande specifikation som beskriver de områden som inte passar in som användarfall som t.ex. användarvänlighet, prestanda och pålitlighet.

6 Design av användargränssnitt

Eftersom uppdragsgivaren vill ha ett grafisk användargränssitt var det viktigt att diskutera användargränssnittsdesign. Det kan finnas människor som är skrämda av användargränssnittsdesign, de tror att det bara handlar om grafisk design. Men grafisk användargränssnittsdesign handlar mer om hur funktionen av ett gränssnitt fungerar och hur en användare önskar att ett gränssnitt ska fungera. Regler och riktlinjer används i designen av användargränssnittet för att uppnå bra funktionalitet av gränssnittet. Ett användargränssnitt är bra designat om programmet uppför sig exakt som användaren vill att de ska fungera. Målet med programmet är att få det fungera som användaren vill att det ska fungera. Genom att göra utvärderingar och tester kan gränssnittets användarvänligt utvärderas.

6.1 Regler och Riktlinjer

Regler och riktlinjer [18] används främst för att förbättra gränssnittet och funktionalitet av gränssnittet. Det som kommer användas främst är följande regler och riklinjer för att uppnå bra användbarhet av släktträdsverktyget.

- *Håll det enkelt***:** Användaren av programmet kommer att anta att användargränssnitt kommer använda den enklaste modellen som möjligt. Detta medför att det inte kan bli problem för användaren. Genom att göra funktionaliteten lätt kommer användaren lättare kunna använda programmet.
- *Val:* För att minska problemen som kan uppstå då användaren måste göra massor av val hela tiden, är det viktigt att jobba för att minska de val som inte är nödvändiga.
- *Tala användarens språk***:** Det är viktigt att programmet kommunicerar med användarens språk så användaren förstår programmet. Genom att använda rätt ord, meningar och koncept som användaren känner till så kommer det att öka användarens förståelse. Programmets användare kommer främsta att vara släktforskare och det gör att systemet måste anpassas och inte använda sig av datatermer utan använd mer fackspråk istället.

- *Involvera användaren:* Genom att låta användaren delta i en dialog under utvecklingen minskar det risken för programmeraren att misstolka vad användaren kräver av programmet.
- *Konsekvens och standard:* Konsekvens och standard är två fundamentala principer för bra design av användargränssnitt. Samma information skall konsekvent presenteras på samma ställe på alla skärmbilder. Webbformulären skall vara utseendemässigt likadana för att underlätta för användaren. Genom att gränssnittet är konsekvent och följer standarden kommer det hjälpa användaren att känna igen sig och det medför att användbarheten kommer att öka.
- *Återkoppling*: Alla händelser ska ge någon form av återkoppling. Med återkopplingar menas att systemet ska berätta när detta arbetar och tala om vad systemet arbetar med. Återkoppling till användaren ska ges direkt när en användare utför en operation.
- *Återhämtning:* Alla användare kan göra fel, t.ex. att de trycker på fel knapp. När ett fel har skett måste programmet återhämta sig till hur programmet var innan felet uppstod. Så återhämning måste finnas i släktträdsverktyget.

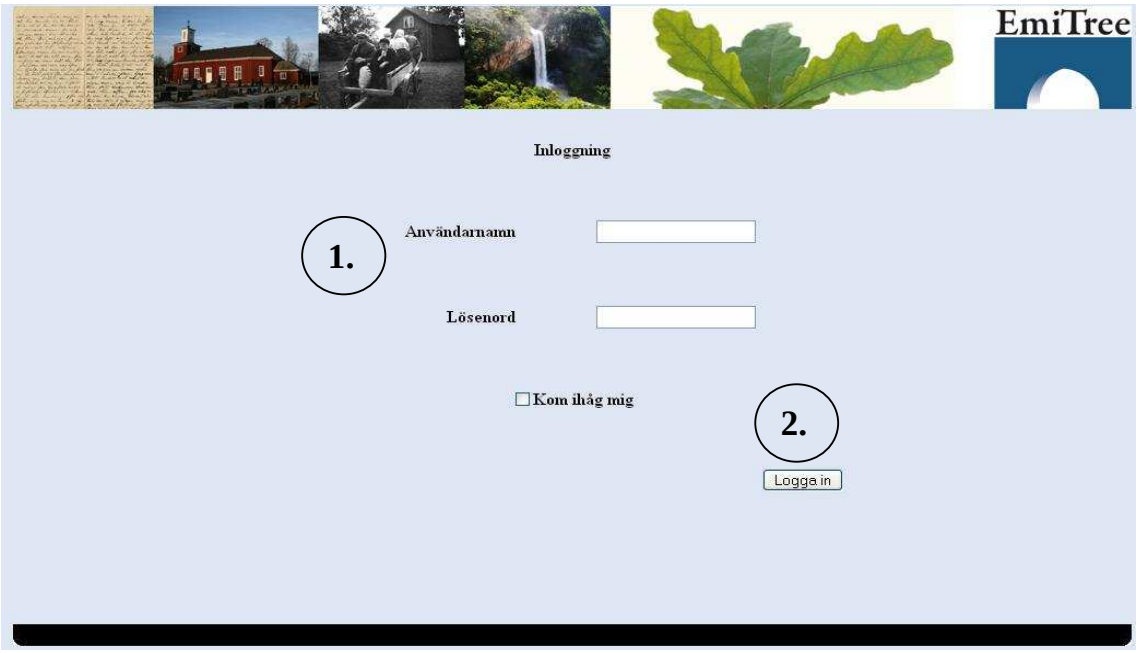
Dessa riktlinjer och regler måste vara anpassade för en webbapplikation eftersom EmiTree kommer vara webbaserat. Enligt Spolsky [18] finns två problem när ett gränssnitt ska designas för webben och de är: tidsfördröjningar och begränsningar på HTML. Tidsfördröjningar går inte att undanröja helt därför det beror på internetstrukturen. För tillfället garanterar inte Internet att det inte finns tidsfördröjningar. HTML är designat för enkla applikationer vilket kan betraktas som problem, men detta problem löstes genom att använda dynamisk HTML.

6.2 Gränssnittet

Gränssnittet av släktträdsverktyget kommer att erbjuda följande operationer som login, importering av släktträdet, visning av släktträdet, visning av lägg till person, skapa relation och sök.

6.2.1 Login

Gränssnittet på login är väldigt simpelt och enkelt. I Figur 19 visas en skärmdump av login, med tillhörande beskrivning.



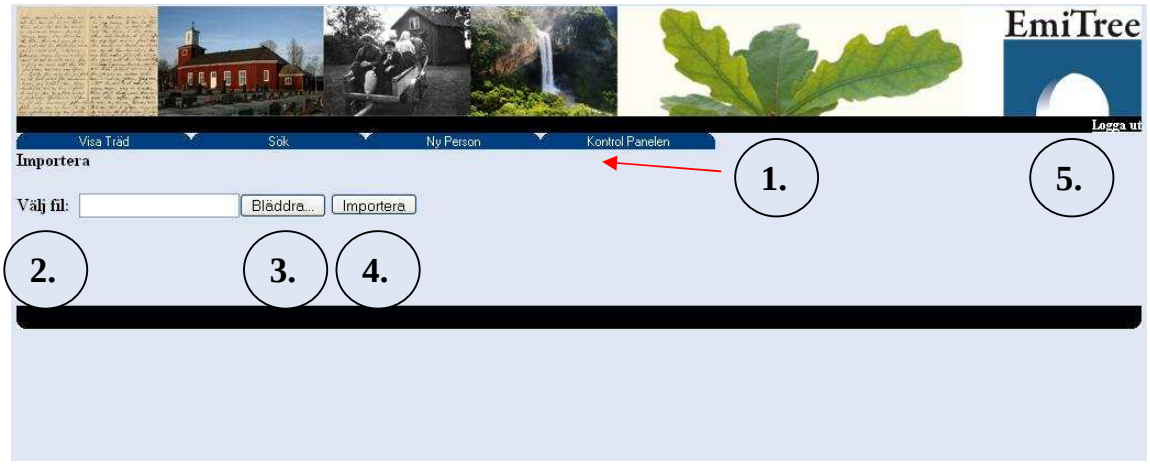
Figur 19: Login sida

Beskrivning:

1. **Inloggning**: I inloggnings delen har etiketter och textboxar används för att kunna mata in användaruppgifter.
2. **Logga in**: Detta är knappen för att mata in användaruppgifter i släktträdsverktyget.

6.2.2 Importering av släktträdet

I [Figur 20] visas en skärmdump av importering, med tillhörande beskrivning.



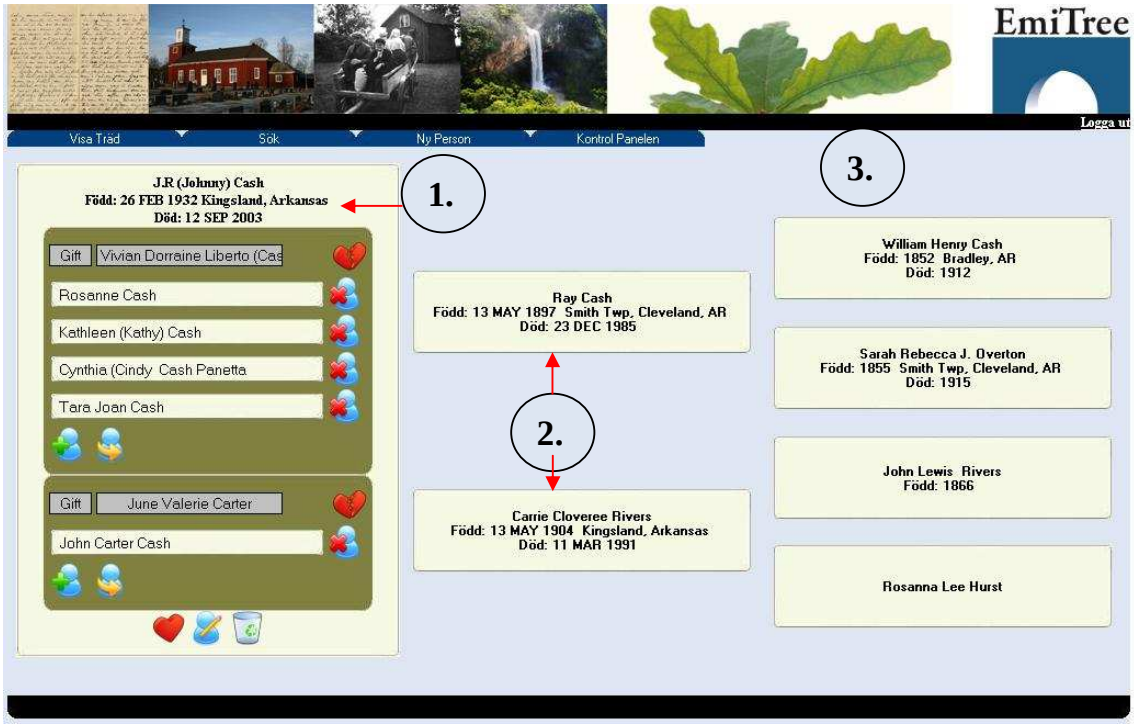
Figur 20: Gränssnittet för importering

Beskrivning:

1. **Huvudmenyn**: Huvudmenyn i programmet där länkar finns till olika sidor.
2. **Textbox:** Textbox för sökvägen av filen.
3. **Bläddra**: Bläddraknapp för att ange fil.
4. **Importera**: Denna knapp används till att importera den valda filen.
5. **Logga ut**: Logga ut knappen, genom att trycka på den loggas användaren ut ifrån programmet.

6.2.3 Visning av släktträd

I släktträdet som visas i Figur 21 används ljusa färger och inte mycket av starka färger. Orsaken är att det blir lättare att läsa och det blir mycket bättre för ögon. Tre generationer kommer att finnas vid visning av släktträdet. I fFigur 21 visas en skärmdump av släktträdet, med tillhörande beskrivning.



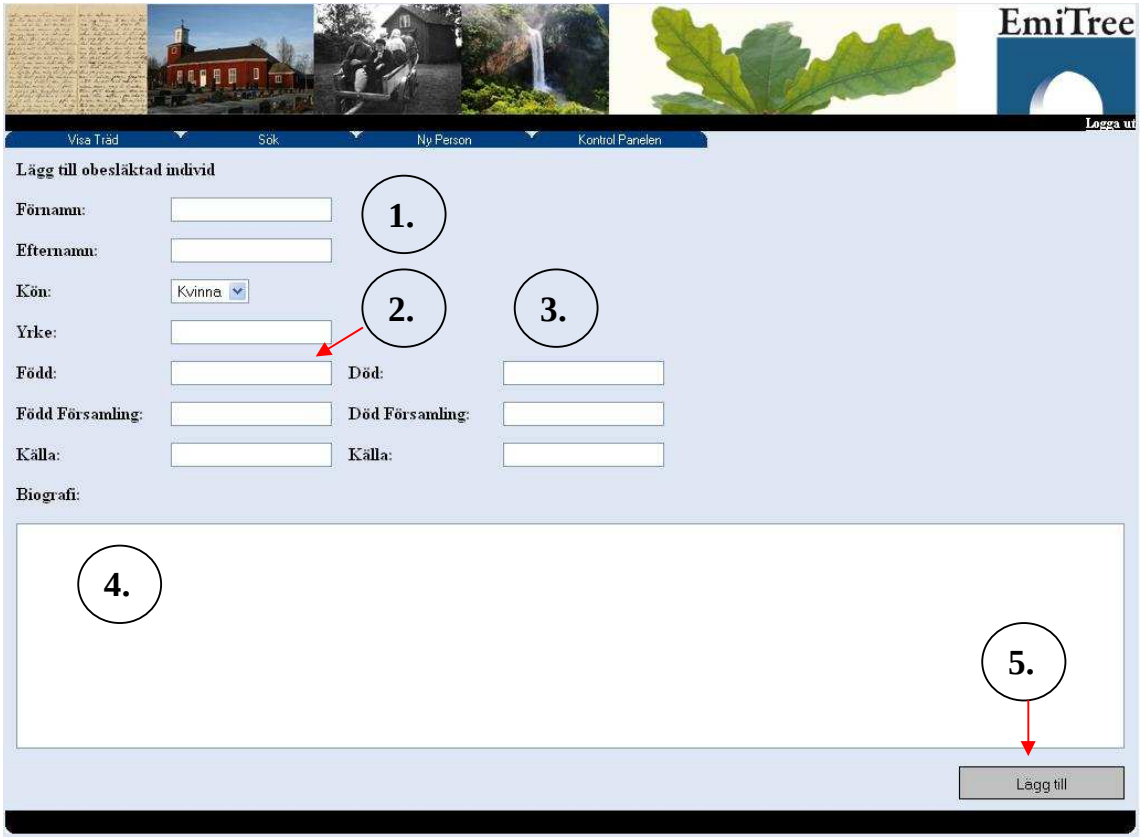
Figur 21: Gränssnittet för visning av släktträdet

Beskrivning:

1. **Centrumperson**: Detta är centrumpersonen i släktträdet. I denna del kommer det att finnas information om individen. Denna information kommer vara namn, när han är född och vart han föddes. I denna del finns knappar som hjälper användaren att utföra trädoperationer. Dessa knappar finns beskrivna i bilaga [B].
2. **Far och mor**: Denna del visar centrumpersons far och mor. Här kommer det också att finnas information om far- och mor individen. Om inte en centrumperson har någon far eller mor kommer det finnas två knappar för att lägga till far eller mor, eller möjligheten att koppla befintliga personer.
3. **Farfar/farmor och morfar/mormor**: Sista delen i släktträd är farfar/farmor och morfar/mormor. Denna del kommer det också att finnas information om individen.

6.2.4 Visning av lägg till person

Gränssnittet av lägg till person presenteras i Figur 22. Etiketter och textboxar används för att kunna inmata data.



Figur 22: Gränssnittet där ny person läggs till

Beskrivning:

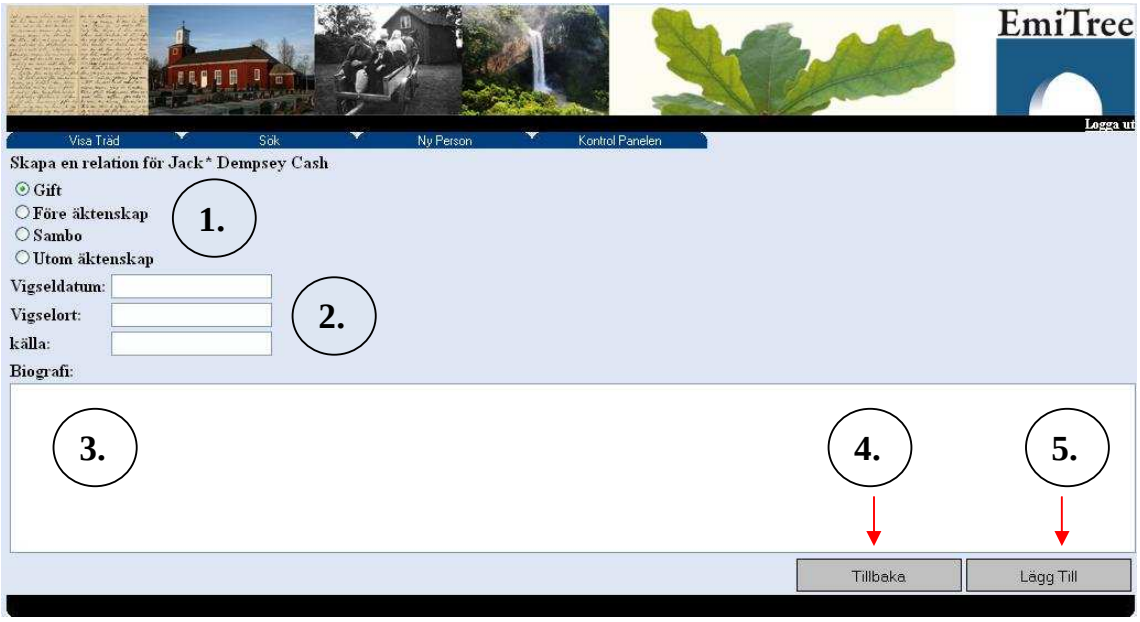
1. **Information**: Basfakta om individen ska skrivas in i denna del.
2. **Födelseinformation**: Här ska födelseinformation skrivas in. När individen föddes, vart han föddes och vilken källa användaren har används sig av.
3. **Dödsinformation**: Här ska dödsinformation skrivas in. När individen dog, vart han dog och vilken källa användaren har används sig av.
4. **Notis**: I notisdelen kan användaren skriva in ytterligare information om individen. Denna information är fritext, och kan ha valfritt innehåll.

5. **Lägg till person**: Detta är knappen för att användaren ska kunna lägga till person i släktträdet.

6.2.5 Skapa relation

I fFigur 23 visas en skärmdump av skapa relation, med tillhörande beskrivning.

På denna sida skapas en relation mellan två människor. Följande relationer är möjliga i programmet: gift, före äktenskap, sambo, utom äktenskap. Etiketter och textboxar används för att mata in relationsinformation.



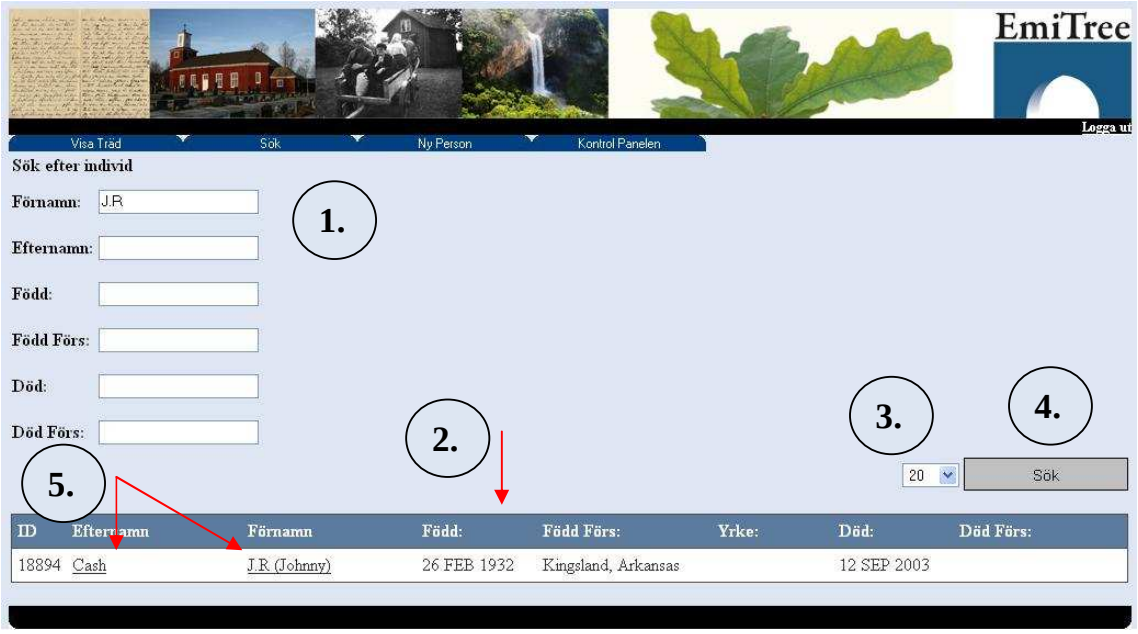
Figur 23: Gränssnittet för skapande av relation

Beskrivning:

1. **Relation:** Radioknappar som anger relationstypen.
2. **Relationsinformation:** Textboxar där relationsinformation ska matas in.
3. **Biografi:** Fritext fält.
4. **Tillbaka:** Tillbaka knapp.
5. **Lägg till:** Knapp för att lägga till relation.

6.2.6 Sök Person

I Figur 24 visas en skärmdump av söka person, med tillhörande beskrivning. Där det går att söka på förnamn, efternamn, födelsedatum, födelseförsamling, dödsdatum samt dödsförsamling. Sökresultatet kommer nedanför i en tabellform, i detta fall är J.R(Johnny) Cash det enda sökresultatet.



Figur 24: Gränssnittet för sökning av personer.

Beskrivning:

1. **Sökparameter:** Här anges sökparametrarna för sökningen.
2. **Sökresultatet:** Sökresultatet i tabellform visas här.
3. **Max antal sökträffar:** Max antal sökträffar. 20,50,100 max antalet sökträffar går att välja på.
4. **Sök knapp:** Knapp för att söka på individer.
5. **Individlänk:** Genom att trycka på antingen förnamnet eller efternamnet länkas användaren till visning av släktträdet med den person som valdes.

6.3 Sammanfattning

I detta kapitel diskuterades användargränssnittsdesign och släktträdsverktygets gränssnitt.
Det beskrivs även i kapitlet hur ett gränssnitt ska utvärderas. I avsnittet [6.1] presenteras regler och riktlinjer för att uppnå bra användbarhet av släktträdsverktyget. De främsta reglerna och riktlinjerna är håll det enkelt, tala användarens språk, och involvera användaren.
För att förstå gränssnittet bättre beskrivs gränssnittets alla delar, se [6.2]. Några av gränssnittets delar är visning av släktträdet, visning av lägg till person och sök person.
För att hitta användbarhetsproblemen användes användningstester i det verkliga livet.

7 Resultat

Kapitlet tar upp information om de tester som utförts under projektet och resultaten för dessa presenteras.

7.1 Testresultat

För projektet var en period på minst tre veckor för testning av systemet inplanerat. Målet med testperioden var att samla in så mycket olika testdata som möjligt för att kunna åtgärda eventuella fel samt få synpunkter på utseende och funktionalitet som kan vara aktuellt att tillföra i senare projekt. En av de viktiga testerna var att låta personalen själva få använda systemet, för att sedan rapportera in sina synpunkter och funderingar. En annan viktig del i testningen är att utföra olika systemtester som ger svar på om programmet uppför sig som det skall när olika sekvenser av operationer utförs samt prestanda för systemet som t.ex. responstider.

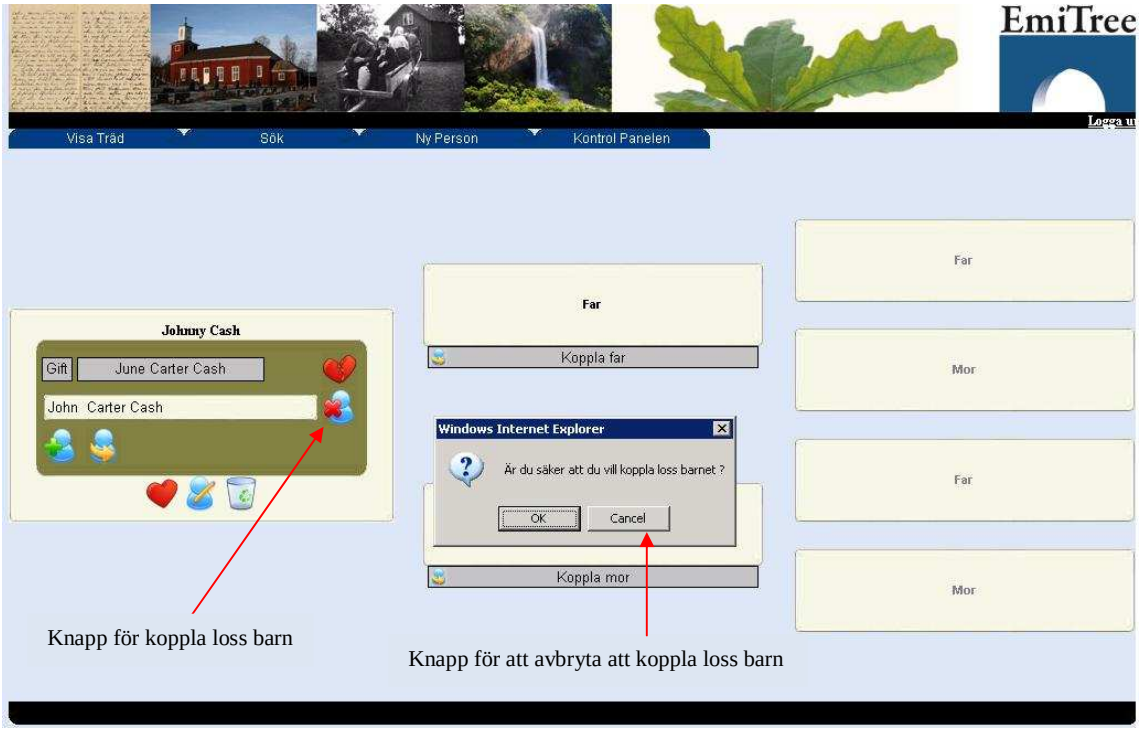
7.1.1 Användartester

Personalen har utfört tester i systemet. De har skrivit ner synpunkter och angett problemen med systemet. Användartestarna gav följande resultat:

- Problem 1: När ett nytt barn ska läggas till en familj som endast har en angiven förälder uppstod ett fel. Detta fel berodde på att båda föräldrarna behövdes för att skapa ett barn, vilket inte alltid är sant. Det finns familjer med bara den ena föräldern och dessa familjer skall också kunna ha barn. Detta fel åtgärdades genom att tillåta ensamstående föräldrar.

- Problem 2: Uppdatering av person. Problemet inträffar när användaren ska uppdatera en person. Vid uppdatering av en person skapades en tom födelsekälla och en tom dödskälla även om fälten för källorna som finns i Figur 22 är tomma. Det ska inte ske därför databassystemet skulle komma då få fullt med tomma fält. Problemet är nu åtgärdat och det åtgärdas genom att systemet kollar om källfälten är tomma. Om källfälten är tomma då tilläggas ingen information i databasen.

- Problem 3: Varningsruta vid att koppla loss barn fungerar inte korrekt. Problemet inträffar när användaren har tryckt på knappen ”koppla loss barn” som finns i Figur 25. Genom att ha tryckt på att koppla loss barn knappen anropas en varningsruta som också visas i Figur 25. Genom att användaren trycker på knappen ”Cancel” som finns i varningsrutan vill användaren att händelse ”Koppla loss barn” inte ska ske. Men i själva verket sker bortkoppling ändå och användaren kommer tro att barnet inte har kopplats loss. Lösningen på detta problem är att fixa varningsrutan så den kommer att fungera korrekt. Detta problem skall åtgärdas i mån av tid eller behöver implementeras först i framtida projekt.



Figur 25: Bildexempel på när barn kopplas loss samt varningsruta.

- Problem 4: Systemet tillåter inte att ta bort källor. För tillfälligt finns det ingen operation för att ta bort källor, detta eftersom inget krav för detta fanns, så detta utelämnades. Denna funktionalitet kommer att läggas till vid framtida projekt.

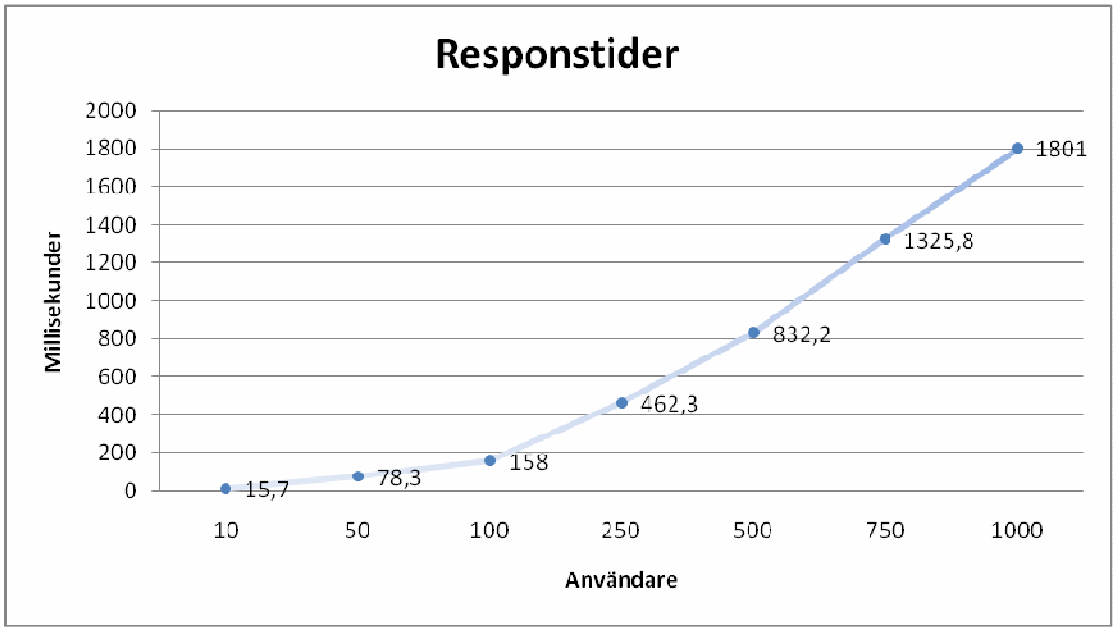
- Problem 5: När användaren skulle t.ex. utföra en sökning, vill denna utföra sökningen genom att trycka på Enterknappen. Då detta gjordes hände inte det som önskades och ingen sökning utfördes. Detta problem åtgärdades direkt genom att en standard operation sattes vid entertryckning.

- Problem 6: Exportering fungerar inte korrekt. Problemet med exportering är att det inte går att lägga exportfilen i vilken mapp som helst utan exportfilen måste läggas på en katalog som ligger på servern. En lösning är att låta användaren ladda ner filen från servern.

7.1.2 Systemtester

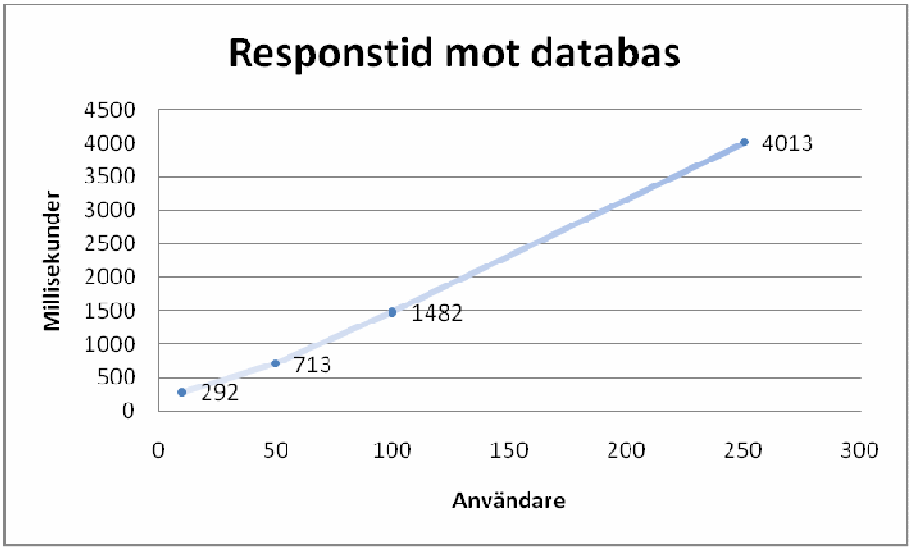
För att ta reda på vilken prestanda som systemet har, utvecklades en testmiljö som kunde simulera att flera användare använde systemet samtidigt. För att få till flera användare användes en lösning där varje användare kördes i en egen tråd. Alla tester är testade från tio stycken användare upp till 1000 stycken samtidiga användare, och medeltiden för operationen räknas ut. Gemensamt för diagrammen är att y-axeln visar medeltiden i millisekunder, och x-axeln anger antalet användare. Dessa tester skall ses som ett ungefärligt och inte som ett exakt resultat då ett eget utvecklat testprogram har använts vid testerna. Detta medför att det är svårt att veta vilka faktorer som påverkar systemet vid utveckling av testmiljöer.

Det första testet som gjordes kontrollerade hur systemet hanterar när flera användare begärde en viss sida. Figur 26 visar resultatet från detta test.



Figur 26: Responstider för olika samtidiga amvändare

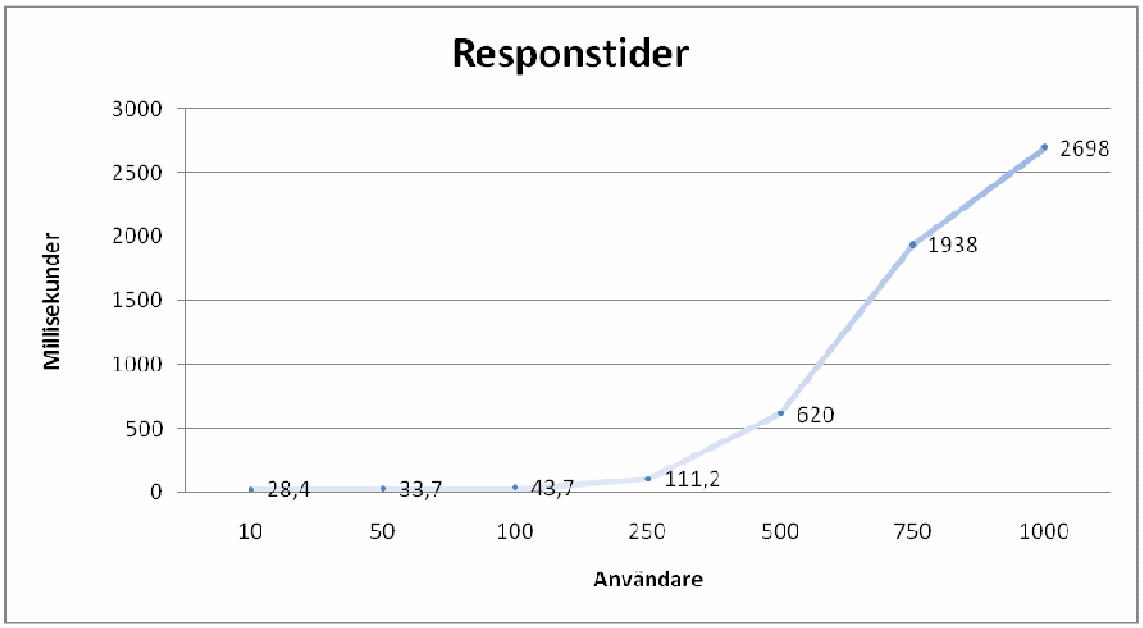
Nästa systemtest som utfördes var att se hur systemet fungerade tillsammans med databasen. Testet utfördes som det första förutom att den sida som anropades hade som uppgift att hämta data från databasen och skriva ut detta. Figur 27 visar resultatet från detta test, och här kunde endast upp till 250 stycken användare kunnat testas. I dessa test var det återkommande att cirka 50 % av fallen resulterade i ett server fel vars orsak var att MySQL har ett antal portar att dela ut när flera uppkopplingar sker samtidigt. Dessa portar har sedan en tid som de är upptagna även efter det att de inte används längre. På grund av denna tidsfördröjning hinner alla portar bli upptagna och när en ny uppkoppling ska ske blir det fel.



Figur 27: Responstider för körning mot databas

Det gemensamma för båda dessa test är att de endast testar hur mycket systemet och servern klarar av att hantera. De testar inte hur systemet reagerar när flera användare i form av verkliga personer använder systemet som inte kan göra flera sidaccesser i sekunden. För att även få in hur systemet reagerar för vanligt bruk gjordes även ett test där datorn simulerade användare som antingen sökte i databasen eller bara ville kolla på ett släktträd, med en viss väntetid mellan accesserna från en till tio sekunder. Denna metod valdes endast för att inte överbelasta systemet, och ingen korrekt statistisk metod har används för detta ändamål. Detta gör att systemet inte alls blir lika belastat och ger en bättre bild av hur vanligt användande påverkar systemet. Figur 28 visar resultatet från detta test, som visar att systemet klarar av minst 1000 stycken samtidiga användare, men vid 500 stycken användare börjar responstiderna stiga kraftigt.

För att användaren inte skall få för mycket väntetid bör det maximala antalet samtidiga användare ligga mellan 500 och 750 stycken.



Figur 28: Responstider vid simulering av vanligt användande

7.2 Sammanfattning

Kapitlet har redovisat resultaten från de test som utförts på systemet, både användarfall och prestandatester som utförts. Användartesterna var en viktig del i testningen då det är mycket bra att få in information från de användare som är tänkt skall arbeta med systemet i framtiden. Prestandatester utfördes för att undersöka hur många samtiga användare som systemet klarade av, och detta gjordes genom att ett eget program togs fram för att simulera flera användare som anropade funktioner på webbsidan och mätte tiden det tog för servern att svara.

8 Slutsats

Kapitlet tar upp utvärdering av projektet där den metod, samt vilka olika steg som används under projektet. Sedan redogörs vad som gjordes bra och vad som resulterade i problem. Slutligen avslutas kapitlet med en lista om huruvida projektet har nått de resultat och mål som uppdragsgivaren hade samt vad som finns att tillägga vid framtida projekt.

8.1 Utvärdering av arbetet

Uppgiften var att skapa ett enkelt och lättanvänd system för att arbeta med släktträd via webben. Detta togs fram utifrån en kravspecifikation som tagits fram av uppdragsgivaren, och som fungerat som mål under projekttiden. Denna del tar upp hur arbetet genomfördes och vilket resultat som uppnåddes.

8.1.1 Planering och förberedelse

Innan projektet drog igång gjordes en tidsplanering som bestod av tre stycken huvuddelar.

1. Experimentdesign
2. Applikationskonstruktion
3. Systemtester och utvärderingar

De moment som ingick i experimentdesignen var design av databasen som tog upp hur denna skulle struktureras och vilka olika tabeller som skulle behövas för att kunna hålla all den information som ett släktträd innehåller. Programdesign var också en viktig punkt i planeringen där de olika klasserna designades med tillhörande operationer och egenskaper.
Ett förslag över gränssnittet togs fram för att uppdragsgivaren kunde ge sin synpunkt på denna och på detta vis snabbt komma fram till en design som skulle fungera för att i framtiden när det var dags att utveckla själva applikationens gränssnitt snabbt kunna se något resultat.

Då flera användare kommer att kunna använda programmet samtidigt krävs det en viss kontroll och säkerhet. Detta togs också fram i början av projektet tillsammans med uppdragsgivaren, som kom med förslag på säkerhetsåtgärder som sedan utvärderades huruvida dessa var aktuella och genomförbara i en tekniks och programmeringsmässig synvinkel.

När den grundläggande designen och planeringen av systemet var klart övergick projektet i den andra fasen. Denna fas handlade om själva utvecklingen av systemet, och det var i denna fas som all programmering, databasdesign och framtagning av användargränssnittet utfördes utefter den design som gjorde vid den första fasen.
Även denna fas delades in i tre delmål för att få en bättre struktur på utvecklingen.
De tre delarna som behövde genomföras var först att göra det möjligt att importera ett befintligt släktträd från en GEDCOM fil. Sedan behövdes även en funktion för att exportera ut ett släktträd från systemet till en GEDCOM fil så att användaren om han önskar skall kunna förflyta ett släktträd mellan olika släktträdsapplikationer utan att data skall gå förlorad.
Den sista delen var att konstruera själva webbsidan som skulle tillhandahålla visningen av släktträdet och ge användaren en rad olika funktioner att arbeta med i släktträdet.

Vid slutet av utvecklingsfasen gjordes en demonstration för de anställda över vad som hade gjorts och hur det fungerar. Detta gjordes för att få de anställda informerade om systemet och hur detta fungerar så dessa sedan kunde sätta sig på var sitt hål och utföra egna användartester som sedan rapporterades in

I och med att projektet gjordes på detta viset resulterade detta i att projektet blev mycket lyckat och den tidsplanering som från början togs fram kunde följas utan problem.

8.2 Projektet

Under projektet fanns framgångar som beskrivs i avsnittet [8.2.1]. I avsnittet [8.2.2] undersöktes om projektets krav blev uppfyllda. I detta delkapitel undersöks även alternativa metoder och lösningar [8.2.3] och rekommendationer för framtida funktionaliteter [8.2.4].

8.2.1 Framgångar med projektet

Den största framgång med projektet var att samtliga krav från uppdragsgivaren blev klara och ett fungerande släktträdsverktyg togs fram. Efter en kort presentation av verktyget för Emigrantregistret blev resultatet att de var mycket nöjda. De anser att systemet har potential att utvecklas och användas i framtida projekt. För tillfället finns det inga gratis webbaserat släktträdsverktyg på marknaden som har den funktionalitet som Emigrantregistret efterfrågar så detta projektets släktträdsverktyg är välkommet. En annan viktig framgång med projektet var att importering/exportering av GEDCOM formatet gick snabbt och lätt att genomföra. En orsak till detta var att en grundlig genomgång av GEDCOM strukturen utfördes för att ta fram de viktigaste nyckelorden.

8.2.2 Kraven

För att undersöka om kraven är uppfyllt jämfördes de krav som har MUST som finns beskriven i avsnittet [3.2]. Under varje krav ges en förklaring varför det är uppfyllt.

Tekniska Kraven MUST, se avsnitt [3.2.1]:

1. Applikationen måste ha ett webbaserat gränssnitt där data ska sparas i MySQL databas.
   Detta krav är uppfyllt.

   Det webbaserade gränssnittet finns beskrivet i avsnitt [6.2]. Allt av systemets data är sparade i MySQL tabeller, detta var ett krav ifrån projektets uppdragsgivare. MySQL tabeller och hur data är sparade i tabeller, se avsnittet [4.2.2].

2. Applikationen måste ha en sökfunktion där användarna av systemet ska kunna söka efter personer som finns i släktträdet.
   Detta krav är uppfyllt.

   Sökfunktionen finns beskriven i avsnittet [6.2.6.] Hur sökfunktionen kan förbättras i framtiden finns beskriven i punkt fyra i avsnittet [8.2.4].

3. Applikationen måste ha en funktion för importering av GEDCOM 5.5 filer samt för exportering från databas till GEDCOM 5.5 fil.
   Detta krav är uppfyllt.

   GEDCOM formatet används för att importera/exportera släktforskningsdata och GEDCOM formatet är beskrivet i avsnittet [2.4]. Hur data är sparade i databaserna, se krav [1] ovanför. Däremot återstår problemet med att få den exporterade filen sparad på klientens dator, se bugg 2 i bilaga [C]

4. Applikationen måste ha en funktion för möjlighet att lägga till, ta bort och redigera personer i släktträdet och även kunna lägga till en obesläktad person i trädet.
   Detta krav är uppfyllt.

   Dessa funktioner är huvudfunktionaliteten i ett släktträdverktyg. Dessa funktioner finns beskrivna i avsnitten [6.2.3] och [6.2.4].

5. Applikationen måste ha funktioner för att skapa relationer mellan personer.
   Detta krav är uppfyllt.

   Skapa relation mellan två partner beskrivs i avsnittet [6.2.5]. Skapa andra relationer som syskonrelation, barn/föräldrar relation skapas automatiskt när en ny person läggs till i släktträdet.

6. Applikationen måste ha en inloggnings möjlighet för att kunna tilldela olika tjänster till användarna av systemet.
   Detta krav är uppfyllt.

   Systemet har en inloggnings möjlighet men den används inte för tillfälligt därför uppdragsgivare har inte bestämt sig om han ska använda detta systems inloggnings möjlighet, se avsnitt [6.2.1] för systemets egen inloggnings möjlighet. Systemets inloggningsfunktion kontrollerar inte typen av roller som t.ex. administratör, användare utan endast om användaren är korrekt inloggad.

7. Applikationen måste ha ett enkelt och användarvänligt gränssnitt.
   Detta krav är uppfyllt.
   Systemet har ett enkelt och användarvänligt gränssnitt. Hur användargränssnittet fungerar och ser ut, se kapitlet [6.2]. Genom att använda lätta färger i hela gränssnittet blev gränssnittet ett lättläst gränssnittet. I gränssnittet finns det ikoner för att göra gränssnittet mer användarevänligt som finns beskriven i ikonförteckning i bilaga [B] .

Kraven på data som ska lagras i släktträdet är beskriven i avsnittet [3.2.2]. Alla de olika datakraven är implementerade i systemet och finns i SQL tabeller som är förklarade i avsnittet [4.2.2]. De krav som gick under SHOULD var krav som skulle implementeras i mån av tid, och berördes inte under detta projekt då all tid lades på de krav som hade MUST.

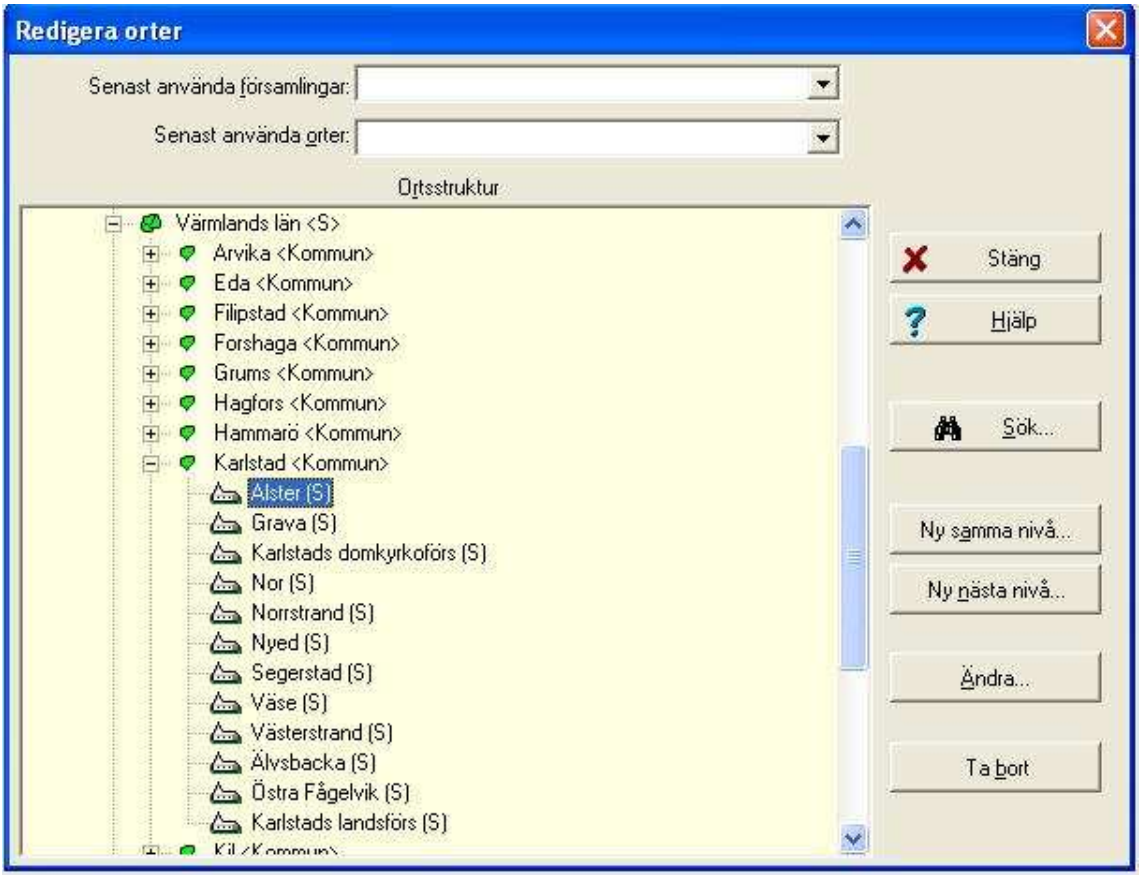
8.2.3 Alternativa metoder och lösningar

I detta projekt gjordes ett försök att göra hela programdesignen innan själva implementeringen gjordes. Detta visades sig inte vara den bästa metoden då det alltid är mycket svårt att på förhand veta vilka problem som kan uppstå eller vilka attribut och operationer som en klass kan behöva. Det hade varit bättre att först göra en liten del av designen för att sedan vid ett senare tillfälle gå tillbaka till designfasen och komplettera vid behov.

8.2.4 Rekommendationer för framtida funktionaliteter

I detta stycke beskrivs de framtida funktionaliteterna för detta projekt där ortskatalog, olika former av utskrift som antavla, stamtavla, tabell, träd och källkatalog är några av dem.

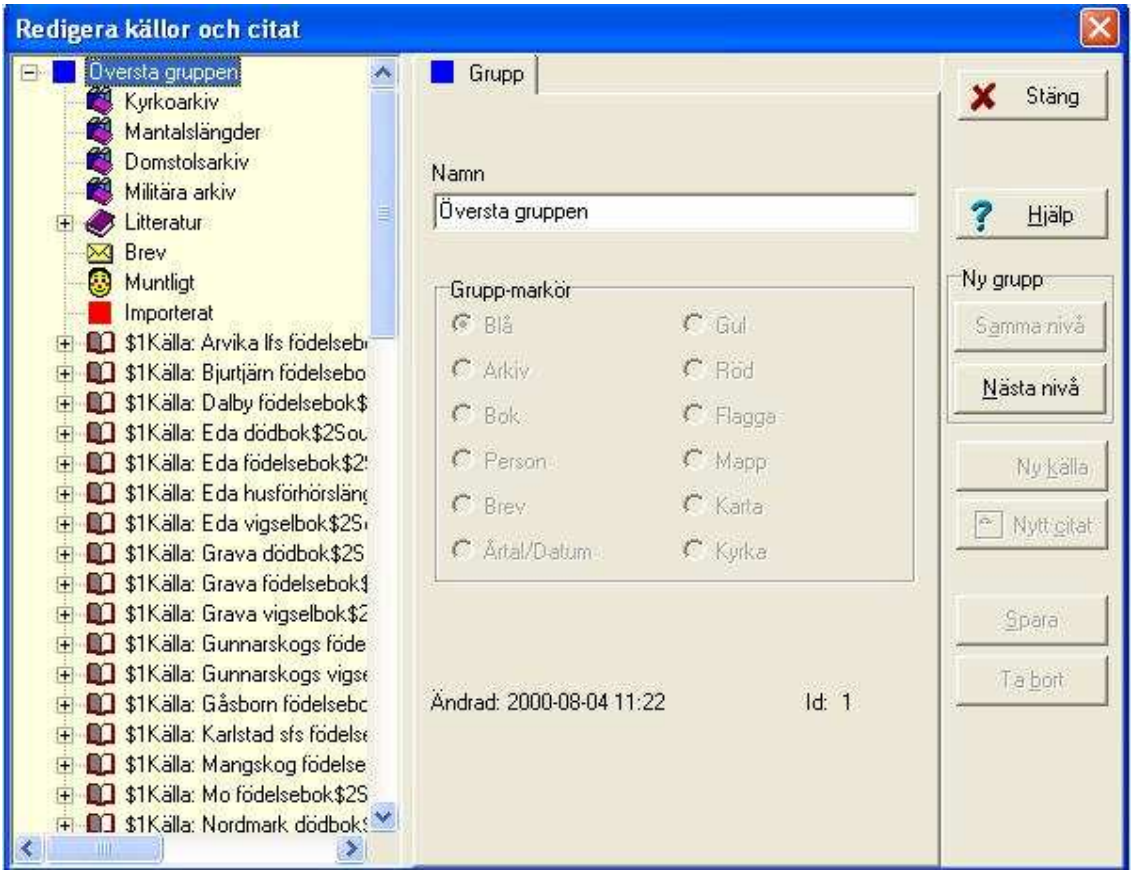
- Ortskatalog. Uppdelning mellan län, orter och församlingar. Genom en ortskatalog går det lättare att ange koordinater till en ort. Ett exempel på hur en ortskatalog kan se ut visas i Figur 29



Figur 29: DISGENs ortskatalog

Denna ortskatalog som Figur 29 visar finns i programmet DISGEN. I DISGENS ortskatalog är Sverige uppdelat i län, län är uppdelat i orter och sin tur är orter uppdelade i församlingar. Systemets framtida ortskatalog ska vara liknande som i DISGEN.

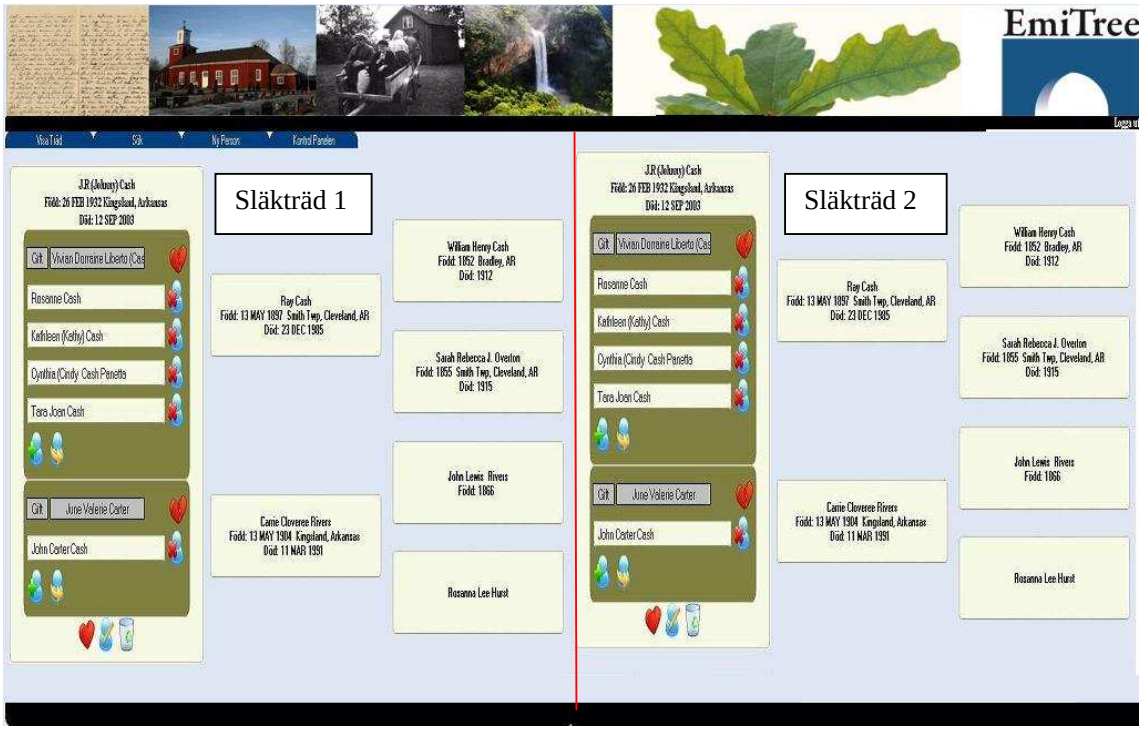
- Olika former av utskrifter som antavla, stamtavla, tabell och träd.
- Källkatalog. I en källkatalog delas källor in i en katalog för att lättare kunna söka upp dem. Källkatalog används för att normalisera källor. Figur 30 visar ett exempel på hur en källkatalog kan se ut.



Figur 30: DISGENs källkatalog

Figur 30 visar hur en källkatalog ser ut i programmet DISGEN. Källor är uppdelade i olika typer av grupper. Där brev och importerat är några av de grupperna.

- Utökning av sökparameter och där av synkronisering av sökformulär. För tillfälligt går det att söka på förnamn, efternamn, födelseår, födelseplats, dödsår, dödsplats. I framtiden ska det även gå att söka på fler parameter som t.ex. yrken, kön, fritext etc. Detta gör att det lättare går att hitta den sökta individen.
- Förbättring av importering och exporterings funktionaliteten.
- Dual-View. Med Dual-View menas användaren ska kunna köra två släktträdsvyer samtidigt och då kan denna lättare beräkna släktskap mellan två personer. Det går också att även lättare koppla ihop personer genom att dra och släppa. Ex. på hur Dual-View kan se ut visas i Figur 31.



Figur 31: Dual-View med två släktträdsvyer.

- Statistik funktion som t.ex. ange medelåldern på de döda i systemet, ange hur många som är snickare bland en församling etc.
- Flockar, kunna separera importerade släktträd samt funktion för framtagning av misstänkta dubbeletter. Separera importerade släktträd funktionaliteten är redan förbered. Systemet har en tabell Author som finns beskriven i avsnittet [4.2.2.7]. Denna tabell kan användas för att separera importerade släkträd genom att sätta en ägare på släktträdet.
- Multimedia som t.ex. foton, ljudklipp mm. Det lättaste sättet att addera multimedia till systemet är att samla all multimedia data i en MySQL tabell. Hur denna tabell kan se ut visas i [Tabell 10].

Tabell 10: Förslag på multimedia tabell

Kolumnnamn	Datatyp	Constraint	Referens
MID	Int	PK, NOT NULL	
FilName	Varchar	FK	
FilType	Varchar	FK	
Data	Binary	FK	

8.2.5 Avslutande kommentarer

Arbetet med projektet har varit intressant och lärorikt. Det intressant med detta projekt har varit att projektet har handlat mycket om släktforskning och speciellt webbprogrammering. Släktforskning är ett aktuellt ämne i Värmland idag på grund av att många av våra förfäder har emigrerat till USA, närmare 90 000 under tidsperioden 1851-1925 [27]. Det har varit intressant att jobba med webbprogrammering med ASP.NET, som är ett vanligt verktyg i arbetslivet idag. Det slutgiltiga resultatet blev ett webbaserat släktträdsverktyg som stödjer de enklaste släktträdsfunktionerna. Det skulle vara intressant om släktträdsverktyget kunde vidareutvecklas och kanske också publiceras på webben i framtiden.

Ett stort tack till Emigrantregistret och speciellt tack till Mathias Nilsson som gjorde detta arbete möjligt.

9 Referenser

[1] *ANSI*, *http://www.ansi.org/*

[2] Emigrantregistret, http://www.emigrantregistret.s.se

[3] DISGEN, http://www.dis.se/disgen.htm

[4] GEDCOM, http://homepages.rootsweb.com/~pmcbride/gedcom/55gctoc.htm

[5] MYSQL, http://www.mysql.com/

[6] ANSEL, http://en.wikipedia.org/wiki/ANSEL

[7] GEDCOM, kapitel 2, http://homepages.rootsweb.com/~pmcbride/gedcom/55gcch2.htm

[8] Shepard George 2006,"Microsoft ASP.NET 2.0 steg för steg", Pagina Förlags AB

[9] Ancestry.se, http://trees.ancestry.se/pt/pedigree.aspx?tid=3196760&pid=-1762644748, 2008-02-18

[10] Microsoft.NET Framework , http://en.wikipedia.org/wiki/.NET_Framework , 2008-02-18

[11] IFRAME, http://www.w3schools.com/tags/tag_iframe.asp, 2008-02-19

[12] Mössenböck Hanspeter, Beer Wolfgang. Birngruber Dietrich, Wöss Albrecht 2003, ".NET Application Development with C#, ADO.NET, ASP.NET and Web Servies",Addison Wesley

[13] Giovanni Boccaccio, http://en.wikipedia.org/wiki/Giovanni_Boccaccio, 2008-06-04

[14] Släktträd, http://en.wikipedia.org/wiki/Family_tree, 2008-03-04

[15] Md5, http://tools.ietf.org/html/rfc1321, 2008-06-04

[16] Indexering, http://www.databasteknik.se/webbkursen/prestanda/index.html, 2008-03-18

[17] Databas, http://www.databasteknik.se/webbkursen/databaser/index.html, 2008-03-18

[18] Joel Spolsky 2001, User Interface Design for Programmer, Apress

[19] Normalisering, http://www.databasteknik.se/webbkursen/normalisering/, 2008-03-31

[20] MoSCoW Prioritering, http://www.coleyconsulting.co.uk/moscow.htm, 2008-03-31

[21] Transaktioner, http://www.databasteknik.se/webbkursen/transaktioner/, 2008-03-31

[22] Larman Craig, "Applying UML and Patterns ", Tredje upplagan, Prentice Hall

[23] Unified Process, http://en.wikipedia.org/wiki/Unified_process, 2008-04-15

[24] GridView Confirm When Delete, http://www.gridviewguy.com/ArticleDetails.aspx?articleID=138, 2008-05-13

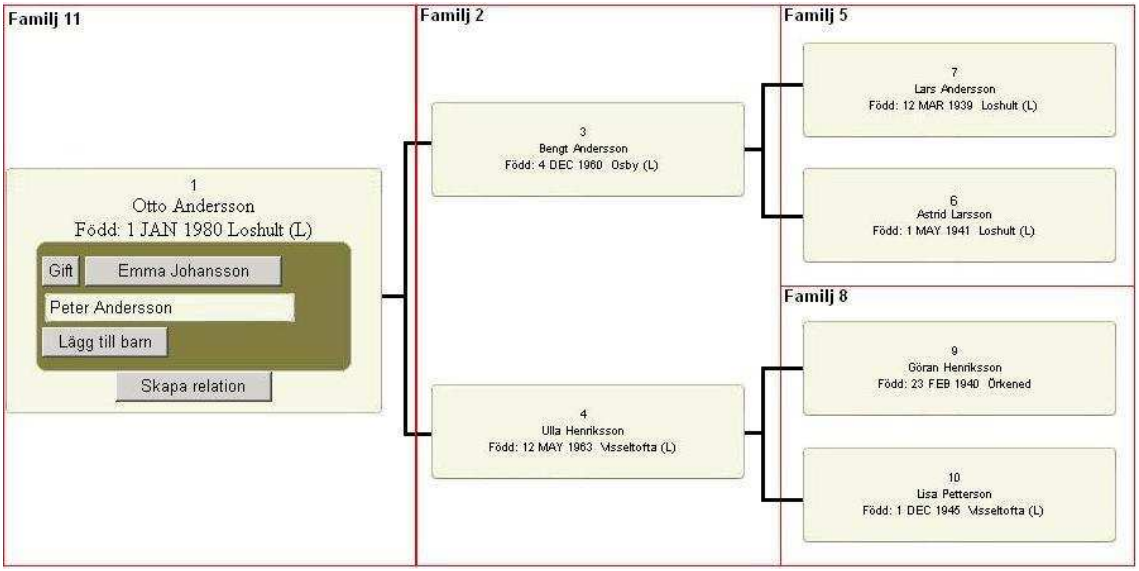
[25] GNU, http://sv.wikipedia.org/wiki/GNU-projektet, 2008-05-20

[26] JSP, http://en.wikipedia.org/wiki/JavaServer_Pages, 2008-05-20

[27] Ljungmark Lars, Den stora utvandringen, Sveriges Radio (923), 1965

A GEDCOM

A.1 GEDCOM fil med tre generationer



Figur 32: Illustration av textfilen i trädform.

Nyckelords definition

INDI := INDI anger starten på en individ, följt av individinformation.

FAM := FAM anger starten på en familj, följt av familjinformation.

Individinformation kan innehålla följande taggar:

SEX := SEX anger könet på en individ. F=Female(Kvinna) och M=Male(Man)

NAME := NAME utger namnet på en individ. Ex) 1 NAME Peter/Andersson

BIRT := Anger att det kommer födelseinformation.

DEAT := Anger att det kommer dödsinformation.

FAMS := Identifierar att en individ är make eller maka i en familj.

FAMC := Identifierar att en individ är barn i en familj.

PLACE := Anger en plats av någon typ som t.ex. födelseplats, dödsplats, etc.

CHAN := Information om en ändring.

DATE := Anger datuminformation.

TIME := Anger tidinformation.

Familjinformation kan innehålla följande taggar:

CHIL := Identifierar ett barn som ingår i en familj.

HUSB := Anger make i en viss familj.

WIFE := Anger maka i en viss familj.

MARR := Information om en relation. Exempel på relationer är gift, sambo, utom äktenskap och före äktenskap.

PLACE := Anger en plats av någon typ som t.ex. födelseplats, dödsplats, etc.

CHAN := Information om en ändring.

DATE := Anger datuminformation

TIME := Anger tidinformation.

Textfilen Gedcom.ged

För att ge läsaren bättre läsbarhet har vissa kommentarer lagts till samt att nya rader har gjorts för varje ny post, och är inget som finns i originalfilen. Figur 32 illustrerar hur denna fil reppresenteras som släktträd.

```
0 HEAD                    /* Visar starten på GEDCOM filen som börjar med en header.*/
1 SOUR DISGEN
2 VERS 8.1d
2 CORP Föreningen DIS - Datorhjälp i släktforskningen
3 ADDR Gamla Linköping
4 CONT S-582 46 Linköping
4 CONT Sverige
4 CONT Tfn: 013-14 90 43
4 CONT E-mail: dis@dis.se
4 CONT Hemsida: www.dis.se
1 DATE 18 FEB 2008
2 TIME 16:09:01
1 SUBM @XREFSUBM@
1 GEDC
2 VERS 5.5            /* Version nummer på GEDCOM: 5.5 */
2 FORM LINEAGE-LINKED
1 CHAR ANSI      /* Textformatet på GEDCOM filen är: ANSI */
0 @XREFSUBM@ SUBM
1 NAME (not supplied)

0 @1@ INDI   /* Individ med ID-nummer 1. */
1 SEX M
 1 NAME Otto/Andersson/
1 BIRT
2 DATE 1 JAN 1980
2 PLAC Loshult (L)
1 FAMC @2@
1 FAMS @11@
```

1 CHAN
2 DATE 18 FEB 2008
3 TIME 16:04:00

0 @2@ FAM /* Familj med ID-nummer 2.
1 CHIL @1@
1 HUSB @3@ 1 WIFE @4@
1 CHAN
2 DATE 18 FEB 2008
3 TIME 16:02:00

0 @3@ INDI */ Individ med ID-nummer 3. */
1 SEX M
1 NAME Bengt/Andersson/
1 BIRT
2 DATE 4 DEC 1960
2 PLAC Osby (L)
1 FAMS @2@
1 FAMC @5@
1 CHAN
2 DATE 18 FEB 2008
3 TIME 16:02:00

0 @4@ INDI /* Individ med ID-nummer 4. */
1 SEX F
1 NAME Ulla/Henriksson/
1 BIRT
2 DATE 12 MAY 1963
2 PLAC Visseltofta (L)
1 FAMS @2@
1 FAMC @8@
1 CHAN
2 DATE 18 FEB 2008
3 TIME 16:03:00

0 @5@ FAM /* Familj med ID-nummer 5. */
1 CHIL @3@
1 HUSB @6@
1 WIFE @7@
1 CHAN
2 DATE 18 FEB 2008
3 TIME 16:03:00

0 @6@ INDI */ Individ med ID-nummer 6. */
1 SEX M
1 NAME Lars/Andersson/
1 BIRT
2 DATE 12 MAR 1939
2 PLAC Loshult (L)
1 FAMS @5@
1 CHAN
2 DATE 18 FEB 2008
3 TIME 16:02:00

0 @7@ INDI /* Individ med ID-nummer 7. */
1 SEX F
1 NAME Astrid/Larsson/
1 BIRT
2 DATE 1 MAY 1941
2 PLAC Loshult (L)
1 FAMS @5@
1 CHAN
2 DATE 18 FEB 2008
3 TIME 16:03:00

0 @8@ FAM /* Familj med ID-nummer 8. */
1 CHIL @4@ /
1 HUSB @9@

1 WIFE @10@ 1 CHAN

2 DATE 18 FEB 2008

3 TIME 16:03:00

0 @9@ INDI / * Individ med ID-nummer 9. */

1 SEX M

1 NAME Göran/Henriksson/

1 BIRT

2 DATE 23 FEB 1940

2 PLAC Örkened (L)

1 FAMS @8@

1 CHAN

2 DATE 18 FEB 2008

3 TIME 16:03:00

0 @10@ INDI /* Individ med ID-nummer 10. */

1 SEX F

1 NAME Lisa/Petterson/

1 BIRT

2 DATE 1 DEC 1945

2 PLAC Visseltofta (L)

1 FAMS @8@

1 CHAN

2 DATE 18 FEB 2008

3 TIME 16:03:00

0 @11@ FAM /* Familj med ID-nummer 11. */

1 MARR

2 DATE 24 JUN 2005

1 HUSB @1@

1 WIFE @12@

1 CHIL @13@

1 CHAN

2 DATE 18 FEB 2008

```
3 TIME 16:04:00

0 @12@ INDI  /* Individ med ID-nummer 12. */
1 SEX F
1 NAME Emma/Johansson/
1 BIRT
2 DATE 14 JUL 1982
2 PLAC Loshult (L)
1 FAMS @11@
1 CHAN
2 DATE 18 FEB 2008
3 TIME 16:04:00

0 @13@ INDI    /* Individ med ID-nummer 13. */
1 SEX M
1 NAME Peter/Andersson/
1 BIRT
2 DATE 29 APR 2003
2 PLAC Osby (L)
1 FAMC @11@
1 CHAN
2 DATE 18 FEB 2008
3 TIME 16:04:00
0 TRLR   /* Visar slutet på en GEDCOM fil */
```

B Ikonförteckning

Nedan ges en tabell över de olika ikoner som finns i systemet samt vad dessa betyder och innebär.

Tabell 11: Förteckning över ikoner som finns i systemet

	**Lägg till barn** Denna används när en användare vill lägga till ett barn i en familj.
	**Individinformation** När användaren trycker på denna visas information om den valda individen. Möjligheten att uppdatera denna information ges också.
	**Koppla loss barn** Genom att trycka på denna kopplas det valda barnet bort från den familj som den tillhör.
	**Koppla person** Denna knapp används då en person skall kopplas till en familj, antingen som barn eller partner och redan finns i databasen. Användaren ges då möjligheten att söka fram denna individ och sedan koppla denna som barn eller partner.
	**Koppla loss relation** Denna knapp kopplar loss en relation från systemet. Detta kan inte utföras om relationen har barn. Då måste dessa först kopplas bort.
	**Skapa relation** Denna knapp ger användaren möjlighet att skapa en ny relation för den valda individen. Detta möjliggör att sedan koppla en partner samt barn till individen.
	**Ta bort individ** Denna knapp tar bort individen från släktträdet. Kan inte utföras om det finns någon familj kopplad till denna. Familjen måste först kopplas bort.

C Bugglista

- Bugg 1: Varningsruta vid koppla loss barn fungerar inte korrekt. Problemet inträffar när användaren har tryckt på knappen koppla loss barn som finns i [Figur 25]. Ett sätt att lösa buggen är att skapa en button eller en linkbutton i ett template fält i en gridview komponent. Genom att använda button eller linkbutton går det att anropa ett javascript som i sin tur skapar en varningsruta. Lösningen beskrivs på webbsidan [24].
- Bugg 2: Exportering fungerar inte korrekt. Problemet med exportering är att användaren inte kan lägga exportfilen i vilken mapp som helst utan exportfilen måste läggas på en katalog som ligger på servern. Ett sätt att lösa detta problem är att systemet skapar en temporär fil på servern som användaren laddar ner till en specifik katalog som användare har valt. Efter användare har laddat ner filen tas den temporära filen bort på servern.