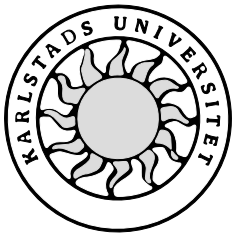


Datavetenskap

Leven Roumenov

Johan Nilsson

Utveckling av ett frånvarorapporteringssystem

Examensarbete, C-nivå

2005:10

Utveckling av ett frånvarorapporteringssystem

Leven Roumenov

Johan Nilsson

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.

Leven Roumenov

Johan Nilsson

Godkänd, 2005-06-02

Handledare: Katarina Asplund

Examinator: Donald F. Ross

Sammanfattning

Vi har utvecklat ett frånvarorapporteringssystem på uppdrag av Nobelgymnasiet i Karlstad. Målet med vårt arbete var att utveckla ett system som var lätt att använda för lärarna och som inte kostar pengar för skolan. Systemet ska också vara oberoende av installation på klientens dator. Resultatet av vårt arbete är ett fungerande frånvarorapporteringssystem som kan användas på skolan. Vi har använt MySQL för att lagra information och PHP för att skapa webbgränssnittet. Huvudfunktionerna är att registrera elevers frånvaro där lärarna bara kan registrera frånvaro på sina egna elever och visa rapporter på frånvaro-statistik över samtliga elever. Systemet kommer att utvärderas under hösten 2005 och eventuellt sättas i funktion till våren 2006.

Development of an absence reporting system

Abstract

We have developed an absence reporting system on behalf of Nobelgymnasiet in Karlstad. The purpose of our project was to develop a system that would be easy to use for the teachers and that was free of charge for the school. The system should also be independent of any installation on the client's computer. The result of our project is a working absence reporting system that could be used on the school. We used MySQL to store the information and PHP to create the web interface. The main functions are to register the pupils' absence where teachers can only register absence of their own pupils and to show reports on the absence statistics. The system is going to be evaluated during the autumn 2005 and probably be put in to use in spring 2006.

Innehållsförteckning

Figurförteckning

Tabellförteckning

1 Inledning

Den här uppsatsen beskriver arbetet med att skapa ett frånvarorapporteringssystem som är utfört för lärarna på Nobelgymnasiet i Karlstad. Vi har båda två gått på Nobelgymnasiet tidigare så vi känner uppdragsgivaren som är en av lärarna på Nobelgymnasiet och har bra kontakt med honom. Skolan har redan idag ett fungerande frånvarosystem, men uppdragsgivaren tycker att det nuvarande systemet har vissa nackdelar. Dessa nackdelar är att det är svårt att navigera i och det är svårt att ta fram rapporter. Det kostar även skolan pengar, eftersom skolan måste köpa licenser för att kunna använda systemet. För att vi skulle kunna klara av att lösa dessa nackdelar var vi tvungna att skapa ett helt nytt system från början, eftersom det nuvarande systemet är skapat av ett annat företag och systemet har inte släppts ut som öppen källkod. Vårt system som vi har utvecklat är ett fungerande system, men det är inte ännu satt i bruk på skolan. Systemet kommer att utvärderas på en avdelning under hösten 2005 och eventuellt sättas i funktion till våren 2006.

1.1 Syfte

Syftet med detta projekt var att skapa ett frånvarorapporteringssystem, som lärarna var med och påverkade, så att systemet blev som lärarna ville ha det. Frånvarorapporteringssystemet kostar inte skolan några pengar att använda utöver maskinvaran som krävs. Lärarna kan registrera frånvaro för eleverna på skolan och skapa rapporter med statistik på hur stor frånvaro en enskild elev har under ett tidsintervall. Frånvarorapporteringssystemet är oberoende av installation på klientens dator, det är webbaserat och enkelt att använda.

1.2 Disposition av uppsats

I kapitel 2 diskuteras framtagningen av kravspecifikationen, vilken har tagits fram och framställts i samarbete med uppdragsgivaren. I kapitel 3 beskrivs möjliga lösningar som kan användas för att utveckla frånvarosystemet. I kapitel 4 och 5 beskrivs designen för databasen och användargränssnittet för systemet. I kapitel 6 beskrivs den implementation som har använts för att skapa system. Examensarbetet avslutas med kapitel 7 där erfarenheter och slutsatser av arbetet är beskrivna.

2 Framtagning av kravspecifikation

Projektet handlar om att vi ska skapa ett frånvarorapporteringssystem, för lärare, som är lätt att använda och navigera i. Med detta menar vi att det inte ska finnas allt för mycket finesser och funktioner, vilket gör att användbarheten försämras. Frånvarosystemet ska vara ett webbaserat system för skoladministrationen. Tanken är att lärarna ska kunna dela på information utan att någon programvara behöver vara installerad på klientdatorn. Allt som ska behövas är en dator som är ansluten till Internet. För att nå programmet används en länk till skolans hemsida, där varje användare har ett eget inloggningskonto. Under projektets gång har vi besökt uppdragsgivaren ett flertal gånger och då har kravspecifikationen arbetats om varje gång. Den slutliga kravspecifikationen har vi valt att beskriva i detta kapitel. I avsnitt 2.1 beskrivs krav för hur gränssnittet ska se ut mer generellt. I avsnitt 2.2 behandlas användarnas krav på hur frånvaroregistreringen ska utföras mer ingående och i avsnitt 2.3 beskrivs användarens krav på hur rapporten ska visas.

2.1 Gränssnittets utseende

Startsidan ska innehålla en tydlig inloggningsruta. Användarnamn och lösenord ska skrivas in och lösenordet ska vara maskerat. Användaren loggar in genom att klicka på en knapp som förflyttar användaren automatiskt till startsidan. Startsidan ska innehålla viktig information och datum som gäller för lärarna. Frånvarorapporteringssystemet ska kunna uppdateras av en ansvarig lärare som har kunskaper i HTML.

Överst på alla sidor ska vem som är inloggad och dagens datum synas tydligt. Huvudlänkarna ska också vara synliga på alla sidorna, de ska vara placerade vertikalt och vara skrivna med en tydlig text som är lätt att läsa. Länkar på huvud-menyn som ska finnas med är *registrera*, *schema*, *rapporter*, *inställningar*, *hjälp* och *logga ut*. Eventuella undermenyer ska visas till vänster på sidan. Menyerna ska vara anpassade till vilken roll användaren har i systemet. Utöver detta ska sidan inte innehålla några animationer eller otydliga bilder.

2.2 Frånvaroregistrering

När användaren har valt att registrera elevfrånvaro ska användaren automatiskt hamna på den aktuella dag då användaren har sin första lektion. Där ska användaren kunna:

- Välja klass.
- Välja en enskild elev.
- Ange frånvarolängden som från början sätts lika med lektionslängden, men ska kunna ändras, tillexempel vid sen ankomst.
- Välja dag där det finns lektioner och då ska den första lektionen som inte är rapporterad under den aktuella dagen visas automatiskt.
- Planera frånvaro framåt i tiden för en enskild elev, tillexempel om elev har begärt ledighet.
- Ange anledning för frånvaron.
- Enbart registrera frånvaro för elever i det egna ämnet.
- Lägga till egna kommentarer.
- Se vem som har rapporterat frånvaron.
- Gå tillbaka och rapportera tidigare frånvaro.

2.3 Rapporter

Rapporter ska kunna skapas antingen i pdf-format eller i HTML-format. För varje rapport ska man kunna välja ett fast datum-intervall genom att skriva in start- och slutdatum. Startdatum ska från början sättas till första lektionen som eleven har och slutdatum ska sättas till dagens datum. Alternativt ska man kunna välja ett fast antal dagar bakåt i tiden som rapporten ska omfatta.

Det ska finnas ett antal grundrapporter färdiga att skriva ut. Man ska också kunna skapa helt egna rapporter där användaren själv väljer vilka fält som ska vara med, fältbredder och fältordning som ska användas och så vidare. Personliga rapporter blir bara tillgängliga för den användare som skapat rapporten. Urvalet sker på olika sätt beroende på om enskilda elever eller flera elever ska väljas. När man väljer att ta fram rapporten för en klass finns en speciell lista som tillåter val av enstaka elever.

3 Undersökning av möjliga lösningar

Från början visste inte vi vilka språk och databashanterare som skulle passa bra för vårt arbete. Därför valde vi att undersöka några olika typer av språk och databashanteraren MySQL. I avsnitt 3.1 kommer vi att göra en undersökning av programspråk och beskriva dem lite utförligare. I avsnitt 3.2 kommer vi att beskriva lite mer om databashanteraren MySQL. I avsnitt 3.3 beskriver vi vilket språk och databashanterare som vi kommer att använda oss av och varför vi valde att använda just det språket och den databashanteraren.

3.1 Undersökning av programspråk

Vi kommer i det här avsnittet diskutera eventuella lösningar till vår uppgift. Vi kommer också att diskutera fördelar och eventuella nackdelar med de olika lösningarna. I detta avsnitt går vi igenom de språk som vi har tittat närmare på.

3.1.1 PHP

PHP [6, 7] är ett serverbaserat flerplattforms-skriptspråk som genererar HTML-sidor till klienter. Detta gör att man får mer kontroll över vad som ska visas för användaren och HTML-sidorna blir inte så statiska, utan användaren kan interagera mer med innehållet och få mer personligt anpassade sidor. Vi beskriver först PHPs historia i avsnitt 3.1.1.1. I avsnitt 3.1.1.2 beskrivs PHP lite allmänt och avslutas med några fördelar och nackdelar med PHP i avsnitt 3.1.1.3.

3.1.1.1 Historia

PHP kommer från en äldre produkt som heter PHP/FI (Personal Home Page/Forms Interpreter). PHP/FI utvecklades av Rasmus Lerdorf under 1995. Han började med att sätta ihop några Perl-skript [23] för att se vilka som besökte hans personliga hemsida. Detta gjorde han mest för att imponera på sina besökare. De Perl-skript som han hade skapat döpte han till PHP. När mer funktionalitet krävdes skrev Rasmus en större C-implementation, som gjorde det möjligt att kommunicera med databaser och att göra små webbapplikationer. Rasmus valde att släppa källkoden för PHP/FI så att andra kunde använda PHP och vara med och utveckla det.

Under 1997 började Rasmus med att implementera en ny version av PHP/FI. PHP/FI hade då några tusen entusiaster runt om i världen och användes på mer än 50 000 domäner. Det motsvarade då cirka 1 % av alla domäner på Internet. Det var fortfarande Rasmus som höll i utvecklingen av PHP, men också ett fåtal andra personer bidrog men kodsnuttar till projektet. PHP/FI 2.0 släpptes officiellt i november 1997, efter att ha funnits i en beta-version en längre tid.

I slutet av 1997 släppte Andi Gutmans och Zeev Suraski PHP 3.0 alfa. Orsaken var att de hade hittat flera brister i PHP/FI när de utvecklade en E-handelsapplikation vid ett universitets-projekt. Detta ledde till att de gick ihop med Rasmus och förklarade att PHP 3.0 skulle bli efterföljaren till PHP/FI 2.0. Betydelsen av bokstäverna PHP byttes då också till Hypertext Preprocessor vid en gemensam omröstning bland utvecklarna. Den största styrkan med PHP 3 var att det gick lätt att lägga till nya funktioner, vilket har gjort att många användare har anslutit sig och bidragit med extra funktionalitet till PHP. Zeev Suraski har tillexempel börjat inkludera stöd för objekt-orienterad syntax. PHP 3 stöder arv och tillåter klasser att ha funktioner och egenskaper men inte så mycket mer. PHP 3 släpptes officiellt i juni 1998, efter nio månaders testande.

Gutmans & Suraski fortsatte att utveckla PHP under vintern 1998, deras mål var att förbättra kapaciteten och att öka modulariteten. De rörde dock knappt den objektorienterade modellen i PHP4, det gjordes inget mer på den förrän i PHP 5. Det var möjligt att hantera flera olika typer av databaser och APIs i PHP 3, men det var inte designat för att göra detta effektivt. När de släppte PHP 4 i maj 2000, hade de arbetat mycket med att göra PHP mer effektivt. Dessutom hade de inkluderat stöd för flera webbservrar, för HTTP-sessioner och infört ett mer säkert sätt att hantera inmatningar från användare. Målet med PHP 5 var att göra sig av med alla svagheter i språket och se till att det stannade bland de ledande webbspråken. I PHP 5 [14] har det satsats mest på utvecklingen av det objektorienterade paradigmet, men även minneshanteringen och stödet för flera trådar har förbättrats. Dessutom har motorn bytts ut och ersatts med Zend motorn 2.0 [8], Windows 95 stöds inte längre av PHP på grund av att det inte stödjer all funktionalitet som PHP använder sig av.

3.1.1.2 Allmänt

PHP är ett skriptspråk som bäddas in i HTML-sidor och tolkas av en webbserver innan sidorna skickas till en webbläsare. Det resulterar i att användaren ser PHP som vanliga HTML-sidor, men ibland kan användaren se på filändelsen, att det är en PHP-sida. Vanligtvis har PHP-sidor ändelserna .php, .php3 eller .phtml. Det går emellertid bra att använda vilken filändelse man vill. Idag används PHP av mer än etthundratusen webbutvecklare och finns på några miljoner webb-sajter, vilket motsvarar cirka 20 % av alla domäner på Internet.

3.1.1.3 Fördelar och nackdelar med PHP

PHP har många fördelar som gör det till ett intressant alternativ.
Några av fördelarna som kan nämnas är följande:

- ***Snabbt:*** PHP är väldigt snabbt i körning, speciellt när det är kompilerat med en Apache modul på en UNIX-maskin. Men eftersom PHP-koden måste tolkas, går det lite långsammare än om koden hade varit lagrad som maskinkod.

- ***Open source:*** Med varje utgåva av PHP finns komplett källkod inkluderad. Detta är bra om man vill kompilera sin egen PHP-installation med bara det nödvändigaste. Det är också bra om en säkerhetslucka upptäcks eftersom det går att åtgärda detta utan att behöva vänta på en officiell uppdatering.

- ***Fungerar bra med andra verktyg:*** PHP gör det lätt att kommunicera med andra program och protokoll. Det går att kommunicera med över 15 av de mest populära databaserna. Det finns stöd för protokollen POP3, IMAP och LDAP. Dessutom finns stöd för att kommunicera med distribuerade objekt som COM och CORBA.

- ***Har ett eget samfund:*** PHP utvecklas av personer från hela värden. Dessa personer hjälper också till med att svara på frågor och ger support till användare dygnet runt sju dagar i veckan. Dessutom gör de detta gratis vilket är en stor fördel mot andra alternativ där supporten kan vara väldigt kostsam.

En nackdel som kan nämnas är följande:

- **Särskiljning av variabler*:*** $-prefixet framför alla variabler är irriterande och störande, speciellt för nybörjarprogrammeraren (ett av de vanligare felen). Dessutom är det ganska onödigt.

3.1.2 ASP

ASP [9] är väldigt likt PHP, man kan säga att det är Microsofts svar på PHP. Den stora skillnaden mellan ASP och PHP är att ASP kostar pengar och bara går att använda med Microsofts egen webbserver. Vi beskriver först ASPs historia i avsnitt 3.1.1.1. I avsnitt 3.1.1.2 beskrivs ASP lite allmänt och så avslutas med några fördelar och nackdelar med ASP i avsnitt 3.1.1.3.

3.1.2.1 Historia

Det började [24] 1995, när Microsoft insåg att de låg efter med utvecklingen på Internet. Innan 1995 hade Microsoft koncentrerat sig mest på utveckling av tidigare teknologier. I början av 1996 släpptes *Internet Information Server* (IIS) till allmänheten. Då var IIS helt gratis att ladda ned. I slutet av 1996 släpptes beta versionen av IIS 3 som en uppgradering till IIS 2. Den största skillnaden med IIS 3 var att Active Server Pages (ASP) var inkluderat. I augusti år 1997 släpptes ASP 2 med IIS 4. Från åren 1998 till 2000 lades extra funktionalitet till ASP och då valde Microsoft att bara släppa ut små uppdateringar av ASP 2 istället för att släppa ut ASP 3. I och med introduktionen av IIS 5 blev ASP 3 tillgänglig för allmänheten.

3.1.2.2 Allmänt

ASP står för ”Active Server Pages” som på svenska kan översättas till ”aktiva sidor på servern”. Eftersom all kod körs på servern, medverkar detta till att man är oberoende av vilken webbläsare användaren kör, så länge webbläsaren stödjer HTML.

ASP kan användas till mycket och en del av de saker som ASP tillåter dig göra är att:

- Skapa dynamiska sidor som därefter returnerar gensvar (feedback) till användaren.
- Hämta och lagra information i en databas genom webbsidorna.
- Uppdatera innehållet på en webbsida utan att ändra något i HTML-koden
- Skapa webbsidor som bara blir användbara av en användare, genom att man lagrar information om den användaren och den informationen bara kan ses och användas av den personen.

3.1.2.3 Fördelar och nackdelar med ASP

ASP har flera fördelar [2] och några av dessa fördelar är att:

- Det går snabbt att komma igång med ASP och hjälpfilerna är väldigt bra skrivna.

- ASP är hårt integrerat med Windows NT och detta medför att sammankopplingen av program och komponenter blir enkel.
- Det är enkelt att koppla in egna klasser som är skrivna i Visual Basic.

ASP har även några nackdelar [2] och en del av dessa nackdelar är att:

- ASP fungerar bara på Microsoft Internet Information Server (IIS) [19].
- Om man kombinerar ASP med Visual Basic får man ett dåligt sätt att kommunicera med distribuerade objekt via CORBA [20].
- Kostar pengar att skaffa.

3.1.3 Java Servlets

Java Servlets [1] är java-kod som exekverar på server-sidan i en server-applikation, där en servlet svarar på anrop från klienter. Servlets är inte bundna till något specifikt protokoll, men vanligast är att man använder HTTP-protokollet. Ordet Servlet refererar ofta till en HTTP-servlet. Vi kommer först att beskriva historien om Java Servlets i avsnitt 3.1.3.1. I avsnitt 3.1.3.2 beskrivs Java Servlets allmänt och avslutas med fördelar och nackdelar med Java Servlet i avsnitt 3.1.3.3.

3.1.3.1 Historia

Java-språket som utvecklades av Sun 1991 [2, 3] var ursprungligen tänkt att användas till hemelektronik. 1993 började man utveckla Java för webben efter att den första webbläsaren Mosaic hade släppts. Då visade det sig att Java var ett perfekt språk för webben eftersom det var plattforms-oberoende och transportabelt. Dessutom är Javaprogram oftast väldigt små och kompakta, vilket leder till att de går snabbt att skicka över nätet.

Det var dock först 1996 som Java fick sitt stora genombrott, då det blev ett populärt språk för att skapa interaktiva webbsidor med hjälp av Java applets. Java applets är program som är direkt inbäddade i webbsidan. När webbläsaren hämtar en sida som innehåller en applet hämtas den hem och körs lokalt på klienten. Men allt eftersom java-applikationerna blev större, krävdes mer bandbredd för att hämta applets från servern. Detta ledde till längre väntetider och dessutom krävdes det att webbläsaren var kompatibel med Java för att kunna köras. För att lösa de här problemen började man flytta java till server-sidan. När koden körs på server-sidan undgår man kompabilitets-problem med webbläsaren och laddningstiderna för klienten blir kortare [4]. Sun var först med att utveckla en webbserver helt skriven i Java.

Denna hade en intressant komponentarkitektur som nu går under beteckningen servlets. Andra leverantörer följde efter och nu finns det flera webbservrar som klarar av att köra servlets. För de som inte kan köra servlets finns det tredjepartsprogram till de flesta webbservrar som gör det möjligt att köra servlets i nästan vilken miljö som helst.

3.1.3.2 Allmänt

En servlet körs helt och hållet inne i en Java Virtual Machine på servern, precis som java applets gör på en klient. Något som dock skiljer en servlet från en java-applet är att en servlet inte har något grafiskt gränssnitt. Därför kallas ibland servlets för applets utan ansikte. Servlets har utökats med ett extra klassbibliotek för att hantera kommunikation med webbläsare. När en begäran har kommit in till en java-servlet kan man i stort sett göra vad som helst med den. Tillexempel så kan man koppla upp sig till en databas via JDBC [21] och göra sökningar, eller så kan man anropa COBRA-komponenter på andra maskiner i nätverket. Servlets kan också användas som en inkörsport till ett större webbaserat system [2].

3.1.3.3 Fördelar och nackdelar med Java Servlets

Java Servlets har många fördelar [4, 5] som gör det till ett intressant alternativ. Några av fördelarna som kan nämnas är följande:

- ***Effektivt:*** En servlets initieringskod körs bara första gången webbservern startar servleten. När servleten är igång kan man göra anrop tills webbservern stängs av. Detta är mycket effektivt eftersom man inte behöver skapa ett nytt objekt varje gång som ett anrop kommer in. Servlets kan också hålla uppkopplingar till externa resurser, som tillexempel databaser. De kan även hålla information i minnet, vilket gör att sökningar går fortare, tillexempel kan de spara kategorierna på en webbsida i en vektor. Det gör att man inte behöver hämta data från en databas varje gång.

- ***Plattformsoberoende:*** Eftersom Servlets är skrivna i java som har ett väl definierat API är det lätt att flytta till ett annat operativsystem, utan att man behöver ändra i källkoden.

- ***Stabilt:*** Servlets har tillgång till hela Java Development Kit, vilket är ett väldigt kraftfullt utvecklingsverktyg. Java tillhandahåller också ett väl definierat undantagssystem för felhantering. Dessutom har det ett inbyggt system för att ta hand

om minnesläckage i systemet, en så kallad garbage collector. Eftersom det är en utökning av Java följer många klasser med från java som tillexempel nätverksstöd, filstöd och databas-tillgång.

- ***Säkerhet:*** Eftersom servleten körs på server-sidan ärver den säkerheten som ges av webbservern. Men en server kan även skydda sig mot servleten med hjälp av Java Security Manager. Servern kan köra servleten i ett övervakande läge att den inte gör skada på servern genom tillexempel dålig skriven kod, vilket kan skada filsystemet.

- ***Integrerat:*** Servleten är nära integrerad med webbservern. Detta betyder att servleten kan samarbeta med webbservern på ett bättre sätt, som tillexempel att utföra loggning och lägga till användare på servern.

- ***Flexibelt:*** Servlet API är designat så att det ska vara lätt att utöka om så skulle behövas. Idag innehåller API:t en klass för HTTP, men imorgon kan det finnas stöd för en ny typ eller kanske en utökning av HTTP har skett. Att Java är ett objektorienterat språk gör att det är väldigt lätt att göra utökningar.

Några av nackdelarna som kan nämnas är följande:

- ***Svårt:*** Java Servlets kan vara svårt att lära sig för nybörjare. Har en mycket högre inlärningströskel jämfört med andra skriptspråk.
- ***Dåligt stöd:*** Det är vanligt att webbhotell inte har stöd för Java Servlets.

3.2 Undersökning av databashanteraren MySQL

I detta kapitel diskuterar vi kring databaser och lite mer ingående om MySQL. Vi börjar med att beskriva vad databaser är och vad de används till i avsnitt 3.2.1. Vi koncentrerar oss på databashanteraren MySQL i resterande del av kapitlet. Vi var redan från början inställda på att använda MySQL eftersom vi har använt det tidigare och att det är gratis att använda. I avsnitt 3.2.2 börjar vi med att berätta MySQLs historia. I avsnitt 3.2.3 och 3.2.4 går vi igenom designen och arkitektur av MySQL.

3.2.1 Databaser

Databaser [11] har använts länge. Ett bibliotek är ett exempel på en databas som inte är elektronisk. Ett bibliotek har böcker och dokument som är organiserade i en viss ordning. När

du vill få tag på en bok eller annan data får du antingen fråga bibliotekarien eller leta i ett periodiskt index. Ett annat exempel på en databas kan vara dokumenten som innehåller information om de anställda på ett företag. Alltså är databaser en samling av data. För att få en bra databas måste databasen vara organiserad och lagrad enligt något system. Det finns många nackdelar med pappersbaserade databaser. Några nackdelar är det fysiska utrymmet som krävs för att lagra databasen och att det kan ta lång tid att hitta det man söker efter i databasen. Det kan också vara svårare att underhålla ett bibliotek, eftersom man måste lägga alla böcker på rätt plats igen. Detta leder till att det blir ganska kostsamt att ha pappersbaserade data. Därför använder man sig ofta istället av elektroniska databaser.

På 60-talet och 70-talet utvecklades *database management systems* (DBMS), eftersom man ville ha ett system som kunde hantera informationen på ett bättre sätt. På svenska förkortar man DBMS till DBHS och det betyder databashanteringssystem. Databashanterare är mjukvara som används för att hantera data i en databas. Några av databashanterarna som finns ute på marknaden är DB2, Informix, INGRES, InterBase, Microsoft Access, Microsoft SQL Server, MySQL, Oracle och många fler. MySQL var tidigt framme med att ta fram ett system som kunde hantera relationsdatabaser. Förkortningen för relationsdatabaser är RDBMS och det står för *relational database management systems*.

En relationsdatabas är en databas som organiserar data i tabeller och därefter skapar förhållanden emellan dessa tabeller. Med hjälp av dessa förhållanden kan man kombinera data från flera tabeller och få fram olika svar i form av tabeller.

Tabell 3-1 och 3-2 är två tabeller som ingår i en databas som heter ”fotboll Division 1”. Man ser att dessa två tabeller innehåller en kolumn som heter ”Lag #”. Detta skapar ett förhållande mellan spelare och lag. Genom att koppla ihop kolumnerna ”Lag #” från båda tabellerna kan man se att Daniel Karlsson spelar för laget Kils AIK. Man skulle även kunna ta fram alla spelare från laget Grums BK genom att skapa en fråga i databashanteraren.

Lag #	**Namn**	**Tränare**
1	Kils AIK	Peter Karlsson
2	Karlstad BK	Simon Eliasson
3	Grums BK	David Eriksson

Tabell 3-1 Lag

Namn	Position	Lag #
Daniel Karlsson	Forward	1
Peter Nilsson	Forward	3
Niklas Svensson	Center	2
Gunnar Bengtsson	Forward	2

Tabell 3-2 Spelare

3.2.2 MySQLs historia

Det började [11, 12] 1979, då utvecklade Michael Widenius ett databas-verktyg som kallades för UNIREG [22]. Michael Widenius kallades också för Monty och han utvecklade detta databas-verktyg åt företaget TcX. UNIREG har skrivits i flera olika språk och kan handskas med stora databaser. 1994 började TcX utveckla webbaserade applikationer och använde sig av UNIREG för detta. Det blev inte lyckat, därför började TcX undersöka mSQL [10], men mSQL 1.x stödde inte någon indexering vilket var viktigt enligt Monty. mSQL kallas också för MiniSQL och är ett relations-databashanteringssystem. Detta databashanteringssystem var byggt för att användas med små och medelstora databaser.

Monty kontaktade David Hughes som var grundaren av mSQL och frågade om han var intresserad av att koppla mSQL till UNIREG's B+ ISAM hanterare. Men Hughes ville inte det eftersom han skulle ha sin egen indexerings-infrastruktur i sin kommande version av mSQL. Då bestämde sig företaget TcX för att skapa sin egen databas-server som skulle ha UNIREG's indexerings-system. TcX skapade databas-servern som var uppbyggt över UNIREG och hade ett API som var ganska likt mSQLs API. Detta skulle underlätta för användarna att flytta från mSQL till TcX egna databas-server. 1995 hade TcX en databas-server som hette MySQL som levde upp till de ställda kraven och då försökte David Axmark övertala TcX så att de skulle släppa MySQL på Internet. Det är fortfarande ingen som vet vart namnet MySQL kommer ifrån. Tydligen hade företaget TcX förstavelsen ”My” över sina kataloger, bibliotek och verktyg och Montys dotter heter också My.

3.2.3 MySQLs design

När man började utveckla MySQL ville man att den skulle vara minst lika snabb som mSQL, men med en större mängd särdrag. Det TcX ville få ut av MySQL var att den skulle vara snabb, robust och enkel att använda. För att uppnå dessa egenskaper, beslutade TcX att göra

MySQL till ett multipeltrådat databasverktyg. Detta var en stor fördel för MySQL eftersom man då kan ha en separat tråd som tar hand om varje inkommande uppkoppling med en extra tråd som alltid kördes för att hantera uppkopplingarna. Flera klienter kan också samtidigt utföra läs-operationer utan att påverka varandra. Medan en tråd är i en tabell och ändrar, kommer alla andra trådar som vill ha tillgång till den tabellen få vänta tills tråden har exekverat klart. En annan fördel med multitrådning är att även om de använder samma processutrymme exekverar de individuellt. Detta leder till att ett system som har flera processorer, kan dela trådarna emellan alla processorer. MySQL har även stöd för många datatyper och SQL-funktioner.

MySQL fungerar på många plattformar. Den har APIer för C, C++, Java, PHP och många fler programspråk. MySQL stödjer både transaktionella och icke transaktionella lagringsverktyg och den kan ta hand om stora databaser. För att ta reda på alla funktioner som MySQL stödjer kan man besöka hemsidan *http://dev.mysql.com*. En viktig funktion som MySQL dock saknar är stöd för objektorienterade datatyper.

3.2.4 Arkitekturen av MySQL

MySQL är egentligen två program, eftersom MySQL använder sig av en klient/server arkitektur. Databasservern finns på en maskin där all data lagras. Klienter är program som kopplar upp sig mot databasservern. Klienter skapar frågor och begär att få informationen de vill ha av databasservern och databasservern skickar informationen till den klient som begäran kom ifrån. MySQL har flera klientprogram och de används för olika ändamål. Några av de vanligaste klienterna som används är:

- ***mysql:*** Denna klient kan användas för att ställa SQL-frågor mot en databas och även för att köra SQL-frågor som finns lagrade i en fil.
- ***mysqldump:*** Denna klient kan användas för att göra backup av databaser eller för att överföra data till en annan SQL-server.
- ***mysqlimport:*** Denna klient kan användas för att importera data i olika filformat till en MySQL tabell.
- ***mysqladmin:*** Denna klient kan användas för att hantera databasserver. Detta inkluderar skapande och borttagande av databaser.

3.3 Sammanfattning

Vi började först använda Java servlets eftersom det var gratis, men insåg senare efter undersökningarna av andra språk att PHP var enklare att använda. PHP har många fördelar som exempelvis att det är väl dokumenterat, gratis och det har ett bra stöd för MySQL. Vi hade lite funderingar på att använda ASP eftersom vi skulle göra vårt arbete i Windows-miljö, men eftersom ASP kostar pengar och fungerar bäst med Microsoft egna produkter, valde vi att inte använda det. Eftersom PHP är gratis och väldigt likt ASP, kändes PHP som det självklara alternativet för vårt arbete.

När det kom till valet av databashanterare hade vi bestämt oss för att använda MySQL redan från början av projektet. Det finns många databashanterare ute på marknaden men de flesta databashanterare kostar pengar. Men vi insåg, efter lite efterforskning att det är många som använder MySQL. Ett exempel på företag som använder MySQL är ThinkHost [18]. Eftersom MySQL är gratis, stabilt och en av de snabbaste SQL-serverna på marknaden insåg vi att MySQL skulle passa vår uppgift väldigt bra. Dessutom fungerar MySQL bra tillsammans med PHP vilket underlättar för oss.

4 Databasdesign

Vi har skapat ett webbaserat frånvarorapporteringssystem som använder sig av en databas. I detta kapitel kommer vi att behandla databasdesignen och i kapitel 5 kommer vi att beskriva gränssnittet. I avsnitt 4.1 börjar vi med att beskriva lite kort om databasen och hur vi kommer att använda den och i avsnitt 4.2 kommer entiteterna och relationerna mellan entiteterna i databasen *frånvaro* att beskrivas. Vi visar hur vi har gjort mappningen av E/R modellen till relationsdatamodellen i avsnitt 4.3 och i avsnitt 4.4 kommer vi att gå igenom hur en bra design fås på en databas med hjälp av normalisering. Vi avslutar kapitlet med en sammanfattning i avsnitt 4.5.

4.1 Introduktion

Vårt system ska huvudsakligen användas för att lägga in och uppdatera information i frånvaro-tabellen. Systemet kommer att hämta information ifrån alla tabellerna (förutom klass/lektion) för att skapa frånvarorapporterna. För att registrera nya elever, lärare och lektioner in i databasen används ett webbaserat gränssnitt. Databasen är designad för att fungera på en enskild skola.

4.2 E/R-diagram

I avsnitt 4.2.1 kommer de olika entiteterna att beskrivas och deras relationer till varandra beskrivs i avsnitt 4.2.2.

4.2.1 Entiteterna

Vi kommer att beskriva entiteterna som kan ses i *figur 4-1*.

- ***Elever:*** Denna entitet ska innehålla information om alla elever som går på skolan. Det som också ska vara med är vilken klass dessa elever går i. Attributet *Elev_Personnummer* är primärnyckeln för denna entitet.

- ***Frånvarande:*** Denna entitet ska innehålla information om elevernas frånvarande eller närvarande på lektionerna. Attributen *Personnummer* och *Lektions_Id* är främmandenycklar i entiteten *frånvarande*. Tillsammans blir dessa två nycklar en primärnyckel som är unik för denna entitet.

- ***Lektion:*** Denna entitet ska innehålla information om alla lektioner som finns på skolan. Det ska även finnas information om vilka lärare som har hand om lektionerna, när de börjar, sluttiderna för lektionerna och vilka dagar de går på året. Denna entitet har *Lärare_Id* som främmandenyckel och *Lektions_Id* är primärnyckeln för denna entitet.

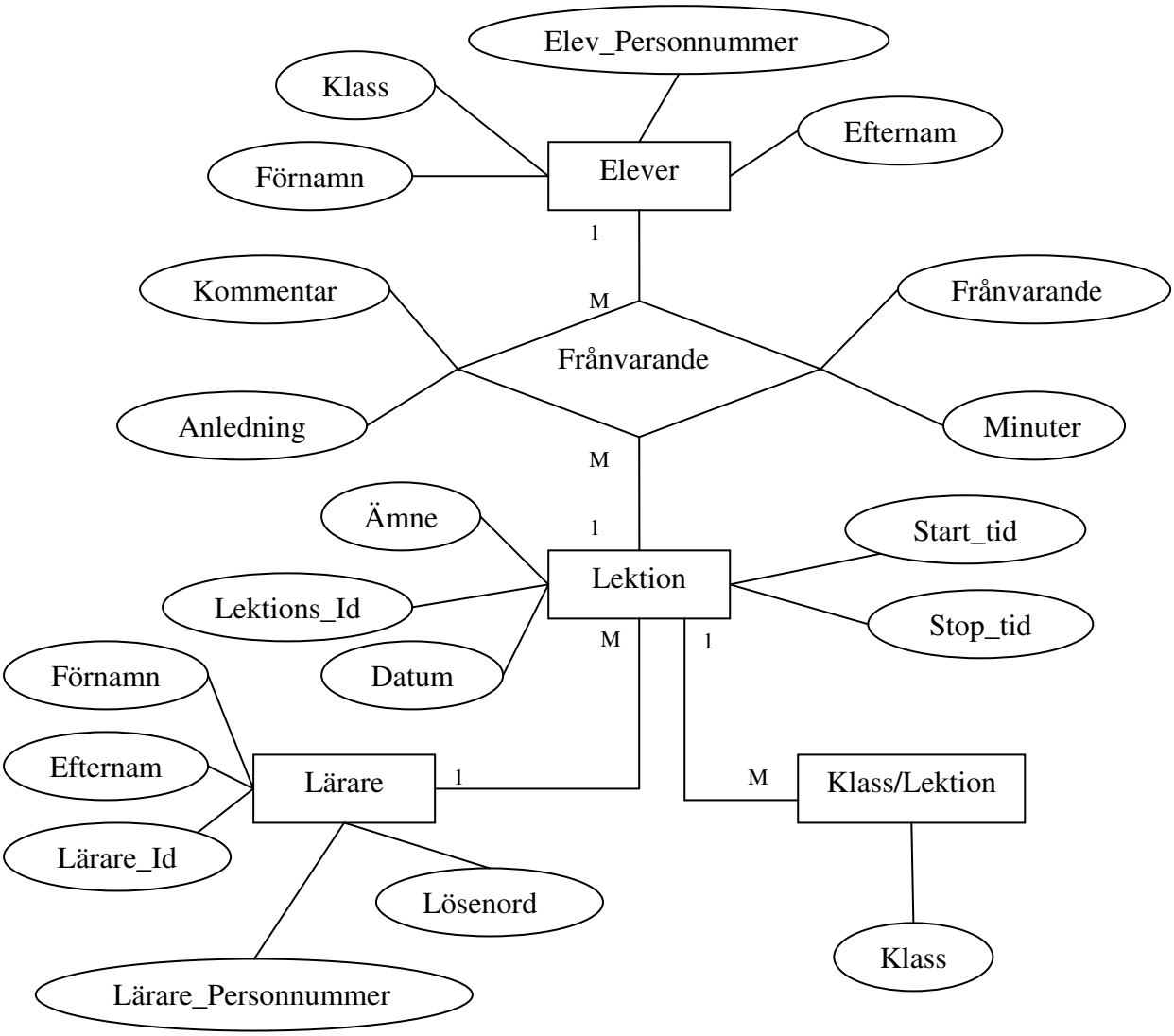
- ***Klass/Lektion:*** Denna entitet ska innehålla information om vilka lektioner en viss klass har. Denna entitet används som hjälp för att skapa entiteten *frånvarande*. Denna entitet har *Lektions_Id* som främmandenyckel och *Klass* är ett hjälpattribut som innehåller alla klasser som finns på skolan. Attributen *Klass* och *Lektions_Id* blir tillsammans en primärnyckel som är unik för denna entitet.

- ***Lärare:*** Denna entitet ska innehålla information om alla lärare på skolan. Attributen *Förnamn*, *Efternamn* och *Lärare_Personnummer* finns med i denna entitet. *Lärare_Id* och *Lösenord* ska också vara med i denna entitet och dessa två attribut kommer att användas för att logga in i frånvarosystemet. *Lärare_Id* är unikt och blir primärnyckel för denna entitet.

4.2.2 Relationerna mellan entiteterna

Mellan entiteterna *Lektion* och *Klass/Lektion* är det en 1:M relation, vilket betyder att en lektion kan ha flera klasser. Det betyder också att en klass bara kan ha en lektion under en viss tidpunkt.

Mellan entiteterna *Lärare* och *Lektion* är det en 1:M relation, vilket betyder att en lärare kan ha flera lektioner. Det betyder även att en lektion bara har en lärare.



Figur 4-1 E/R-diagram

4.3 Mappning av E/R-modellen till relationsdatamodellen

I tabellerna 4.3-1 till 4.3-5 finns mappningarna av E/R-modellen. För att få en utförligare beskrivning av relationsdatamodellen hänvisar vi er till Bilaga A.

Entitet	Attribut
Elever	Elev_Personnummer
	Klass
	Förnamn
	Efternamn

Tabell 4-1 Elever

Entitet	Attribut
Frånvarande	Minuter
	Kommentar
	Frånvarande
	Anledning
	Elev_Personnummer+
	Lektions_Id+

Tabell 4-2 Frånvarande

Entitet	Attribut
Lektion	Ämne
	Lektions_Id
	Datum
	Start_Tid
	Slut_Tid
	Lärare_Id+

Tabell 4-3 Lektion

Entitet	Attribut
Lärare	Förnamn
	Efternamn
	Lärare_Id
	Lärare_Personnummer
	Lösenord

Tabell 4-4 Lärare

Entitet	Attribut
Klass/Lektion	Klass
	Lektions_Id+

Tabell 4-5 Klass/Lektion

4.4 Normalisering

Normalisering [15] är en process som används för att få en bra design på databasen. Normalformerna är en mängd regler som beskriver vad man bör och inte bör göra i tabellstrukturen. Här nedan beskrivs de normalformer som har använts vid skapandet av *frånvaro*-databasen. Vi vill även säga att det finns högre normalformer än det vi har använt oss av. För att uppfylla en högre normalform, är ett krav att de underliggande normalformerna måste vara uppfyllda.

Första normalformen: En tabell måste innehålla atomära värden, det vill säga högst ett värde per cell i tabellen.

Andra normalformen: Första normalformen måste vara uppfylld samt varje attribut som inte är en del av primärnyckeln måste vara fullständigt beroende av primärnyckeln.

Tredje normalformen: Andra normalformen måste vara uppfyllt samt varje icke-nyckelattribut måste vara oberoende av något annat icke-nyckelattribut.

4.5 Sammanfattning

Vi har skapat en databas som befinner sig i tredje normalformen och som uppfyller våra behov för vårt nuvarande system. Det kan tänkas att ytterligare tabeller och egenskaper kan behövas lägga till om systemet ska användas på andra skolor. Databasen består av fem stycken tabeller som finns lagrade i en MySQL-databas. Databasen är oberoende resten av systemet, den kan därmed fungera med andra program eller system om så skulle önskas. Det vill säga, det ska inte vara några problem att skapa ett fristående program som kan skapa ett schema och lägga in det i databasen. I det nuvarande systemet är databasen inbakad i systemet, vilket gjorde att andra program inte kunde använda data för andra tillämpningar.

5 Användargränssnitt

I detta kapitel kommer vi att beskriva användargränssnittet i vårt frånvarosystem och funktionaliteten som finns i systemet. Vi har valt att beskriva användargränssnittet genom att beskriva de olika webbsidornas utseende och vilka funktioner som finns tillgängliga och hur de fungerar. I avsnitt 5.1 beskrivs inloggningssystemet och i avsnitt 5.2 beskrivs startsidan. I avsnitt 5.3 beskrivs hur registreringen av frånvaro för elever fungerar. I avsnitt 5.4 beskrivs skapandet av rapporter och i avsnitt 5.5 beskrivs framtida utvecklingar av frånvarosystemet och vi kommer även att nämna några andra funktioner som vi inte har implementerat ännu. Dessa funktioner har vi inte implementerat på grund av att tiden som vi hade på oss var ganska kort för att hinna implementera alla funktioner som fanns med i kravspecifikationen som vi fick från uppdragsgivaren.

5.1 Inloggning

Eftersom bara den lärare som är ansvarig för lektionen ska kunna rapportera frånvaro för sina egna elever, skapades ett inloggningsystem som gör att bara behöriga lärare kan rapportera frånvaro eller ändra frånvaro. Inloggningssidan som kan ses i figur 5-1 består av två textrutor, ena textrutan för användarnamn och den andra textrutan för lösenord. När lösenordet matas in maskeras det för att personer som står bredvid inte ska kunna se lösenordet som matas in. Under dessa textrutor finns det en knapp som man måste trycka på för att logga in i frånvarosystemet. Efter att man har tryckt in knappen görs en kontroll av användarnamnet och lösenordet, så att de överensstämmer med databasens användarnamn och lösenord. Ifall användarnamnet, lösenordet eller båda två inte stämmer överens med användarnamnet och lösenordet som finns lagrat i databasen, kommer det att visas en text över inloggningsrutan att ogiltigt användarnamn eller fel lösenord har matats in.

Figur 5-1 Inloggning

5.2 Startsida

När användaren har loggat in med korrekt användarnamn och lösenord kommer användaren till en sida som vi kallar för hem vilket kan ses i figur 5-2. Sidan består av två delar, en övre del och en undre del. Den övre delen följer med alla sidor och innehåller alla de viktigaste länkarna. I översta delens vänstersida finns det också namn på läraren som är inloggad och dagens datum. Det är härifrån användaren kommer att kunna navigera till de olika undersidorna i systemet. De länkar som finns är:

- **Hem:** Denna länk leder användaren till första sidan som innehåller information om viktig information och händelser på skolan. Vi kommer att gå igenom mer vad den innehåller i slutet av detta avsnitt.
- **Registrera:** Denna länk leder till sidan där användaren kan registrera frånvaron på. Vi kommer att gå igenom mer vad den här sidan innehåller i avsnitt 5.3.

- **Schema:** Denna länk leder till den sidan där användaren kan skapa schemat. Denna funktion fungerar men den är inte helt utvecklad ännu och för mer information om denna funktion kan man läsa i avsnitt 5.5.1.
- **Rapporter:** Denna länk leder till sidan där användaren kan använda för att ta fram rapporter över hur frånvarostatistiken ser ut för elever eller över en hel klass. Vi går igenom rapporterna mer i avsnitt 5.4.
- **Inställningar:** Denna länk leder till sidan där användaren kan byta lösenord som används för att logga in på frånvarosystemet. Denna funktion är ännu inte utvecklad i systemet.
- **Hjälp:** Denna länk leder till en sida där användaren kan få hjälp med hur frånvarosystemet fungerar. Denna sida innehåller ingen information ännu.
- **Logga ut:** Denna länk loggar ut användaren från frånvarosystemet.

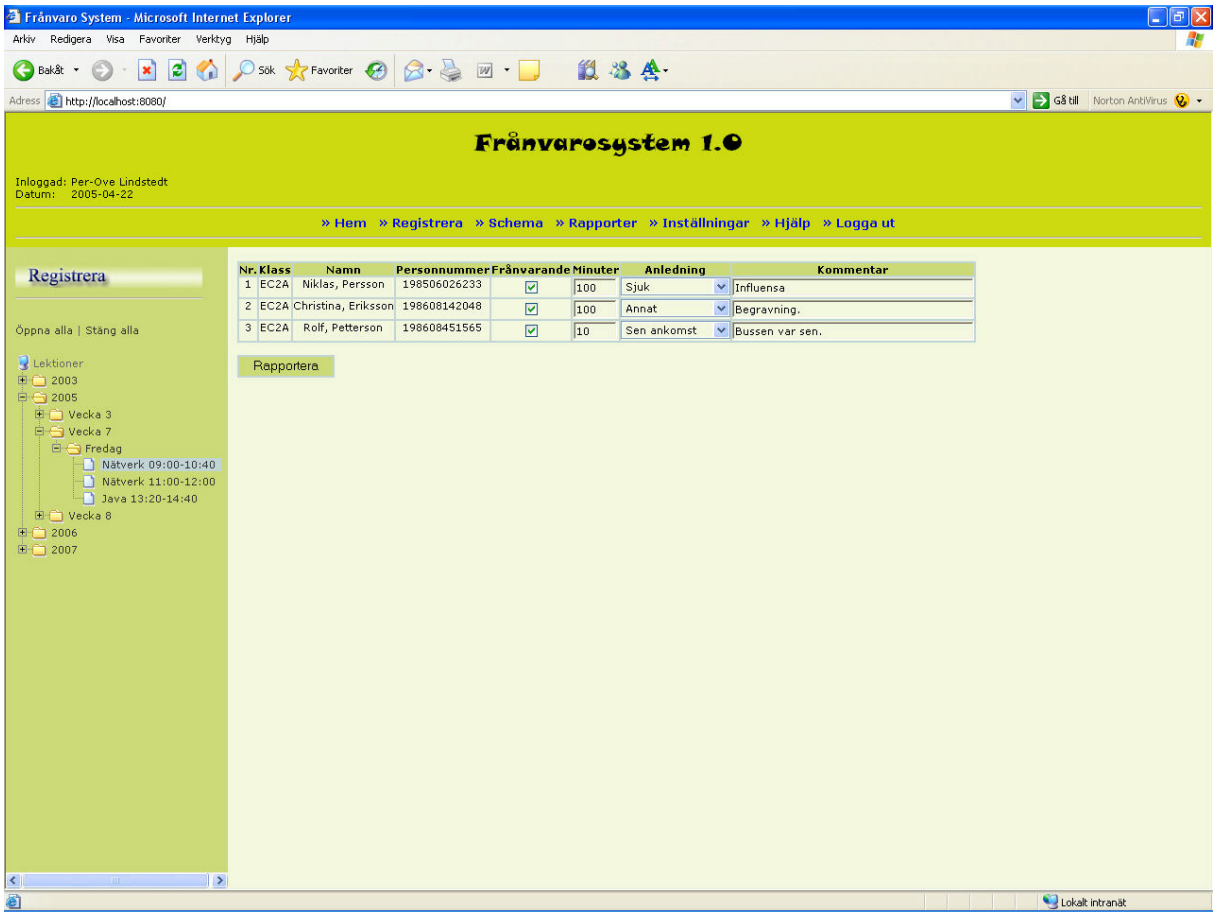
I den nedersta delen i figur 5-2 visas, efter att man har loggat in, den sidan som vi kallar för hem. Den kommer att innehålla viktig information och händelser som inträffar på skolan. Eventuellt kan man även lägga upp hur många lektioner som inte är rapporterade. Informationen om viktiga händelser kommer att vara lätt att uppdatera av en ansvarig på skolan som har lite kunskaper i HTML. Eventuellt skulle man kunna utveckla ett webbsystem för att lägga in information istället för att använda HTML. Eftersom det inte var något krav från uppdragsgivaren har vi dock valt att inte göra det.

Det är i den understa delen av sidan som information visas när man klickar på någon av länkarna som beskrevs tidigare i detta avsnitt. Vad de sidorna innehåller kommer vi att gå igenom i följande avsnitt.

Figur 5-2 Hem

5.3 Registrera frånvaro

I detta avsnitt kommer vi att beskriva hur registreringen av frånvaro går till och vi kommer även att beskriva utseendet (gränssnittet) av frånvaroregistreringen som finns i figur 5-3. Det är här som läraren registrerar frånvaron för eleverna.



Figur 5-3 Registrering

5.3.1 Utseende

Frånvaroregistreringssidan är indelad i två ytterligare delar som kan ses i figur 5-3. I den vänstra delen finns det en träd-meny som innehåller alla lektioner som läraren har, har haft och kommer att ha längre fram i tiden. Överst i träd-menyn finns det år som läraren har lektioner. Under året finns det undermenyer med alla veckor som läraren har lektioner. Under varje vecka finns det undermenyer med alla dagar som läraren har lektioner och under dagarna finns lektionerna som läraren har den dagen. När läraren har hittat den lektion som han/hon vill registrera frånvaro på, klickar han/hon på den länken. I denna länk står lektionens namn och vilken tid lektionen börjar och slutar.

Tabellen som finns i den högra ramen innehåller:

- **Klass:** Här står det vilken klass eleven går i.
- **Namn:** Här står elevens namn.
- **Personnummer:** Här står elevens personnummer.

- **Frånvarande:** Denna tabellruta innehåller en kryssruta som antingen kan vara kryssad eller inte kryssad. Om den är kryssad, då är eleven frånvarande från lektionen.
- **Minuter:** Denna tabellruta innehåller en textruta. I denna textruta skriver användaren antalet minuter som eleven är frånvarande från lektionen.
- **Anledning:** Denna tabellruta innehåller en rullgardinsmeny. I denna meny finns det olika anledningar för varför eleven kan vara frånvarande.
- **Kommentar:** Denna tabellruta innehåller en textruta. I denna textruta kan användaren/läraren skriva egna kommentarer.

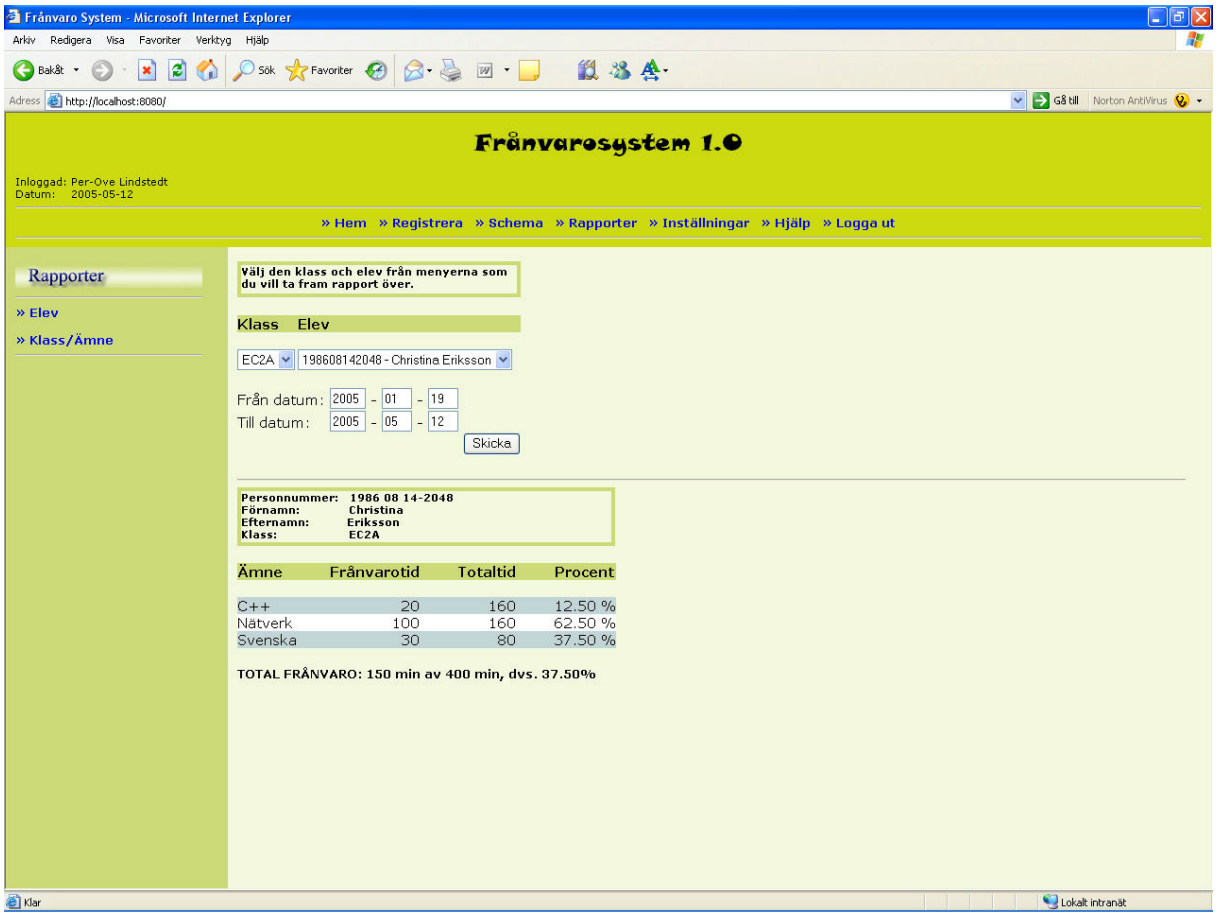
Under tabellen finns det en helt vanlig submit-knapp som heter rapportera som man klickar på för att bekräfta inmatningen.

5.3.2 Funktioner/funktionalitet

Träd-menyn som finns i figur 5-3 är ordnad i kronologisk ordning och den hämtar information från databasen. Träd-menyn är också unik för varje användare. Med det menas att ifall användaren ”Per” loggar in i systemet kommer bara lektionerna som denna användare har, synas i träd-menyn. Det finns även två extra funktioner för träd-menyn där man kan välja att öppna eller stänga alla undermenyer. Detta är praktiskt ifall användaren har öppnat många undermenyer och inte vill stänga alla menyerna manuellt. När användaren har valt lektionen som han/hon vill registrera frånvaro på visas en tabell med alla elever som går den lektionen. Efter att man har fyllt i formuläret måste man trycka på knappen rapportera för att uppdatera ändringarna i databasen. Därefter kommer man till en sida där man kan se om registreringen blev lyckad eller inte lyckad.

5.4 Ta fram rapporter

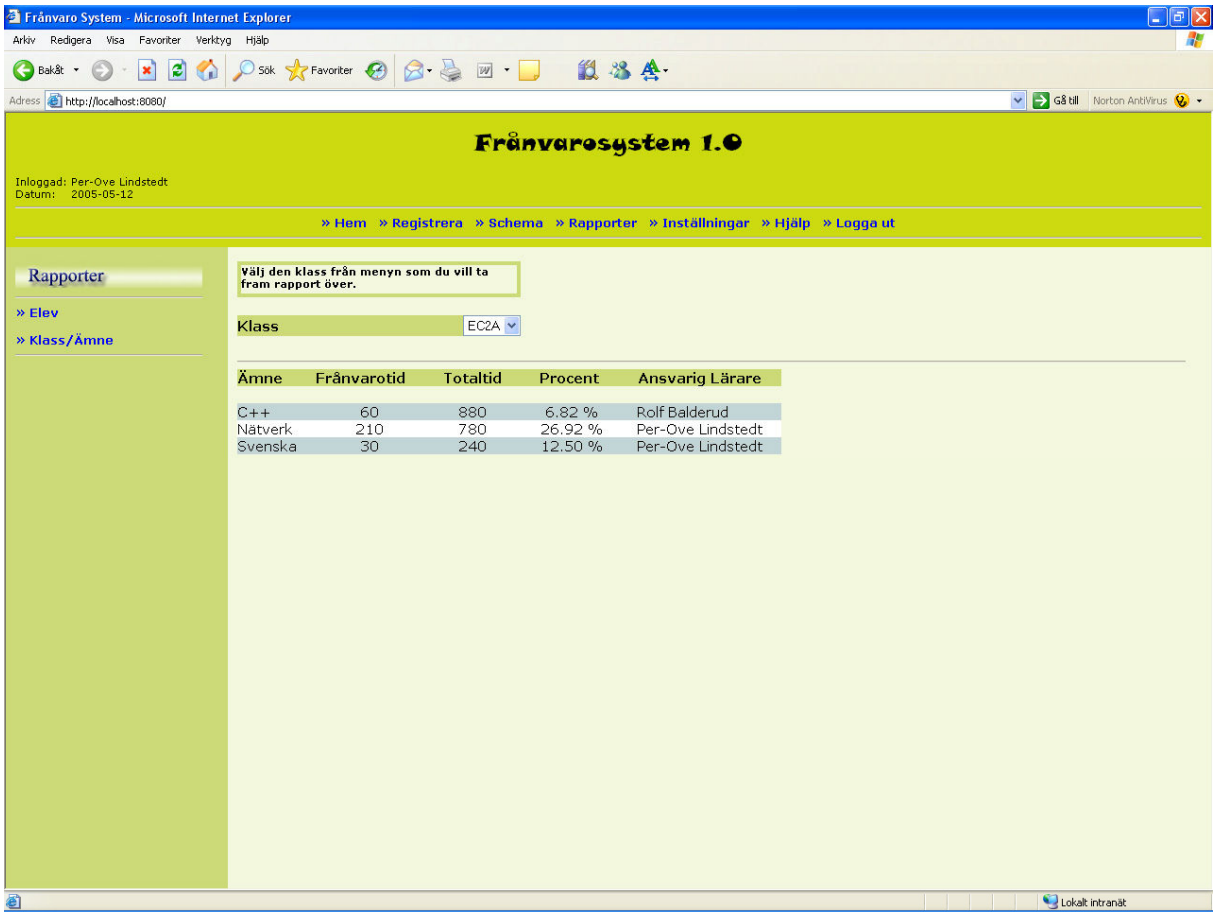
I detta avsnitt kommer vi att beskriva hur framtagningen av rapporter går till och vi kommer även att diskutera utseendet (gränssnittet) av rapporter som finns i figur 5-4 och 5-5. Det är här som användaren/läraren kommer att kunna ta fram rapporter för en elev eller för en hel klass. Dessa funktioner kan vara användbara när man ska ha föräldramöte eller om användaren vill ha en rapport över frånvaron till en lektion eller för en hel klass, så att man kan se om några åtgärder behöver göras ifall frånvaron är hög.



Figur 5-4 Rapporter för en elev

5.4.1 Utseende

Efter att ha klickat på länken Rapporter kommer man till en sida som innehåller två delar som visas i den undre delen av sidan. Den vänstra delen innehåller två länkar. Länken ”Elev” tar fram den information som finns i den högra delen i figur 5-4, mer om det finns i avsnittet 5.4.1.1. Länken ”Klass/Ämne” tar fram den information som finns i den högra delen i figur 5-5, mer om det finns i avsnittet 5.4.1.2.



Figur 5-5 Rapporter för en klass

5.4.1.1 Skapa elevrapport

Vi kommer nu att gå igenom informationen som visas i figur 5-4 i nedre delen till höger. Denna del består av två bitar. Den ena biten som finns ovanför linjen är ett formulär. Detta formulär består av en meny som innehåller alla klasser. I den andra menyn finns alla elever som går i den klassen man valt i den tidigare menyn. Efter dessa två menyer finns det textrutor där man får skriva in startdatum och slutdatum för när man vill ta fram rapporterna. Under textrutorna kommer det en knapp som heter ”Skicka”. Den andra biten som finns under linjen består av en ruta som innehåller information om eleven. Information som ingår i rutan är elevens personnummer, elevens namn och vilken klass eleven går i. Informationen är till för att läraren ska kunna se vilken elev rapporten gäller för. Under informationsrutan finns det en tabell med frånvarostatistiken över lektionerna som eleven har. Den information som denna tabell innehåller är:

- **Ämne:** Här finns namnet på ämnet.
- **Frånvarotid:** Här finns frånvarotiden för ämnet.
- **Totaltid:** Här finns den totala tiden för ämnet.

- **Procent:** Här finns antalet procent som eleven har varit frånvarande i ämnet.

Under tabellen visas en summering av informationen i form av den totala frånvarotiden i både minuter och procent.

5.4.1.2 Skapa klassrapport

Vi kommer nu att gå igenom information som visas i figur 5-5 i nedre delen till höger. Denna del består också av två bitar. Den ena biten som finns ovanför linjen är ett formulär. Detta formulär består av en meny som innehåller alla klasser. I den andra biten som finns nedanför linjen kommer det först fram klassens namn som man valt i formuläret och sedan en tabell. Den information som denna tabell innehåller är:

- **Ämne:** Här finns lektionens namn.
- **Frånvarotid:** Här finns frånvarotiden för ämnet.
- **Totaltid:** Här finns den totala tiden för ämnet.
- **Procent:** Här finns antalet procent som klassen har varit frånvarande i ämnet.
- **Ansvarig lärare:** Namnet på läraren som är ansvarig i ämnet.

5.4.2 Funktioner/funktionalitet

Om användaren vill skapa en rapport för en elev måste användaren klicka på Länken ”Elev”. Då uppdateras den nedre högra delen som kan ses i figur 5-4. Då kommer alternativen klass och elev fram som man kan välja. I alternativrutan klass finns alla klasserna på skolan som alternativ. Då användaren har valt en klass, fylls alternativrutan elever med elever som går i den valda klassen. När användaren har valt en elev kommer det automatiskt fram en rapport med den valde elevens lektioner och statistik över frånvaron. Det kommer också fram två datum som är startdatumet och slutdatumet för rapporten. Startdatumet är satt till datumet för den första lektionen och slutdatumet till dagens datum. Användaren kan ändra datumintervallet till önskat intervall, men för att uppdateringen av rapporten ska gå igenom måste användaren klicka på knappen skicka. Då uppdateras rapporten med det nya datumintervallet. Det inställda datumintervallet gäller även för andra elever i samma klass, om man skulle vilja se samma rapport för en annan elev i samma klass.

Om användaren vill skapa en rapport över frånvaro-statistiken för en hel klass måste användaren klicka på Länken ”Klass/Ämne”. Då uppdateras den nedre högra delen som kan ses i figur 5-5. Då kommer alternativrutan klass fram med alla klasser på skolan, där kan

användaren välja vilken klass han vill se rapporten över. När användaren har valt en klass blir rapporten över klass automatiskt synlig för användaren. Vill användaren se en rapport över en annan klass är det bara att ändra klass i alternativrutan som finns över rapporten uppdateras rapporten med den valda klassens rapport.

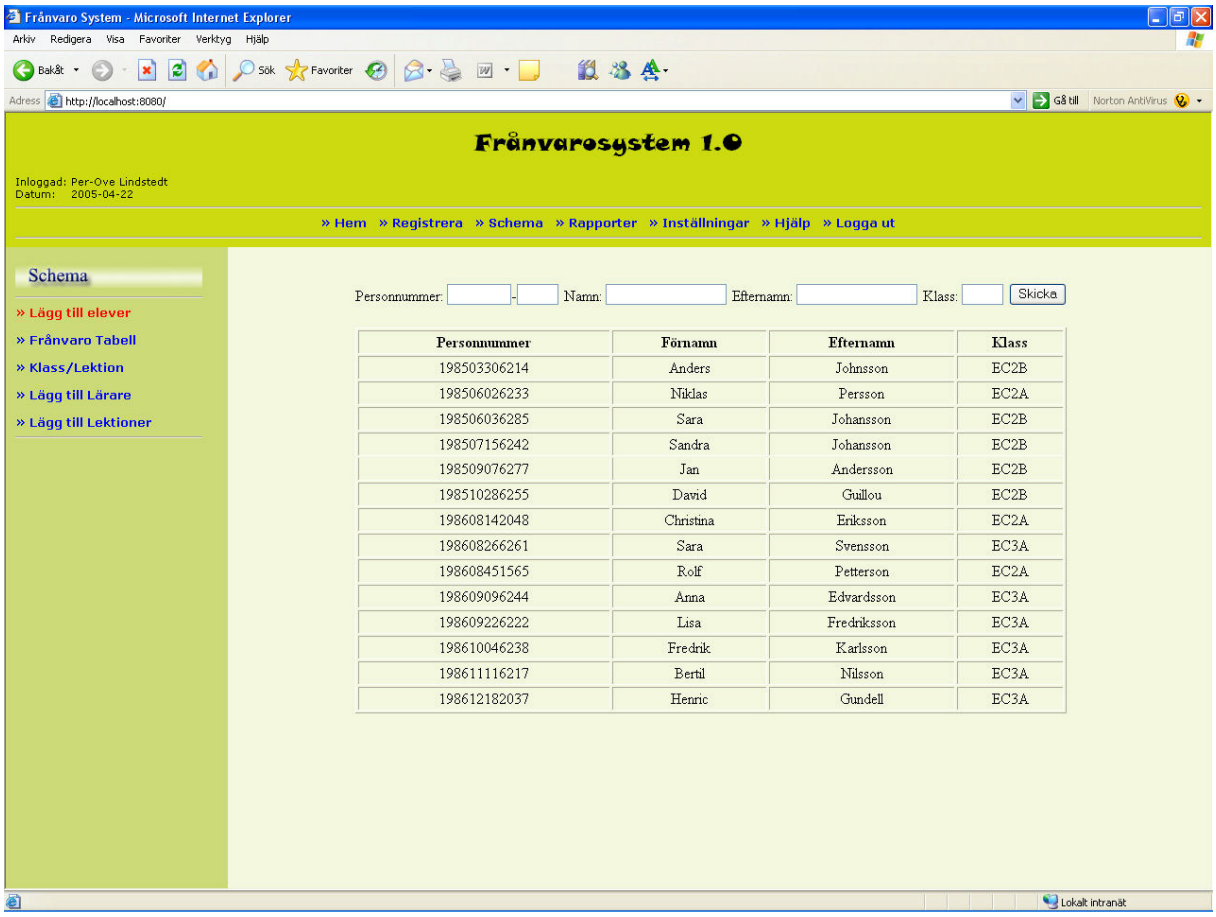
5.5 Framtida utvecklingar av systemet

I detta avsnitt kommer vi att ta upp några fler funktioner som skulle kunna läggas till i frånvarorapporteringssystemet. En del av dessa har vi nämnt tidigare men vi tar med dem här igen för att samla alla på ett ställe i rapporten.

5.5.1 Schemagenerator

I figur 5-6 kan man se en liten del av schemageneratorn. Det är här man ska kunna uppdatera frånvaro-databasen och det är här man ska kunna skapa ett schema som gäller för eleverna och ett schema som gäller för lärarna. Vi har hunnit göra väldigt lite av schemageneratorn men de funktioner som vi hittills har hunnit göra är:

- Lägga till nya elever.
- Frånvaro tabell/Uppdatering av frånvarotabellen.
- Lägga till klasser med tillhörande lektioner.
- Lägga till en ny lärare/användare.
- Lägga till nya lektioner för varje dag som varje lärare har.



Figur 5-6 Schema (Lägg till elever)

5.5.2 Ta fram rapporter i PDF-format i ett eget fönster

När man har valt ta fram en rapport för en elev eller för en klass, då skulle man kunna göra om rapporten från HTML till PDF-format för att få en snyggare utskrift.

6 Implementation

I detta kapitel kommer vi att beskriva delar av PHP-koden och vissa SQL-frågor. PHP-koden som vi valt att beskriva i avsnitt 6.1 är de delar som vi tyckte var mest intressanta. SQL-frågorna som beskrivs i avsnitt 6.2 är de viktigaste och de mest förekommande

6.1 PHP-kod

Vi kommer först att beskriva hur sessioner används i avsnitt 6.1.1. I avsnitt 6.1.2 beskriver vi hur uppkopplingen mot en MySQL-server fungerar i PHP, vilket följes av avsnitt 6.1.3 som beskriver hur frågor ställs mot en MySQL-databas.

6.1.1 Hur sessioner används i PHP

För att användaren ska kunna komma in i systemet behöver han/hon ett användarnamn och ett lösenord. För att göra inloggningen och användarens information säkrare har vi använt oss av sessioner [16]. Sessioner i PHP används för att skydda hemlig data. Man kan även använda SSL (Secure Socket Layer) [17] för att skydda integriteten hos sessionerna. Sättet som man kan använda SSL på är genom att aktivera det på servern.

För att använda sessioner behöver man anropa funktionen *session_start( )* som aktiverar sessioner. Anropet till *session_start( )* måste göras allra först i koden. Om man därefter vill lagra information som är bunden till sessionen kan man använda den superglobala arrayen *$_SESSION [ ]*. Att *$_SESSION [ ]* är superglobal array innebär att den är tillgänglig och kan användas i hela koden. När man är färdig med att använda sessioner eller när man loggar ut kan man välja att anropa funktionerna *session_unset( )* för att frigöra alla sessions-variabler och *session_destroy( )* för att förstöra all data som är associerad till den använda sessionen.

6.1.2 Uppkoppling till en MySQL-Server

För att skapa en uppkoppling i PHP mot en MySQL-databas av en nyare version än 4.1.2 används mysqli-tillägget [16] för att få tillgång till all funktionalitet. För att skapa ett mysqli-objekt anropar man konstruktorn med värddator, användarnamn, lösenord och databasens namn. Värddatorn är datorn som MySQL-servern befinner sig på, men kan även vara IP-

adressen till datorn. För att auktorisera sig används användarnamn och lösenord. Eftersom servern kan ha mer än en databas behövs även databasens namn för att specificera vilken databas man vill göra en uppkoppling emot.

För att se om uppkopplingen lyckades kan man göra en kontroll med hjälp av funktion *mysqli_connect_errno()* som returnerar 0 om inget fel har inträffat, annars returnerar den en fel-kod. Här nedan kan man se hur vår kod ser ut för uppkoppling. Vi har valt att placera koden för uppkopplingen i en separat fil som vi inkluderar varje gång som vi behöver göra en uppkoppling mot databasen.

```
$conn = new mysqli("localhost","användarnamn",
                   "Lösenord","databas_namn");

/* check connection */
if (mysqli_connect_errno())
{
    printf("Connect failed: %s\n", mysqli_connect_error());
    exit();
}
```

När man inte behöver uppkopplingen till databasen mer är det god sed att göra en nedkoppling för att frigöra de resurser som kopplingen har lagt beslag på. Nedkopplingen görs genom att anropa funktionen *close( )* för det aktuella kopplingen. I vårt fall har vi kallat kopplingsobjekt till MySQL-Server för *$conn,* vilket resulterar i nedkoplingssatsen som kan ses nedan.

```
$conn->close();
```

6.1.3 Användandet av en SQL-fråga i PHP

För att ställa en fråga mot en SQL-databas används metoden *query( )* för kopplings-objektet i PHP. Metoden tar emot en SQL-fråga som argument och returnerar ett mysqli_result-objekt. För att plocka fram data ur ett mysqli_result-objekt används funktionen *fetch_row().* Mysqli_result-objekt har även många andra metoder, som tillexempel *fetch_field( )* för att hämta en kolumn och *data_seek( )* används för att flytta den interna pekaren i databasen. Men mysqli_result-objektet innehåller även egenskaper som *num_rows* som returnerar antalet

rader och *field_count* som returnerar antalet kolumner. Nedan kan man se hur en fråga skapas och hur man får fram resultatet.

```
$resultat = $conn->query("SELECT * FROM elever e");

while($row = $resultat->fetch_row()) { //kod }
```

6.2 SQL-frågor

I detta avsnitt ska vi beskriva några olika SQL-frågor som vi har implementerat. Många av SQL-frågorna vi har använt är ganska lika, därför har vi valt att beskriva bara de viktigaste och mest förekommande frågorna i avsnitt 6.2.1. I avsnitt 6.2.2 beskrivs några av funktionerna som vi har använt i MySQL-frågorna.

6.2.1 Vanliga SQL-frågor

Fråga 1 är en enkel fråga som hämtar kolumnen *Fornamn* från tabellen elever. Om man vill få fram alla kolumner från en tabell får man använda stjärnoperatorn istället för *e.Fornamn* och det kan ses i fråga 2.

```
1. SELECT e.Fornamn FROM elever e

2. SELECT * FROM elever e
```

I fråga 3 hämtas de rader som innehåller samma *Lektions_Id* som PHP-variabeln *$LektionsID*.

```
3. SELECT f.Elev_personnummer FROM `frånvaro` f
   WHERE Lektions_Id = '$LektionsID'
```

I fråga 4 används kommandot *UPDATE* för att uppdatera de befintliga raderna i tabellen. I vårt exempel nedan uppdateras kolumnerna *Frånvarande*, *Antal_minuter*, *Anledning* och *kommentar* med respektive PHP-variabler *$Franvarande, $Minuter, $Anledning* och *$Kommentar* där *Lektions_Id* och *Elev_personnummer* stämmer överens med PHP variablerna *$LektionsID* och *$Personnummer*.

```
4. UPDATE `frånvaro` f SET f.Frånvarande = '$Franvarande',
   f.Antal_minuter = '$Minuter', f.Anledning = '$Anledning' ,
   f.kommentar = '$Kommentar' WHERE f.Lektions_Id = '$LektionsID'
   AND f.Elev_personnummer = '$row[0]'
```

Kommandot *INSERT* används för att lägga in värden i tabeller. För att få värdet i rätt kolumn används funktionen *values()* för att para ihop rätt värde med rätt kolumn. I fråga 5 paras kolumnerna i ordningen *Lärare_Id, Lärare_förnamn, Lärare_efternamn* och *Lärare_personnummer* med respektive PHP-variabler.

```
5. INSERT INTO lärare (Lärare_Id, Lärare_förnamn,
   Lärare_efternamn, Lärare_personnummer) values('$Larare_Id',
   '$Fornamn', '$Efternamn', '$Personnummer')
```

Nyckelordet *DISTINCT* används för att ta slippa dubbletter och endast få fram unika värden i resultatet från en SQL-fråga. Vi har använt det när vi skapade en listruta med alla klasser, vilket kan ses i fråga 6. Vi har även använt *ORDER BY* för att få resultatet i samma ordning varje gång frågan skapas.

```
6. SELECT DISTINCT Klass FROM elever e ORDER BY Klass
```

6.2.2 Funktioner i MySQL

I detta avsnitt kommer vi att beskriva några funktioner som vi har använt. Vi gör det genom att visa en SQL-fråga som vi har använt och sedan beskriva vad funktionerna gör. Frågan ska returnera ämnesnamn, den totala tiden för ett ämne, den totala frånvaron för ett ämne och frånvaron i procent med två decimaler.

```
7. SELECT  Ämne,  SUM(Antal_minuter),
   SUM((TIME_TO_SEC(TIMEDIFF(l.Lektion_Slut,l.Lektion_Start)))/'60
   '), ROUND((SUM(Antal_minuter) /
   SUM((TIME_TO_SEC(TIMEDIFF(l.Lektion_Slut,l.Lektion_Start)))/'60
   ') )*100 ,2) FROM `frånvaro` f, lektioner l Where l.Lektions_Id
   = f.Lektions_Id AND Elev_personnummer = '$Elev_pnr' AND l.Datum
   >= CONCAT('$Datum_FranAR','-','$Datum_FranMan','-
   ','$Datum_FranDag') AND l.Datum <= CONCAT('$Datum_TillAR','-
   ','$Datum_TillMan','-','$Datum_TillDag') GROUP BY Ämne
```

Först grupperas alla ämnen med *GROUP BY ämne*. Anledningen till det är att kunna använda funktionen *SUM( )* som då kan beräkna den totala summan. För att beräkna den totala lektionstiden för ett ämne användes *TIMEDIFF( )* som ger skillnaden mellan när lektionen startar och slutar. För att konvertera resultatet till sekunder används funktionen *TIME_TO_SEC( )*. För att få den totala lektionstiden används funktionen *SUM( ).* För att få frånvaron i procent med två decimaler används funktionen *ROUND(X, D)*, där argumentet X avrundas till D decimaler. Vi har använt funktionen *CONCAT( )* för att bygga ihop en datum-sträng som vi kan jämföra med datumet som finns i databasen. *CONCAT(str1,str2,..)* lägger ihop strängarna i den ordningen som de kommer som argument till funktionen.

Slutsatser

När vi började arbeta med projektet visste vi väldigt lite om hur svårt det skulle bli att utföra arbetet. Vi hade inte stora kunskaper inom webbprogrammering och databaser. Därför var vi tvungna att lägga tid på att hitta det rätta språket för oss och sedan på att lära oss språket och databashanteraren som vi behövde använda för att skapa frånvarosystemet.

I början av projektet hade vi en kravspecifikation som vi fick av uppdragsgivaren. Vi har fått bearbeta kravspecifikationen lite och den slutliga kravspecifikationen kan ses i kapitel 2. De flesta krav som ställdes av uppdragsgivaren har hunnits med, vissa funktioner återstår att implementeras eftersom tiden inte räckte till. De delar som vi ännu inte hunnit implementera är att lärarna ska kunna utforma sina egna rapporter genom att välja vilka fält som ska vara med och andra egenskaper som tillexempel fältbredd. De funktioner som vi hittills har implementerat har testats av både oss och uppdragsgivaren för att se att allt fungerar som uppdragsgivaren vill att systemet ska fungera. Vi kan även nämna att uppdragsgivaren och andra lärare på Nobelgymnasiet är nöjda med det resultat som vi har åstadkommit.

Eftersom mycket var nytt för uppdragstagarna, har det dykt upp en del problem under projektets gång. Ett av dessa problem var uppkopplingen av Apache, PHP och MySQL. Anledningen till detta var att vi först i början inte hittade mycket information om hur man skulle använda mysqli-tilläget i PHP5. Ett annat problem som vi hade var när vi skapade träd-menyn som används vid registrering av frånvaro för att ta fram lektionerna som lärarna har varje dag. Då fick vi problem med blinkningar som uppkom efter att ha klickat i trädet som var irriterande. Det berodde antagligen på att trädet inte var optimalt skrivet. Vi valde efter ett tag att ta ett redan befintligt JavaScript-träd och omarbeta det så att det passade oss, istället för att lägga ner mer tid på vårt träd.

För de personer som ska göra ett liknande arbete rekommenderar vi att man bör ha bra kunskaper i webbprogrammering och databaser. Du bör lägga ner mer tid åt att skapa kravspecifikationen innan man börjar med själva arbetet och efter att kravspecifikationen är färdig, bör du lägga mer tid åt att skapa databasen innan du börjar med implementationen. Om databasen är bra gjord från början blir det lättare att implementera. Det är bra om man skriver

rätt namn på de variabler och funktioner som används i implementeringen från början så att namnen beskriver dess funktionalitet. Det är även väldigt viktigt att skriva ner de problem som dyker upp under projektets gång, annars är det enkelt att glömma bort dem.

Referenser

[1] Amanda W. Wu, Haibo Wang and Dawn Wilkins, *Preformance Comparison of Alternative Solutions for Web-To-Database Applications*, 2000-10-28

[2] Omander, M, Din guide till Internetprogrammering, *Bonnier Icon*, 1998

[3] Ronne Erik, Java, *Deocendo Läromedel AB*, 2000

[4] Goodwill James, Developing Java Servlets Second Edition, *SAMS*, 2001

[5] Hunter Jason, Java Servlet Programming, *O´Reilly & Assocsicates*, 1998

[6] The PHP Documentation Group, `http://se2.php.net/manual/sv/print/history.php`, 2005-02-09

[7] Converse Tim and Park Joyce, PHP 4 Bible, *IDG Books Worldwide Inc*, 2000

[8] Zend, the PHP company, `http://www.zend.com/zend/future.php`, 2005-02-09

[9] Francis Brian, Beginning Active Server Pages 2.0, *Wrox Press Ltd*, 1999

[10] The Linux Web Server CD Bookshelf, `http://www.unix.org.ua/orelly/linux/sql/`, 2005-02-23

[11] Reese George, Managing & Using MySQL, *O´Reilly & Assocsicates*, 2002

[12] BuBois Paul, MySQL, *New Riders*, 2000

[13] MySQL documentation, `http://dev.mysql.com/doc/`, 2005-02-16

[14] Gutmans Andi, PHP 5 Power Programming, *Prentice Hall PTR*, 2004

[15] Luke Welling, MySQL Tutorial, *Sams Publishing*, 2003

[16] The PHP website, `http://se2.php.net/`, 2005-04-21

[17] Wikipedia, `http://en.wikipedia.org/wiki/Secure_Sockets_Layer`, 2005-04-21

[18] ThinkHost, `http://www.help.thinkhost.com/web-development/databases/why-mysql_378.html`, 2005-02-17

[19] Microsoft, `http://www.microsoft.com/WindowsServer2003/iis/default.mspx`, 2005-02-18

[20] OMG, `http://www.omg.org/technology/documents/formal/corba_iiop.htm`, 2005-02-18

[21] Sun Microsystems, `http://java.sun.com/`, 2005-02-18

[22] Linux self help,
http://www.linuxselfhelp.com/mysql/manual_Unireg.html, 2005-02-23

[23] The source for Perl, http://www.perl.com/, 2005-02-23

[24] Mesbah Ahmed, ASP.NET Web Developer's Guide, *Syngress Publishing*, 2000

A Relationsdatamodellen

A.1 Beskrivning av tabellen elever.

Elev_Personnummer	**Beskrivning:**	Elevens personnummer som är unik
	Datatyp:	VARCHAR(12)
	Obligatorisk:	Ja
Klass	**Beskrivning:**	Den klass som eleven går i
	Datatyp:	VARCHAR(10)
	Obligatorisk:	Ja
Förnamn	**Beskrivning:**	Elevens förnamn
	Datatyp:	VARCHAR(45)
	Obligatorisk:	Ja
Efternamn	**Beskrivning:**	Elevens efternamn
	Datatyp:	VARCHAR(45)
	Obligatorisk:	Ja
Nycklar		
Primär:	Elev_Personnummer	
Främmande:	-	

A.2 Beskrivning av tabellen frånvarande.

Minuter	**Beskrivning:**	Antal minuter som eleven är frånvarande
	Datatyp:	INTEGER
	Obligatorisk:	Nej
Kommentar	**Beskrivning:**	Lärarens kommentarer
	Datatyp:	VARCHAR(45)
	Obligatorisk:	Nej
Frånvarande	**Beskrivning:**	Om eleven är frånvarande eller närvarande
	Datatyp:	ENUM( )
	Obligatorisk:	Ja
Anledning	**Beskrivning:**	Anledning för att eleven är frånvarande
	Datatyp:	ENUM( )
	Obligatorisk:	Ja
Elev_Personnummer	**Beskrivning:**	Elevens personnummer som är unik
	Datatyp:	VARCHAR(12)
	Obligatorisk:	Ja
Lektions_Id	**Beskrivning:**	Unikt nummer för lektionen
	Datatyp:	INTEGER
	Obligatorisk:	Ja
Nycklar		
Primär:	(Lektions_Id, Elev_Personnumer)	
Främmande:	Lektions_Id, Elev_Personnummer	

A.3 Beskrivning av tabellen lektion.

Ämne	**Beskrivning:**	Namnet för kursen
	Datatyp:	VARCHAR(45)
	Obligatorisk:	Ja
Lektions_Id	**Beskrivning:**	Unikt nummer för lektionen
	Datatyp:	INTEGER
	Obligatorisk:	Ja
Datum	**Beskrivning:**	Det datum då lektionen går
	Datatyp:	DATE
	Obligatorisk:	Ja
Start_Tid	**Beskrivning:**	Tiden då lektionen börjar
	Datatyp:	TIME
	Obligatorisk:	Ja
Slut_Tid	**Beskrivning:**	Tiden då lektionen slutar
	Datatyp:	TIME
	Obligatorisk:	Ja
Lärare_Id	**Beskrivning:**	Unikt nummer för läraren
	Datatyp:	VARCHAR(6)
	Obligatorisk:	Ja
Nycklar		
Primär:	Lektions_Id	
Främmande:	Lärare_Id	

A.4 Beskrivning av tabellen lärare.

Förnamn	**Beskrivning:**	Lärarens förnamn
	Datatyp:	VARCHAR(45)
	Obligatorisk:	Ja
Efternamn	**Beskrivning:**	Lärarens efternamn
	Datatyp:	VARCHAR(45)
	Obligatorisk:	Ja
Lärare_Id	**Beskrivning:**	Unikt nummer för läraren
	Datatyp:	VARCHAR(6)
	Obligatorisk:	Ja
Lärare_Personnummer	**Beskrivning:**	Lärarens personnummer som är unik
	Datatyp:	VARCHAR(12)
	Obligatorisk:	Ja
Lösenord	**Beskrivning:**	Lärarens lösenord för att logga in i systemet
	Datatyp:	VARCHAR(45)
	Obligatorisk:	Ja
Nycklar		
Primär:	Lärare_Id	
Främmande:	-	

A.5 Beskrivning av tabellen klass/lektion.

Klass	**Beskrivning:**	Klassen som går i den speciella lektion
	Datatyp:	VARCHAR(10)
	Obligatorisk:	Ja
Lektions_Id	**Beskrivning:**	Unikt nummer för lektionen
	Datatyp:	INTEGER
	Obligatorisk:	Ja
Nycklar		
Primär:	(Klass, Lektions_Id	
Främmande:	Lektions_Id	

B Koppling mellan Apache, MySQL och PHP

För att skapa en koppling mellan Apache, PHP och MySQL Server måste man först installera dessa programvaror och sedan göra en del ändringar i några filer för att få allt att fungera. I avsnitt B.1 beskrivs vilka programvaror som används, vart man kan hitta dem och hur de ska installeras på datorn. I avsnitt B.2 beskrivs de ändringar man bör göra med filerna.

B.1 Installation av programvara

De programvaror som användes under skapandet av *frånvarosystemet* var MySQL 4.1.7, Apache HTTP Server 2.0.52 och PHP 5.0.3. Dessa programvaror användes i Microsoft Windows miljö.

B.1.1 Apache

1. Gå till `www.apache.org`.
2. Klicka på *HTTP Server* länken.
3. Klicka på *download from a mirror*.
4. Klicka på *Win 32 Binary MSI Installer* länken för att ladda ner *apache_2.0.52-win32-x86-no_ssl.msi*.
5. Efter att ha laddat ner klickar du på MSI filen för att påbörja installationen av Apache HTTP Server.
6. Klicka på *next*. Då kommer du att se *Server Informations screen*.
7. Skriv in denna information:
    - *Domain namn:* Tillexempel domainname.com.
    - *Server namn:* Tillexempel server.domain.com.
    - Nätverks administratörens e-mail adress.
    - För vem ni installerar Apache.
8. Välj ”*All users*” om du vill ha en server som är tillgänglig för vem som helst eller välj ”*Only the current user*” om du vill ha en server som ska användas för testningens ändamål.
9. Klicka på *next* för att välja en installationstyp.

10. Klicka *next* efter att ha valt installationstyp.
11. Om du inte vill ha dina Apache-filer sparade i default sökvägen, klicka då på *Change* och välj sedan en alternativ sökväg och sedan klickar du på nästa.
12. Klicka på *install* för att slutföra installationen.

B.1.2 PHP

1. Gå till `www.php.net`.
2. Klicka på *downloads*-länken.
3. Klicka på *php-5.0.2-Win32.zip*-länken som finns under rubriken *PHP 5.0.3* på *Windows Binaries*.
4. Packa upp zip-filen med ett unzip program på c:\php\. Php-mappen finns inte på roten men du får skapa den själv. Alla filer som finns i packade filen ska hamna i mappen php.

B.1.3 MySQL

1. Gå till `www.mysql.com`.
2. Klicka på *developer zone*.
3. Klicka på länken *Generally Available (GA) 4.1.7* som finns under rubriken *MySQL Database Server*.
4. Klicka på länken *Pick a mirror* som finns under rubriken *Windows (x86)*.
5. Klicka på länken *HTTP* eller *FTP* för att börja ladda ner.
6. Gå till mappen där du sparade ner zip-filen och packa upp den filen.
7. Öppna *setup.exe* filen kör installationen.
8. Klicka *Next* för att komma till *informational screen*. Rekommenderas starkts att läsas.
9. Klicka på *Next* för att komma till *Choose Destination Location screen*.
10. Klicka på *Next* om du vill att den ska sparas på *default* sökvägen, annars kan du välja att spara den på en annan plats.
11. Klicka på *Next*.
12. I nästa fönster har du möjlighet att skräddarsy din egen installation, men du rekommenderas att välja en *typical* installation.
13. Klicka på Next när du har valt installationstyp och sedan klickar du på *Finish* för att avsluta installationen.

När installationen av *MySQL Database Server* är färdig kan du ladda ner *mysql-query-browser-1.1.5-winl* och *mysql-administrator-1.0.19-win*. Dessa två är grafiska klienter. Istället för att skriva alla frågor i *dos*, kan du använda dig av *MySQL Administrator* och *MySQL Query Browser*. I *MySQL Administrator* kan du skapa alla entiteter med sina attribut och du kan även ange vilka attribut som ska vara främmandenycklar och primärnycklar. Du kan även ange vilka entiteter som ska vara kopplade till varandra. I *MySQL Query Browser* kan du skapa en massa frågor över din databas och se vad du får för svar.

B.2 Ändringar med filerna

B.2.1 Ändringar i PHP

- Gå in i mappen *ext* som finns under *c:\php* och kopiera filerna *php_mysql.dll* och *php_mysqli.dll* och klistra in dem i mappen *php*.
- Gå in i mappen *php* som finns under c:\ och ändra filen *php.ini-dist* till *php.ini*
- Öppna filen *php.ini* och gör dessa ändringar:
  extension=php_mysqli.dll
  mysqli.default_port = 3306
  Om ni har valt porten 3306 för MySQL.

- Efter att ha gjort dessa ändringar väljer du att spara *php.ini* och stänger ner den. Sedan kopierar du filen *php.ini* och klistrar in den i mappen *Windows* som finns under roten c:\.

B.2.2 Ändringar i Apache

- Öppna *httpd.conf* som finns under *C:\Program\Apache Group\Apache2\conf* med en textredigerare och lägg till dessa ändringar:
  LoadModule php5_module c:/php/php5apache2.dll

  AddType application/x-httpd-php .php3 .html .php .php5
  AddType application/x-httpd-php-source .phps

  Ändringarna nedan görs för att all html- php-kod som du skriver ska sparas under mappen test och då kommer apache att leta efter koden i mappen test
  DocumentRoot "C:/Program/Apache Group/Apache2/test"

<Directory "C:/Program/Apache Group/Apache2/test">

Efter att du har gjort dessa ändringar väljer du att spara *httpd.conf* och stänger ner den.

- Gå in i mappen *php som finns under c:\php* och kopiera filen *php5ts.dll* och sedan klistrar du in den i mappen *bin* som finns under *C:\Program\Apache Group\Apache2*.