Webbaserat bokningssystem för studiehandledare

Rasmus Nylund

Examensarbete för Tradenom (YH)-examen

Utbildningsprogrammet i informationsbehandling

Raseborg 2015

EXAMENSARBETE

Författare: Rasmus Nylund

Utbildningsprogram och ort: Informationsbehandling, Raseborg

Handledare: Rolf Gammals

Titel: Webbaserat bokningssystem för studiehandledare

Datum: 05.05.2015 Sidantal: 28 Bilagor: 1

Abstrakt

Mitt uppdrag för examensarbetet är att utveckla ett webbaserat bokningssystem där studerande kan boka handledningstider till studiehandledaren.
Uppdragsgivaren är Yrkeshögskolan Novia och bokningssystemet används inom Raseborgsenheten.
Syftet är att förenkla bokningsprocessen för både studeranden och studiehandledaren.

Bokningssystemet är installerat på skolans intranät och är åtkomligt via internet med en webbläsare för alla som har inloggningsuppgifter till Novia. Bokningarna görs via ett kalender gränssnitt och att boka en tid kräver inte användarkonto. Studeranden har efteråt möjlighet att avboka mötet via en länk som skickas i ett e-postmeddelande. Bokningssystemet informerar per e-post båda parterna om ändringar och uppdateringar som sker till en bokning.

Studiehandledaren kan logga in till systemet för att administrera bokningarna och andra inställningar som har med bokningssystemets funktionalitet att göra.
Bokningssystemet stöder även studiehandledarens rapportering till arbetsgivaren över arbetsprestationer.

Bokningssystemet är huvudsakligen programmerat med PHP men även Javascript används vid validering av data och då Ajax anrop görs. Databashanteraren som används är MySQL.

Examensarbetet beskriver till stor del de tekniska aspekterna i bokningssystemet, men de är skrivna så att läsaren inte behöver ha tidigare kunskaper inom ämnet. Även användargränssnittet ur studerandens och studiehandledarens synvinkel beskrivs i detalj.

Språk: Svenska Nyckelord: Bokningssystem, PHP, Fancybox, Ajax

---

BACHELOR'S THESIS

Author: Rasmus Nylund
Degree Programme: Business Information Technology, Raseborg
Supervisors: Rolf Gammals

Title: Web based booking system for student counsellor /
Webbaserat bokningssystem för studiehandledare

Date: 5 May 2015 Number of pages: 28 Appendices: 1

Abstract

The task for the thesis is to develop a web-based booking system where students can book counselling sessions to the student counsellor. The commisioner of the thesis work is Novia University of Applied Sciences and the booking system is used by the unit in Raseborg.
The aim is to simplify the booking process for both the students and the counsellor.

The booking system runs on Novia's intranet and is accessible with a web browser for anyone who has log in credentials to Novia. Bookings are made through a calendar interface and no user account is required. Students have the ability to cancel meetings with a link sent to them by e-mail. The booking system informs by e-mail both parties if any changes are made to a reservation.

The student counsellor can login to the system to administer the bookings and other settings related to functionality of the system.
The booking system also supports the counsellor's reporting of work performance to the employer.

The programming of the system has mainly been done with PHP, but also Javascripts are used. A MySQL database is used for storing information.

The thesis largely describes technical aspects of the booking system but it is written so that the reader does not need to have previous knowledge of the subject. The user interface from the booker's and counsellor's point of view is also described in detail.

Language: English Key words: Booking system, PHP, Fancybox, Ajax

---

Innehållsförteckning

1 Inledning....1
1.1 Syfte och målsättning....2
2 Val av verktyg....2
2.1 Linux....3
2.2 Apache....3
2.3 MySQL....3
2.4 PHP....4
2.5 Javascript....4
2.6 Geany....5
3 Användargränssnitt....5
3.1 Att boka tid....6
3.2 E-post utskick....9
4 Administrering....10
5 Programmets tekniska uppbyggnad....12
5.1 Dynamisk layout....13
5.2 Sessioner....13
5.3 Filerna....14
5.4 Databasen....17
5.5 Ajax....19
5.6 Inloggning....20
5.7 Fancybox....21
5.8 Funktioner....21
5.9 Säkerhet....23
6 Testskede....23
6.1 Virtuell server som plattform för projektet....24
6.2 Mötet med studiehandledaren angående testningen....24
6.3 Förbättringar och korrigeringar enligt möte....26
7 Avslutning....27
8 Källförteckning....28

1 Inledning

Studiehandledningen på Yrkeshögskolan Novia är ett viktigt verktyg som finns till studerandens förfogande under hela studietiden. Handledningens syfte är att förebygga problem samt hjälpa studeranden att utvecklas och studera smart. Varje enhet har en egen studiehandledare och bokningssystemet som jag har utvecklat är för Raseborgsenhetens handledare.
(Novia.fi, 2014)

Studiehandledaren för handledningssamtal med studeranden och har tidigare bokat in samtalen för hand i en kalender. Bokningarna har studiehandledaren mottagit antingen per telefon eller via e-post. Denna bokningsprocess är klumpig och tidskrävande, eftersom bokningen måste passa både för den studerande och handledaren.
Mitt uppdrag har varit att utveckla ett webbaserat bokningssystem för studiehandledaren för att förenkla bokningsprocessen. Bokningssystemet körs på Yrkeshögskolan Novias intranät, vilket betyder att endast personer med inloggningsuppgifter kommer åt systemet.

Det nya bokningssystemet förenklar bokningsprocessen markant för både studeranden och handledaren, då endast lediga tider kan bokas och upptagna tiderna visas som bokade. Studiehandledaren behöver således inte lägga tid och resurser på att passa in tiderna och hålla reda på dem i papperskalendern. Dock har handledaren möjlighet att låta bli att godkänna ett möte om tiden inte passar av någon anledning. Studeranden har också möjlighet att avboka mötet elektroniskt via en länk som skickas i bekräftelsee-postmeddelandet.

Bokningssystemet är uppbyggt som en kalender, först väljer man rätt dag och sedan lägger man till en bokning på en ledig tid. När bokningen är skapad, blir både studiehandledaren och studeranden informerad om detta med ett e-postmeddelande. För att bokningen skall bli fastslagen loggar studiehandledaren in till bokningssystemet och godkänner mötet.

Bokningssystemet har genomgått ett testskede före publiceringen och tack vare det har användarvänligheten och funktionaliteten finslipats. Först testades användargränssnittet ur studerandens synvinkel och senare testades funktionaliteten ur administratorns synvinkel.

Det senare testskedet gjordes tillsammans med studiehandledaren för att i detalj gå igenom de krav som ställs av bokningssystemet.

1.1 Syfte och målsättning

Syfte med detta projekt är att skapa ett webbaserat system som möjliggör bokning av studiehandledningstider. Bokningssystemet är skapat med användbarheten i fokus, både ur studerandens och studiehandledarens synvinkel.

Examensarbetet är skrivet så att läsaren inte behöver ha tidigare kunskaper av bokningssystem eller systemutveckling. Det finns kodexempel med i arbetet, men den är kommenterad för att läsaren skall förstå dess grundprincip.

Jag har länge varit intresserad och sysslat med utveckling av webbapplikationer, så målet med detta arbete var att vidareutvecklas och lära mig nytt.
På mitt jobb arbetar jag till en viss del med liknande uppgifter, men detta bokningssystem är dock det största projekt jag någonsin utvecklat.

2 Val av verktyg

Examensarbetet är utfört på en plattform uppbyggd av fyra olika delprogram som tillsammans kallas för "LAMP". LAMP står för "Linux", "Apache", "MySQL" samt "PHP" och alla dessa delar baserar sig på öppen källkod och är gratis att ladda ner från internet. Jag valde verktyg som licensieras med öppen källkod eftersom dessa brukar uppdateras ofta och det finns massor med information och stöd tillgängligt på internet.

Jag har arbetat på examensarbete genom att ha installerat en LAMP server på min egen stationära maskin. På det viset har testningen av projektet varit snabb och smidig då allt har utförts på samma maskin. I ett senare skede flyttade jag över bokningssystemet till en virtuell server, varifrån det alltid var tillgängligt och testningen krävde inte att jag hade min stationära maskin påslagen.

2.1 Linux

Linux är själva operativsystemet. Egentligen är Linux endast en kärna för operativsystem, som sedan körs i samband med övrig programvara och kombinationen blir ett operativsystem. En stor del av nuvarande Linux distributionerna innehåller GNU programvara och en del är av den åsikten att själva distributionen i sådana fall skall heta "GNU/Linux". I praktiken kallar man ändå operativsystemet för bara "Linux", antagligen görs det för enkelhetens skull.
Som sagt finns det många distributioner av Linux. Vissa företag tillverkar egna distributioner och lägger till dem även stängd kod och säljer dem för en större summa, men många distributioner är ihop plockade av endast öppen källkod programvara, vilket möjliggör att ladda ner gratis hela distributionen. Linux samt GNU programmen ges ut under GPL licensen, vilken tillåter ändring av källkoden samt återdistribuering.
Själv använder jag Kubuntu 14.04 som är en KDE version av Ubuntu.

2.2 Apache

Apache är webbservern som upprätthåller webbsidorna som sedan skickas till besökarens webbläsare. Apache är känt för att ha hjälpt med tillväxten av World Wide Web. Sedan april 1996 har Apache varit den mest använda http servern på webben och i nuläget körs ca 60% av alla världens webbsidor på den. Apache utvecklas och underhålls av ett öppet samhälle av utvecklare som styrs av "The Apache Software Foundation" gruppen, som i sin tur innehar rättigheterna för programmet. Apache ges ut under dess egen licens som heter "Apache License". (Wikipedia).

2.3 MySQL

MySQL är ett "database management system" (DBMS) som kan kommunicera med massor av olika programmeringsspråk. I LAMP system är det PHP språket som kommunicerar med MySQL genom ett antal kommandon. Själva innehållet i databasen överförs med hjälp av SQL förfrågningar (Standard Query Language), vilket som namnet antyder är ett standardiserat förfrågningsspråk. MySQL grundades och utvecklades av ett svenskt företag (MySQL Ab). År 2008 köpte Sun Microsystems programsviten och år 2010 köptes hela Sun Microsystems företaget av Oracle, som i denna dag äger och upprätthåller MySQL. MySQL är mycket populärt databashanteringssystem inom Linux-världen, men efter att det

således av MySQL Ab har även andra databashanterare ökat i popularitet. Grundarna av MySQL började arbeta med en ny databashanterare år 2009 som heter MariaDB. Dess popularitet ökar hela tiden och en orsak till detta är att MySQL uppdateringarna som Oracle kommer ut med, är i stor utsträckning skyddade med upphovsrätt. MariaDB går under GNU General Public License, vilket betyder att källkoden är öppen och användandet är gratis. (Wikipedia).

2.4 PHP

PHP står för "PHP: Hypertext Preprocessor" och är ett programmeringsspråk som körs oftast på en webb-server. I sådant fall genererar den HTML kod som sedan visas på klientens arbetsstation. Arbetsstationen behöver således inte förstå PHP, utan endast standardiserad HTML kod. Detta gör underhållandet betydligt enklare eftersom uppdateringar måste endast göras på servern. PHP är ett skriptspråk som i jämförelse till många programmeringsspråk inte behöver kompileras, det är genast färdigt att köras. PHP utvecklas av "PHP Group" och ges ut under dess egen licens kallad "PHP License". (Wikipedia).

2.5 Javascript

Javascript erbjuder extra funktionalitet och egenskaper utöver det som standard HTML kan prestera. Javascript är också som namnet säger ett skriptspråk, men i motsats till PHP, som körs på servern, körs Javascript lokalt i klientens webbläsare. Detta gör t.ex. en validering av inmatad formulärinformation mycket snabb. Utan Javascript skulle data först behöva flyttas till servern, var ett PHP skript skulle validera informationen och sedan skicka antingen ett godkännande eller ett nekande tillbaka till klienten. För att vara helt säker på att data som användare matar in är korrekt, bör man validera den både i webbläsaren och på servern (Augustsson & Folkesson, 2010, s. 135).

I bokningssystemet använder jag ett gratis Javascript-bibliotek som heter jQuery. Detta bibliotek underlättar utvecklingsarbetet och vissa delar av bokningssystemets funktionalitet kräver jQuery för att fungera. Nästan 35% av de 1 000 000 mest besökta webbsidorna använder sig av jQuery, som för tillfället är det populäraste Javascript-biblioteket (Wikipedia).

2.6 Geany

Geany är den editor jag använder för skrivandet av programkoden. Geany är ett så kallat IDE (Integrated Development Environment), alltså en integrerad utvecklingsmiljö.
Den kommer med som standard i många Linux distributioner och är mycket resurssnål och lättanvänd.
Jag har egentligen bara några krav då det kommer till utvecklingsmiljö och Geany som säkert många andra liknande system har dem.
Några av egenskaperna i programmet som jag uppskattar är:
1. Möjligheten att ha flera flikar samtidigt öppna. Detta ger mig möjligheten att ha alla projektets filer öppnade samtidigt och enkelt navigera mellan dem.
2. Dess förmåga att känna igen kodspråket och därefter färga skilda delar av koden med olika färger. Detta gör koden mera överskådlig.

3 Användargränssnitt

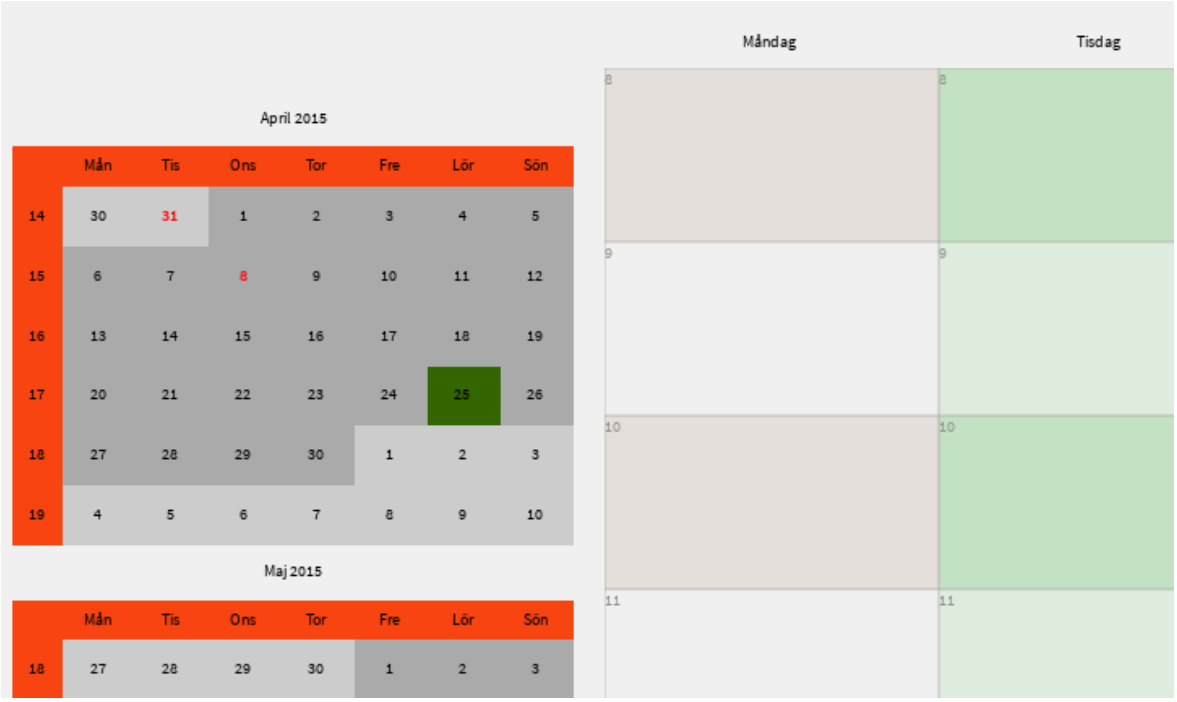
Bokningssystemets användargränssnitt är designat för att vara enkelt och tydligt att använda.

Huvudsidan är uppdelad i två bredvidliggande områden. Vänstra delen tar upp ungefär en femtedel och högra delen tar upp resten. Vänstra delen innehåller två rutfält som är under varandra med vy över hela månader. Översta rutfältet innehåller aktuella månadens alla dagar och den undre visar likadan vy över inkommande månad.

Månadsrutan visar dagarnas datum antingen med röd eller svart text. Röd text betyder att dagen i fråga har insatta möten och svart text att inga möten är insatta. Röd text betyder ändå inte att hela dagen är bokad, så det kan finnas lediga tider kvar. Svart text indikerar att inga möten finns insatta för befintlig dag.

Månadsrutorna har alltid lika många rutor i sig oberoende av dagmängden i månaden som visas. Om månadens första dag är t.ex. en fredag, visas den veckans alla dagar före fredagen med föregående månadens datum. Med samma princip visas veckan med månadens sista dag, alltså visas datum som hör till inkommande månaden på de sista rutorna.

Månadsrutorna visar även på kolumnen längst till vänster veckonummer för varje vecka och på raden längst uppe visas veckodagarnas namn. Figur 1 visar en specifik del av huvudsidan, där en del av månads- och veckovyn syns.



Figur 1: Delar av månads- och veckovyn

Genom att klicka på en dag i någondera månad visas den dagens samt hela veckans alla möten i högra delen av huvudsidan. Veckovyn tar ungefär ¾ av hela utrymmet på sidan och den är uppdelad i sju delar som representerar varje veckodag, börjande med måndag till vänster och söndag till höger. Fortsättningsvis är varje dag uppdelad i rutor som representerar hela timmar. Första timmen börjar från 8.00 och sista rutan börjar med 15.00 och slutar med 16.00. Start- och sluttiden befinner sig i varsin variabel, som studiehandledaren kan ändra från administreringsfönstret. Samtliga dagar använder samma start- och sluttid.

Befintliga möten visas som rutor som sträcker sig över den tidsperioden de blivit bokade för och innanför rutorna står "BOKAD TID". En besökare kan således inte få annan information om bokade möten, förutom dess start och sluttid. Se bilaga 1 för helhetsbild över bokningssystemet.

3.1 Att boka tid

För att boka en tid, skall först rätta dagen väljas från månadsvyn till vänster. Därefter klickar man på önskat klockslag i veckovyn på den önskade dagen.

I månadsvyn kan endast aktuella datumet eller nyare väljas. Samma princip gäller vid veckovyn, endast klockslag i framtiden kan klickas på och de är markerade med grön bakgrundsfärg. De klockslagen man inte kan boka på har grå bakgrundsfärg. Lördag och söndag har således alltid grå bakgrundsfärg.

Som standard kan en bokning göras tidigast ett dygn plus en timme framåt från aktuella tiden. Denna tidsfaktor beaktar inte lördag och söndag, eftersom man inte kan göra bokningar till dessa dagar. Exempelvis om klockan är 10.45 på en fredag, blir första möjliga klockslaget man kan boka på klockan 11 inkommande veckas måndag.

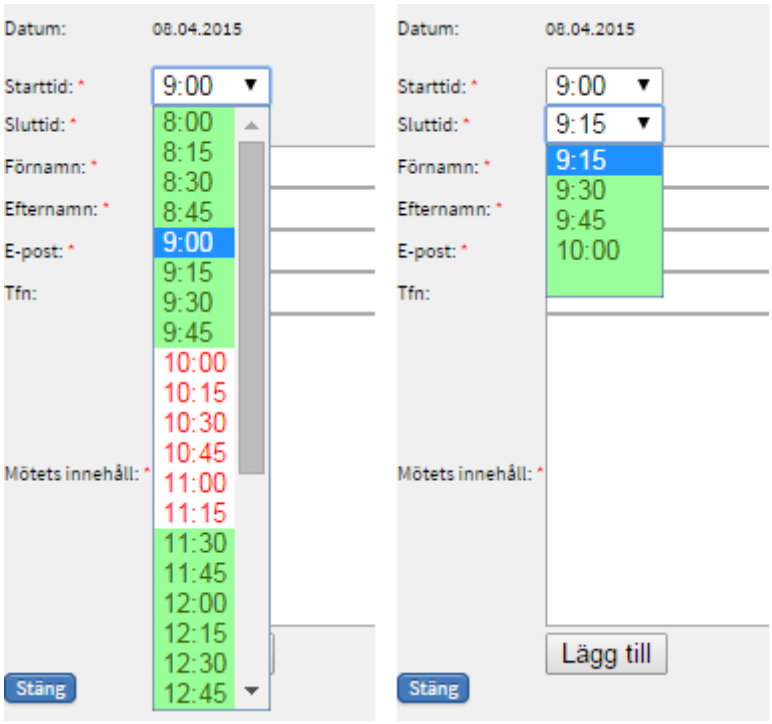
Studiehandledaren kan från administreringsfönstret ändra denna tidsfaktor till önskat värde.

När man klickar på en önskad tidpunkt öppnas en Fancybox ruta där besökaren med hjälp av ett formulär kan ansöka om ett möte. Varje timme är uppdelad i 15 minuters sektioner, vilket innebär att ett möte kan starta och sluta vid jämna timmar, kvart över, halv och kvart i.

Start- och sluttiden skall först anges från rullgardinsmenyerna. Starttiden får som standard värde den timme man klickat på och sluttiden blir 15 minuter framåt från starttiden.

Tiderna som kan väljas tar i beaktande redan bokade möten, så om ett befintligt möte slutar inom samma timme man bokat, skjuts starttiden framåt till samma tidpunkt som det befintliga mötets sluttid är. Starttidens rullgardinsmeny visar tiderna som redan innehåller möten med röd text och de går inte att välja. De tider som är lediga visas med grön text.

Varje gång en starttid väljs, anropas en funktion som räknar ut vilka sluttiderna kan vara och uppdaterar sluttidens rullgardinsmeny med dessa alternativ. Figur 2 illustrerar hur starttiderna visas i rullgardinsmenyn om man klickat på klockslaget 9 och på samma dag finns redan ett möte från 10.00 till 11.30. Figuren visar även sluttiderna och här kan ses att sista möjliga tid som går att välja är 10.00, eftersom då börjar det existerande mötet.



Figur 2: Val av start- och sluttid vid mötesbokning

Förutom tiderna skall även de övriga obligatoriska fälten fyllas i. De är förnamn, efternamn, e-post, samt beskrivning över vad mötet kommer att handla om. Telefonnummer är enda fältet som inte är obligatoriskt, alltså får det lämnas tomt om så önskas.

I utförda test av användbarhet har det kommit fram att ett vanligt tankefel är att förutsätta att alla användare anger uppgifter på samma sätt (Eriksson, 2008, s. 279).

För att minska sannolikheten för misstag vid skapandet av ny bokning, är systemet uppbyggt med den principen, att användaren skall välja bland färdiga alternativ som bjuds ut.

När all data är ifylld, skickas mötesbegäran in genom att klicka på "Lägg till" knappen. Formulärets innehåll skickas först till ett Javascript som validerar informationen. Valideringen kontrollerar till först att alla obligatoriska fällt är ifyllda och sedan jämförs e-postadressen mot vissa kriterier, för att minska sannolikheten att den är felaktigt skriven.

Då något av fälten inte godkänns, visas ett förklarande felmeddelande med röd text en stund, var efter det tonas bort. Om valideringen godkänner alla fält, skickas de med Ajax anrop till update_data.php filen som i sin tur kör in uppgifterna till databasen. När mötet är insatt, visas meddelande "Bokning mottagen" i två sekunder och sedan stängs Fancybox rutan som användes för att lägga till mötet.

Nu syns valda veckan och dess bokade möten, även det som just sattes till.

Bokningar som görs, får automatiskt en status "Inte godkänd", men ändå syns de genast som "BOKAD TID" och går inte att boka på. Endast om studiehandledaren raderar bokningen, kan dessa reserverade tider användas på nytt för nya bokningar.

Om två eller flera besökare skulle samtidigt försöka lägga till ett möte på samma lediga plats, skulle den som först hinner skicka begäran vara den vars möte bokas. För att undvika att databasen blir korrumperad, körs en låsning på databasen i samma väva som anropet om ny bokning körs. Detta innebär att endast den process som lägger till ett möte har tillgång till databasen. När allt är insatt, öppnas databasen igen så att en ny insättning av möte är möjlig. Insättningen av ett möte till databasen görs enligt kodexempel 1.

Kodexempel 1: Databaskörningen som lägger till ett möte.

```
//Här låses databasen, så endast denna process har tillgång.
$sql_query1 = "LOCK TABLES client, meeting WRITE";

//Denna förfrågan lägger in mötet i databasen.
$sql_query2 = "INSERT INTO meeting (date, hour_start, hour_stop, content, status, code)
VALUES ('" . $m_date . "','" . $m_start . "','" . $m_end . "','" . $content . "','2','" . $auth_code . "')";

//Denna förfrågan lägger bokarens uppgifter i databasen. Kopplingen till mötet görs med last_insert_id().
$sql_query3 = "INSERT INTO client (firstname, surname, email, phone, active, meeting_id)
VALUES ('" . $firstname . "','" . $surname . "','" . $email . "','" . $phone . "','1',last_insert_id())";

//Denna låser upp databasen, så andra processer har tillgång.
$sql_query4 = "UNLOCK TABLES";
```

3.2 E-post utskick

Bokningssystemet är uppbyggt så att information angående nya bokningar samt ändringar till befintliga, skickas ut per e-post åt både bokaren och studiehandledaren.

När en ny bokning blir insatt av en besökare, anropas funktionen sendMail() vars uppgift är att skicka e-post. Funktionen anropas två gånger. Första gången skickar den ett e-postmeddelande till bokaren, alltså till den adress som angetts i formuläret då ny bokning skapats. Andra gången skickas ett e-postbrev till studiehandledaren, för att informera om att en ny bokning har blivit skapad. E-postadressen som skickas till, kan studiehandledaren själv ange från administreringsfönstret.

Meddelandet som skickas till bokaren har som ämne ”Bokningen mottagen” och innehåller mötets tidpunkt samt en länk som möjliggör avbokning av mötet.

Eftersom mötena kan handla om känsliga ämnen, syns aldrig mötets innehåll i något e-post som bokningssystemet skickar ut. Detta var ett krav som uppdragsgivaren hade.

När bokningen är insatt kan studiehandledaren godkänna mötet genom att logga in till bokningssystemet. Genast då bokningen godkänts, anropas igen sendMail() funktionen och nu skickas ett e-postmeddelande till bokaren om att mötet godkänts. Denna gång anropas sendMail() funktionen endast en gång, för det finns ingen orsak att skicka meddelande till studiehandledaren, då det var hon som godkände mötet.

Om bokaren bestämmer sig för att avboka mötet, görs det genom att klicka på avbokningslänken i e-postmeddelandet. Detta öppnar en ny webbsida som endast frågar efter orsak till avbokningen. Här måste orsak anges, för om det lämnas tomt och man klickar på "Avboka" knappen, visas en text som informerar om detta. Efter att avbokningen med orsak skickats in, uppdateras mötets status till "avbokat" och sendMail() funktionen anropas igen två gånger. Nu skickas en bekräftelse till bokaren angående avbokningens mottagande, samt ett meddelande till studiehandledaren för att informera om avbokningen. Tiden där bokningen varit insatt, syns fortfarande utåt som "BOKAD TID" och försvinner som sagt först då studiehandledaren raderar mötet.

4 Administrering

Studiehandledaren kan logga in med användarnamn och lösenord till bokningssystemet för att få tillgång till mera funktioner. Administreringen sker med samma användargränssnitt som bokningssystemet har för alla besökare, men flera alternativ uppstår.

Efter lyckad inloggning kan studiehandledaren bläddra månaderna bakåt och framåt med hjälp av länkarna Föregående och Nästa ovanför månadsvyn. När månaden byts, visas i veckovyn den veckan som månadens första dag hör till. Mötena som visas i veckovyn har nu mötets ärende inskrivna i rutorna i stället för "BOKAD TID" som visas för en vanlig besökare. Genom att klicka på ett möte öppnas en Fancybox ruta som visar mötets all information, alltså all den information som bokaren har angett. För övrigt visas alternativen "Godkänt" och "Inte godkänt" i form av radio-buttons via vilka en ny bokning kan godkännas. Under radio-button alternativen kommer tre knappar. "Radera", "Editera" och "Stäng".

"Radera" knappen raderar mötet och frigör dess tid för eventuell ny bokning, så länge som tiden i fråga är "grön", alltså att den är inom möjlig boknings tid. Då studiehandledaren raderar ett möte, tas det inte ännu bort från databasen, utan dess status bara ändras till "raderat" och bokningssystemet tolkar det som ett obefintligt möte.

"Editera" knappen öppnar mötets detaljer insatta i formulärs fält, alltså blir vyn den samma som då en bokare lägger till ett nytt möte, men nu är fälten färdigt ifyllda med bokningens information. Efter att eventuell ändring är gjord, informeras bokaren om detta med ett e-postmeddelande.

"Stäng" knappen finns alltid med i Fancybox rutan oberoende om en vanlig besökare eller studiehandledaren öppnat rutan och knappen stänger alltså fönstret, så man kommer tillbaka till huvudsidan. Som standard har Fancybox ett kryss uppe i högra hörnet, via vilken rutan går att stänga, men den har jag inaktiverad p.g.a. större kontroll över vad besökaren kan göra. Om Fancybox rutan stängs genast efter att ny bokning lagts till, avbryts funktionen som uppdaterar veckovyn. Detta gör att insatta mötet inte syns någonstans och därför inaktiverade jag möjligheten för besökaren att stänga fönstret på andra sätt än via egen knapp.

Vid klickning på ett visst möte, öppnas olika administrerings möjligheter för just det mötet, men för att göra massändringar kan studiehandledaren istället klicka på länken "Öppna administrerings fönstret". Denna knapp finns i administreringspanelen, ovanför "Logga ut" knappen. Via den öppnas igen ett nytt Fancybox fönster och här fås en överblick över alla mötena utan att man måste bläddra bakåt och framåt i månads- och veckovyn. Längst uppe i fönstret finns en meny med alternativen; "Visa alla möten", "Visa årets möten", "Visa raderade möten" och "Inställningar". "Visa alla möten" är som standard valt och här listas alla möten som finns i systemet i kronologisk ordning. Alla alternativen som visar möten, gör det på samma sätt, men enligt kriterierna som länkarnas namn anger. Mötena visas på egna rader och längst till vänster har varje rad en box via vilken ett krux kan bockas i. Ovanför finns en box "Välj alla" och genom att bocka för den, väljs alla rader. Genom att klicka på nytt i rutan, avmarkeras alla mötena. Under sista raden finns en "Radera" knapp som raderar alla de möten som valts, fast här gäller samma princip som med enskilda mötets "Radera" knapp, dvs. att mötets status bara ändras till "raderat".

För att sedan göra en slutlig radering, som raderar egentliga raden från databasen, skall "Visa raderade möten" alternativet väljas. Här visas nu de möten vars status är "raderat" på samma sätt som i de andra alternativen, med den skillnaden, att när man här raderar mötena, försvinner de helt och hållet från databasen.

"Inställningar" alternativet möjliggör editering av studiehandledarens egna uppgifter, som är namn, användarnamn och lösenord. Härifrån kan även administratorns e-postadress uppdateras. Denna adress används av bokningssystemet då e-post skickas automatiskt till studiehandledaren.

För övrigt kan härifrån anges buffert tiden som bestämmer skillnaden mellan aktuell tid och första möjliga tid då en bokning kan göras. Dessa variabler lagras i databasen i egen tabell.

5 Programmets tekniska uppbyggnad

Programmet är uppbyggt så att sidan förnyar sig så få gånger som möjligt. Detta betyder att webbläsaren befinner sig på en och samma fil och endast delar av sidan uppdateras. Dessa uppdateringar görs med hjälp av Ajax anrop, som hämtar på begäran information från skilda filer och den i ett bestämt element. Huvudsidan som webbläsaren visar när en besökare anländer till bokningssystemet befinner sig i index.php filen. Denna fil fungerar som en mall och innehåller en mängd olika element. Dessa element får sedan sitt innehåll från olika filer via Ajax anrop, som gör att hela sidan inte uppdateras, utan endast innehållet i elementen.

T.ex. fältet till vänster som visar månadsrutorna är inne i ett element som heter box_wrapper_left2. När studiehandledaren bläddrar via länkarna bakåt eller framåt i månaderna, anropar länkarna inte rakt en fil utan ett Javascript som sedan med hjälp av Ajax hämtar innehållet och lägger det innanför rätt element.

Personligen tycker jag om att programmet är uppbyggt så här, redan bara för att adressfältet visar alltid samma adress och så ser det mycket finare ut när inte hela sidan förnyar sig.

Men andra platser där Ajax används skulle inte kunna göras på annat sätt. Här menar jag rullgardinsmenyn för sluttiden då ny bokning görs. Den hämtar informationen om varje gång en starttid väljs.

Två månaders alla möten hämtas och läggs in i sessionsvariabeln. Dessa två månaderna är de som syns till vänster i månadsvyn, så när studiehandledaren bläddrar bakåt eller framåt bland månaderna, hämtas mötena varje gång på nytt från databasen för de två aktuella månaderna. Men när veckovyn uppdateras via klick på dagar i månadsvyn, hämtas valda veckans möten rakt från sessionsvariabeln.

5.1 Dynamisk layout

Programmet är konstruerat med dynamisk layout. Detta betyder att allt som syns på sidan har storlekar angivna i procent, så storleken på de olika vyerna är relativa till varandra samt till webbläsarens storlek. Så gott som alla delområden av sidan är uppbyggda med element som möjliggör att dela upp sidan i flera enskilda fält. Med detta element delas sidan först upp i större rektanglar, en för månadsvyn och en för veckovyn. Förutom dessa finns även en "header" och en "footer" som breder sig ut över hela sidans bredd. Elementens attribut anges med hjälp av "Style Sheets", som deklareras i en skild css fil. Bland annat anges elementens bredd och höjd och som sagt anges de med procenttal i detta projekt för att uppnå dynamisk layout. Totala ytan en webbläsare kan visa är 100% i både bredd och höjd. Månadsvyn har en bredd på 18% och veckovyns bredd är 75%. Mellan dessa två vyer och i vänstra och högra marginalen finns smala rektanglar som tillsammans har en bredd på 7%. De är osynliga och finns till för att finjustera de synliga vyernas position.

Dynamiska layouten används för att säkra att sidan visas korrekt oberoende om besökaren använder en 7 tums tablet eller en 60 tums TV skärm.

En annan aspekt som säkrar att bokningssystemet visas korrekt är en funktion som kontrollerar vilken webbläsare besökaren använder och enligt det väljer rätt css fil. Alla webbläsare tolkar css kod på lite olika sätt och därför måste koden skrivas skilt för varje webbläsare. Detta säkrar att bokningssystemet både fungerar och ser lika ut på så många enheter som möjligt. I framtiden är det dock meningen att alla webbläsare skall tyda kod på exakt samma sätt vilket kommer att underlättar webbutveckling en hel del.

5.2 Sessioner

En session skapas då webbläsaren anländer till bokningssystemet och den fungerar som en behållare med individuell information för varje enskild besökare. Fast bokningssystemet hoppar mellan olika filer, finns sessionsvariabeln alltid till förfogande och oberoende vilken fil som körs, kan den använda all information sessionsvariabeln innehåller.

Utan sessionsvariabeln, måste informationen hämtas på nytt varje gång en ny fil körs.

Bokningssystemet är byggt så att när en besökare anländer till sidan, kontrolleras först om sessionen redan finns eller inte. Om den inte finns, så skapas den. Sedan hämtas aktuella

och nästa månadens alla möten in till den. Alla filer som sedan behöver komma åt mötenas uppgifter tar den informationen rakt från sessionen. Behovet av att göra anrop till databasen minskar således och därför blir även systemet snabbare. Informationen om mötena under dessa två månader används i många olika syften, men besökaren får aldrig fram annan information än mötenas start- och sluttider.

Sessionsvariabeln som skapas är unik för varje enskild webbläsare som anländer till sidan. Den sparas endast på servern, vilket ur säkerhetssynvinkel är bra, för då har klienten ingen möjlighet att komma åt dess innehåll förutom via bokningssystemet.

Detta innebär att sessionen kan innehålla känslig data och ingen utomstående har tillgång till den.

Innehållet i sessionsvariabeln finns tillgängligt så länge som webbläsaren befinner sig vid bokningssystemet. Om webbläsaren stängs, raderas den session som var skapad specifikt för besökaren i fråga. Sessionen har även en inbyggd time-out funktion, som gör att variabeln raderas om den inte anropas inom utsatt tid. Så om en besökare är vid bokningssystemet och gör inga anrop under sessionens time-out tid, raderas sessionen, fast webbläsaren fortfarande visar sidan. Med tanke på sådana tillfällen, har jag skapat ett system som kontrollerar sessionens status vid anrop och ifall sessionen har gjort time-out, visas ett nytt fönster. Detta fönster informerar om att sessionen förfallit och att sidan kommer att laddas om efter att man klickat på ok knappen.

5.3 Filerna

index.php

Index.php fungerar som en initieringsfil och är den filen som webbläsarens adress alltid befinner sig vid. Filen inkluderar och anropar sedan andra filer och funktioner för att visa innehåll.

conf.php

Här finns olika konfigureringsvariabler och konstanter som t.ex. databasens inloggningsuppgifter. Även tidszonen, aktuella tiden samt vecko- och månadsnamnen genereras från denna fil.

class_lib.php

Denna fil innehåller hela programmets alla funktioner och klasser. Exempel på funktioner berättas mera om i kapitel 5.8 Funktioner.

head.php

Här anropas css filerna, jQuery filen samt Fancybox filen. Dessutom finns största delen av Javascript funktionerna här som med Ajax anropar andra php filer.

browser.php

Anropet på css i header filen görs egentligen via denna browser.php fil, som har en enkel men viktig uppgift, nämligen att känna igen besökarens webbläsare och enligt den bjuda ut rätt css fil. Detta måste göras, för att fast css baserar sig på standarder, tyder ändå webbläsarna Stylesheet element på olika sätt. För att sidan skall visas korrekt oberoende av vilken webbläsare besökaren använder, måste det finnas olika css filer.

displaymonth.php

Största delen av anrop till funktioner och hämtande av data sker i denna fil. Här initieras nyckelvariabler, som i många fall skapas i array format. T.ex. skapas här arrays som innehåller hela månaders alla datum och så hämtas här hela månaders alla möten från databasen. I slutet av filen anropas en funktion som ritar ut ett rutfält med en hel månads alla dagar.

month.php

Filen är en fortsättning till displaymonth.php och hjälper till med formandet av månadsfältet. Här görs även varje datum i månadsfältet till länkar för att visa valda dagens innehåll och samtidigt hela veckans alla dagar.

getweek.php

Denna fil ritar ut största delen av ytan som visas på huvudsidan, nämligen veckovyn. Filen hämtar aktuella veckans alla möten från sessionsvariabeln och ritar ut dem på sina rätta platser. Ett mötes start- och slutnivå räknas ut enligt totala mängden timmar, som visas per dag och placeras ut i relation till det. Varje möte som syns kan även klickas på. Detta anropar en Javascript funktion som i sin tur anropar Fancybox initieringen. Fancybox är ett skilt ”open source” projekt som visar innehåll innanför ett fönster, så att huvudsidan fortfarande syns i bakgrunden, men den blir mörkare så att nya fönstret markeras som det

aktiva. Fancybox beskrivs djupare i kapitel 5.7 Fancybox.

showmeeting.php

Showmeeting.php körs innanför Fancybox rutan och visar antingen ett befintligt mötes information eller ett formulär där möte kan läggas till. När ett befintligt möte visas, kan det via knappar raderas eller editeras och mötets status visas med radio buttons. Statusen kan vara godkänd eller inte godkänd. När möten läggs till, är de som standard inte godkända, utan studiehandledaren måste godkänna dem.

update_data.php

Då studiehandledaren öppnar ett möte och ändrar dess status från showmeeting.php, så anropas update_data.php filen med Ajax anrop. Här körs uppdateringen av status ändringen till databasen och ifall man valt att lägga till ett möte, läggs det till databasen också härifrån. Ändringar som görs till befintliga möten skickas per e-post till den personen som satt in mötet. Dessa e-post skickas även härifrån.

get_endtime.php

Då ett möte läggs till, visas alla möjliga sluttiderna i en rullgardinsmeny. Eftersom mötena inte kan överlappa varandra, måste sluttiden beakta befintliga bokade möten. Denna uträkning av sluttiden görs i get_endtime.php filen och den anropas med Ajax varje gång ett alternativ väljs från rullgardinsmenyn för starttider.

avbokning.php

När man har lagt till ett möte, skickas ett bekräftelsee-postmeddelande till adressen man angett. I detta e-post finns en länk via vilken avboking.php filen öppnas. Här kan den som klickat på länken avboka mötet genom att ange obligatorisk orsak till avbokningen.
Om avbokningen lyckas, skickas e-post om händelsen till studiehandledaren samt bokaren.

login.php

Denna fil har i sig endast inloggningsformuläret samt Javascript som kontrollerar fältens innehåll och skickar data vidare. Som standard är inloggningsfälten gömda och för att få fram dem anropas ett Javascript. Formuläret innehåller två fält; användarnamn och lösenord.

login_data.php

Hit skickas användarnamn och lösenord med hjälp av ajax anrop från inloggningsformuläret och sedan kontaktas databasen för att validera att inloggningsuppgifterna stämmer. Om användarnamn och lösenord är rätt angivna, uppdateras sessionsvariabeln med information att användaren är inloggad och har mera rättigheter än en vanlig besökare.

get_admin.php

Om inloggningen lyckas anropas getAdmin() funktionen, som kör med ajax anrop get_admin.php filen. Den aktiverar ett område som välkomnar inloggade användaren med texten "Välkommen Förnamn Efternamn" och så visas en länk "Öppna administrerings fönster".

admin.php

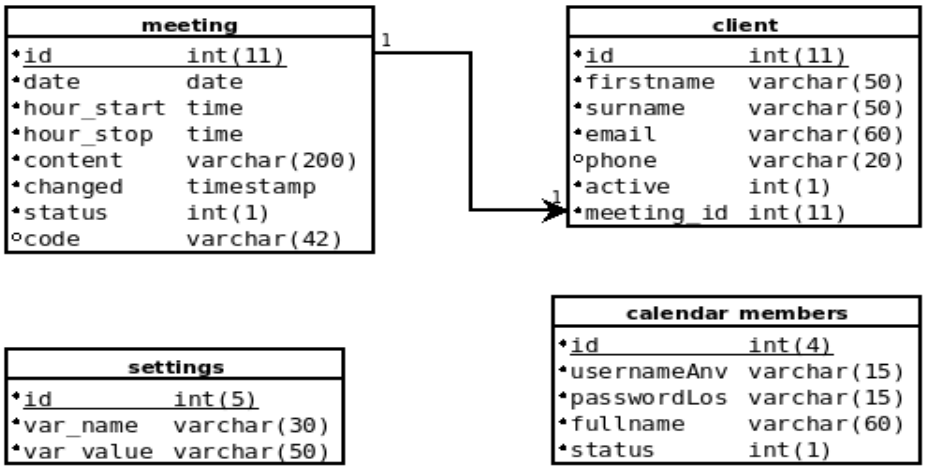
Denna fil körs på samma sätt som showmeeting.php, inne i ett Fancybox fönster. Här visas längst upp en meny med alternativen; Visa alla möten, Visa årets möten, Visa raderade möten och Inställningar. Som standard är "Visa alla möten" valt och här listas alla möten som finns i systemet i kronologisk ordning. De övriga alternativen visar enligt länkarnas namn mötena enligt olika kriterier förutom Inställningar som möjliggör editering av användarens uppgifter samt några andra inställningar.

5.4 Databasen

Databasen består av fyra tabeller; meeting, client, settings och calendar_members. Mötena sparas i meeting tabellen och bokarens uppgifter sparas i client tabellen. Eftersom studiehandledaren inte får upprätthålla ett personregister, lagras bokarens uppgifter alltid på nytt med de uppgifter som anges då ny bokning skapas. Ett möte i meeting tabellen är länkat till en bokare i client tabellen, vilket innebär att relationen är ett-till-ett.
Relationen skapas då en bokning läggs till. Först lagras mötets uppgifter i meeting tabellen varefter bokarens uppgifter samt mötets id läggs till i client tabellen. Denna relation visas med en pil mellan meeting och client tabellerna i figur 3.

Settings tabellen innehåller inställnings information, som studiehandledaren kan ändra. Dessa inställningar är: administratorns e-postadress, första och sista klockslagen i bokningssystemet och variabeln som anger skillnaden mellan aktuell tid och tidigaste möjligheten för att skapa en bokning.
Dessa inställningar kan studiehandledaren ändra via administrator fönstret.

Calendar_members tabellen innehåller användaruppgifterna för de användare som kan logga in till bokningssystemet som administratörer. I produktionsläge kommer tabellen att innehålla endast en användare, nämligen studiehandledaren, men under testperioden har där funnits en till användare, som jag själv loggat in med.



Figur 3: ER diagram av databasen

När studiehandledaren raderar ett möte, så skall också dess bokare raderas. För att göra denna funktion enklare, skapade jag även en "CASCADE DELETE" relation mellan meeting och client tabellerna. Detta innebär att databashanteraren upprätthåller en referensintegritet mellan dessa tabeller. Alltså när ett möte raderas, behöver man inte skilt radera bokaren, utan det sker automatiskt tack vare denna inbyggda relation.

5.5 Ajax

Ajax står för Asynchronous Javascript and XML. Det är en relativt ny teknik som bildats av att sätta ihop dessa redan länge existerande delarna:

1. HTML och CSS

2. Javascript

3. XML och XSLT

4. XMLHttpRequest

Ajax gör det möjligt att hämta data från servern till webbläsaren utan att hela sidan måste förnyas på nytt. Data transporteras således a-synkroniserat.

Webben har länge varit o-dynamisk och stel. Användare började kräva mera funktionalitet till de statiska webbsidorna och olika tekniker har under åren försökt lösa detta. Man har tidigare försökt få mera egenskaper till webbsidor med t.ex. Java och Flash, men dessa tekniker har krävt att användaren installerar skilda instoppar-program (plugins) till webbläsaren. Termen "Ajax" användes första gången i februari 2005 av utvecklaren Jesse James Garrett. Inte länge efter det började Google utveckla tekniken och därmed blev de en stor drivande kraft för av Ajax. I dag implementerar Google Ajax i så gott som alla sina tjänster. T.ex. används tekniken vid vanlig sökning på deras sida. När man börjar skiva in i sökfältet, ger sidan i real tid förslag på vad den tror du kommer att skriva. Efter varje tangent tryck uppdateras förslaget på nytt.

Utvecklare kan med hjälp av Ajax producera funktionalitet till webbapplikationer som förut var endast möjligt i skrivbordsapplikationer, alltså program gjorda för arbetsstationer.

En sida som inte använder Ajax hämtar hela sidan på nytt efter en förfrågning. Sidan som visas har också vanligen en egen URL, alltså ändras slutet i adressfältet med t.ex. "index.php?id=1". Klickar man på en annan sida, blir adressen kanske "index.php?id=2". Detta har inneburit att varje sida kan länkas rakt till antingen från en utomstående sida eller genom att bokmärka sidan. Med en sida som använder ajax är detta inte rakt möjligt vilket kan kräva vänjning från användarnas sida. Och eftersom förfrågningarna görs internt av sidan, fungerar inte "Bakåt" knappen heller alltid som man vant sig. Dessa saker måste tas i beaktande vid utveckling med ajax fast de är små (Asleson & Schutta, 2007, s. 1-23).

I bokningssystemet implementeras Ajax i så gott som alla anrop som hämtar data. Tanken är att sidan inte behöver förnyas i onödan, utan endast den information som hämtas skall

förnyas på sidan. Således när månaden byts, förnyas endast månadsvyn och när veckan byts, uppdateras endast veckovyn. Då ny bokning läggs till, hämtas sluttiderna varje gång på nytt då en starttid väljs. På det viset märks anropet bara med att alternativen i rullgardinsmenyn för sluttiderna uppdateras. Allt annat hålls oförändrat vilket gör användarupplevelsen behagligare.

5.6 Inloggning

Inloggningen sker genom att ange användarnamn och lösenord i fälten under månadsvyerna till vänster. Inloggningsformuläret är som standard gömt och det fås synligt genom att klicka på Admin länken längst nere i mitten. Figur 4 visar en del av bokningssystemets huvudsida, där inloggningsformuläret är synligt till vänster under månadsvyn och till höger syns en liten del av veckovyn.

Figur 4: Inloggningsformuläret samt delar av månads- och veckovyn

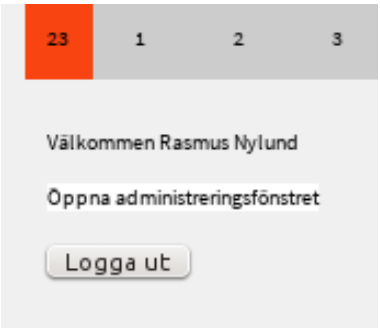
Inloggningsrutan skickar med Ajax anrop angivna användarnamnet och lösenordet till login_data.php varifrån kontakt tas till databasen för att kontrollera att de stämmer. Om de stämmer, läggs ett specifikt värde till sessionsvariabeln. Detta värde kontrolleras sedan vid varje fil som kräver inloggning. Om värdet stämmer överens, vet filen att användaren är inloggad och då får sidan visas.

Om användarnamnet och lösenordet inte är rätt angivna, berättar login_data.php om detta till login.php filen, och inloggningsformuläret visas fortsättningsvis.

När inloggningen har lyckats, visas en administreringspanel istället för inloggningsformuläret. I administreringspanelen visas ett välkomstmeddelande samt en länk via vilken administreringsfönstret öppnas. Längst ner finns en utloggningsknapp som tar kontakt till login_data.php filen på samma sätt som inloggningen gör. Men istället för att skicka inloggningsuppgifter, så skickas en variabel som berättar om utloggningen.

Detta tömmer bort värdet från sessionsvariabeln som indikerade att användaren är inloggad, varefter inloggningsformuläret visas igen.

Figur 5 visar administreringspanelen samt en del av månadsvyn.



Figur 5: Administreringspanel

5.7 Fancybox

Fancybox är ett system som möjliggör visandet av bilder och annat innehåll på webben på ett elegant sätt. Innehållet öppnas i ett eget fönster så att bakgrunden skuggas till för att betona Fancybox rutans innehåll. Många webbsidor visar sitt bildgalleri innehåll med hjälp av Fancybox.

Programvaran är utgivet under Creative Commons Attribution-NonCommercial licensen, som betyder att det får användas fritt i icke kommersiella ändamål.

I bokningssystemet använder jag Fancybox bl.a. då en ny bokning skapas och i samtliga fall använder jag iframe teknik i intagandet av data. På det viset fungerar sidorna innanför Fancyboxen självständigt.

5.8 Funktioner

I programmerings projekt används ofta funktioner, men de är inte nödvändiga. En funktion kan definieras som ett underprogram som fungerar självständigt. Om någonting i koden skall utföras många gånger i olika skeden av programmet, är det bra praxis att föra in denna kod i en funktion. Funktionen kan sedan enkelt anropas var som helst och på det viset behövs inte koden skrivas om varje gång den behövs, endast anropet måste göras. Funktioner brukar skapas så att de kan ta emot argument. Argumenten bearbetas sedan av

funktionen och slutresultatet returneras tillbaka dit anropet gjordes.

I bokningssystemet har jag skapat alla funktionerna i class_lib.php filen. Denna fil inkluderas sedan till de filer som har behov av funktionerna.

Exempel på en enkel funktion som används i bokningssystemet för att konvertera datum från engelsk standard till skandinavisk kan ses i kodexempel 2. När funktionen anropas med argument som t.ex. "2015-04-15", görs det på följande sätt: formatDate(”2015-04-15”); Detta anrop skulle returnera: "15.04.2015".

Kodexempel 2: Funktion som konverterar datum

```php
function formatDate($input)
{
    $pieces = explode("-", $input);
    $final = $pieces['2'] . "." . $pieces['1'] . "." . $pieces['0'];
    return $final;
}
```

Kodexempel 3 illustrerar funktionen som används för att uppdatera statusen på ett befintligt möte. Statusen berättar om mötet är godkänt, icke godkänt, borttaget eller avbokat. Till funktionen kan ges fyra olika parametrar. Id berättar identifikationsnumret för mötet och status berättar vilken status mötet skall få. De två andra parametrarna authC och reason anges bara då mötet avbokas via länken som skickas till bokaren. AuthC är mötets specifika kod och reason är orsaken till avbokningen som är ett obligatoriskt fält.

Kodexempel 3: Funktion som uppdaterar status för befintligt möte

```php
function UpdateData($id=0, $status=99, $authC=0, $reason="NULL")
//Denna funktion uppdaterar statusen för ett befintligt möte.
{
    global $db_host, $db_user, $db_pass, $db_name;
    date_default_timezone_set('Europe/Helsinki');

    if ( $status != 99 && $authC == 0) //Detta körs när statusen ändras av administratören
    {
        $stamp = date("Y-m-d H:i:s");
        $sql_query = "UPDATE meeting SET status=" . $status . ", changed='" .
        $stamp . "' WHERE id=" . $id;
    }
    else if ( $id == 0 && $authC != 0 ) //Detta körs om mötet avbokas med kod.
    //Orsaken till avbokningen läggs i content fältet.
    {
        $stamp = date("Y-m-d H:i:s");
        $sql_query = "UPDATE meeting SET status=" . $status . ", changed='" . $stamp .
        "', content='" . $reason . "' WHERE code='" . $authC . "' AND status<=2";

    }
    $Database = new Database($db_host, $db_user, $db_pass, $db_name);
    $Database->query($sql_query);
    $res = $Database->affected_rows;
    $Database->close();
    return $res;
}
```

5.9 Säkerhet

Bokningssystemet körs i Novias intranät, vilket betyder att endast personer med behörighet, alltså personal och studeranden, kommer åt systemet. Oberoende av detta, är bokningssystemet byggt så säkert som möjligt.

Studiehandledarens lösenord ligger krypterat i databasen vilket minskar möjligheten för en utomstående att ta reda på det. I många av de intrångsfall som sker, är det databasen som först knäcks. Om lösenorden är sparade i klartext, får gärningsmannen full tillgång till systemet. Därför skall lösenord alltid sparas i databasen i krypterad form.

Krypteringen sker med MD5, som anses vara en "en vägs" hash algoritm. Med detta menas att det inte finns någon möjlighet att dekryptera hash koden som krypterats med MD5, tillbaka till dess ursprung. (Gilmore, 2006, s. 529).

Krypteringen ändrar det egentliga lösenordet till en 32 tecken lång kod. Varje gång studiehandledaren loggar in, krypteras lösenordet och jämförs med kodstrengen i databasen för att kontrollera om lösenordet är korrekt eller inte.

Sql-injection är en metod som används för att olovligt bryta in sig till databaser. Denna metod fungerar inte mot bokningssystemet, eftersom databasförfrågningarna som körs, tar in endast kontrollerad data.

Förutom dessa säkerhetsåtgärder, har jag implementerat en kontroll i varje fil som anropas, som försäkrar att vissa specifika variabler skickas med i anropet. Om dessa saknas, tillåts inte körningen av filen. Detta är en metod som även minskar sannolikheten för intrång.

6 Testskede

Testskede har pågått under en längre period där jag först har bett några vänner bekanta sig med bokningssystemet. Före det har jag inte gett någon information åt dem om hur systemet fungerar, så deras testningar har bra simulerat hur studeranden reagerar vid bokningssystemets ibruktagning.

Det är alltid bra att ha utomstående att testa system, då man själv enkelt blir blind för eventuella problem ur en användares synvinkel. Testningen är en väsentlig del i programmering av system som kommer att ha många användare. Alla tänker och beter sig på olika sätt och därför är bokningssystemet skapat med stort fokus på användbarheten. Systemet skall var enkelt och logiskt att använda och möjligheten att göra fel skall vara så

liten som möjligt. Alla dessa aspekter har blivit bra genomgådda under testningen och enligt den feedback som kommit har jag korrigerat fel och andra estetiska problem. Meningen är ju ändå att för slutanvändarna göra ett så bra systemet som möjligt.

Användarna får i högre grad det systemet de önskar, om testarbetet lyckats (Eriksson, 2008, s 19).

6.1 Virtuell server som plattform för projektet

Under testskedet har bokningssystemet körts på en virtuell server som min vän och jag har skaffat tillsammans. Servertjänsten köper vi av företaget Digital Ocean som erbjuder snabba virtuella platformer avsedda framförallt för utvecklare. Vi prenumererar på deras billigaste lösning som innehåller 512MB arbetsminne, enkärning processor, 20GB lagringsutrymme på SSD disk och 1TB dataöverföring per månad. Detta paket kostar endast $5/mån. vilket vi tycker är förmånligt då vi dessutom har flera olika domäner på samma server. Digital Ocean har sin hårdvara placerad på olika platser världen om och vi valde att ha vår server i Amsterdam, eftersom det är närmast Finland av de alternativ som fanns att välja mellan.

På virtuella servern har vi installerat en serverversion av Ubuntu Linux och webbsidorna drivs med Apache, PHP5 och MySQL. Mera information om dessa fås i kapitel 2.

6.2 Mötet med studiehandledaren angående testningen

Efter testskedet med mina vänner, var nästa steg att få studiehandledaren med och testa. Med vännernas testningar var syftet att göra användargränssnittet så lätt att använda som möjligt ur en utomståendes synvinkel. Och eftersom jag inte gav inloggningsuppgifter åt dem, var deras användande i likhet med hur det kommer att vara för studeranden då bokningssystemet är i produktion.

För att komma igång med studiehandledarens testskede skapade jag ett administratörs konto åt henne och sedan hade vi ett möte där jag visade hur bokningssystemet fungerar.

Under mötet skapade vi en testbokning för att få en bokares perspektiv av systemet. Här visade jag hur befintliga möten tas i beaktande i val av start- och sluttid samt hur bokningssystemet visar felmeddelande om inte alla fält fylls i.

Efter att bokningen var skapad, gick vi igenom e-postmeddelandet som bokningssystemet skickar till både studiehandledaren och bokaren.

Sedan loggade vi in som administratör och gick igenom hur möten kan godkännas eller tas bort. Varje gång statusen för en bokning ändras mellan godkänd, inte godkänd och raderad, skickar bokningssystemet e-postmeddelande för att berätta om ändringen. Vi funderade på hur innehållet i dessa e-postmeddelanden borde se ut beroende av vilken bokningens status är.

Meddelandena innehöll i detta skede information om vad som skall behandlas under mötet, alltså den texten som bokaren skriver i "innehåll" fältet vid skapandet av bokningen. Studiehandledaren påpekade att denna information absolut inte får synas i något e-postmeddelande som bokningssystemet skickar ut. Hon förklarade att endast tiderna för mötet får synas, eftersom då finns ingen möjlighet för utomstående att ens av misstag kunna få reda på konfidentiell information som endast berör bokaren och studiehandledaren.

Vi gick även igenom andra förbättringsförslag och nya egenskaper som borde läggas till bokningssystemet. Vissa av dem var självklara, men som jag inte ännu hade hunnit skapa. Exempel på en sådan egenskap var möjligheten för studiehandledaren att byta sitt användarnamn och lösenord. Andra egenskaper var mindre självklara, men sådana som studiehandledaren behövde. Bl.a. hade hon behov av en egenskap som listar upp alla bokningar hon haft under året. När studiehandledaren är inloggad till bokningssystemet, kan hon bläddra bakåt och framåt i månadsvyn och på det viset kunna se en veckas alla bokningar i gången. Detta skulle dock vara ett mycket tidskrävande sätt att luska ut ett helt års alla möten.

En annan egenskap som fattades var möjligheten att avboka ett möte man ansökt om. Nu kunde endast studiehandledaren radera bokningar. Denna egenskap hade jag redan tidigare funderat att skulle vara bra att skapa. Jag berättade att smidigaste sättet skulle vara att via en länk i bekräftelse e-postmeddelandet kunna göra avbokningen.

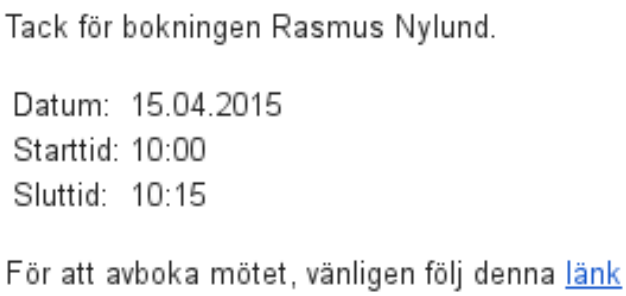
Administrering av inställningar som styr bokningssystemets funktionalitet, var även en egenskap jag velat skapa men som ännu fattades. Dessa inställningar används alltså redan av bokningssystemet, men de går inte att ändra via administrerings panelen. Inställningarna som skall kunna administreras är följande:

- Administratorns e-postadress, hit skickar bokningssystemet automatiskt e-post vid olika tillfällen.

- Start- och sluttiden som anger vilka tider är möjliga att boka på. Veckovyn ritas upp enligt dessa.
- Tidsfaktorn som anger tiden mellan aktuell tid och första möjliga tiden för att skapa en bokning. I veckovyn syns de tider som är möjliga att boka på med grön bakgrund.

6.3 Förbättringar och korrigeringar enligt möte

Möjligheten för bokaren att avboka ett möte gjorde jag som sagt genom att lägga med en länk i bekräftelse e-postmeddelandet. Länken kommer även med i e-postmeddelande som skickas till bokaren då studiehandledaren godkänner mötet. Länken visas enligt figur 4 som illustrerar bekräftelse e-postmeddelandet innehåll.



Figur 6: Bokningsbekräftelse e-postmeddelande

Länken innehåller en kod som genereras då bokningen skapas. Denna kod är ett slumpmässigt tal som skapas av aktuella tiden presenterad i mikrosekunder. Talet krypteras sedan till ett hashtal som är 32 tecken långt. Detta tal förlängs ännu genom att lägga till mötets datum i okrypterad form. Datumet representeras med 8 tecken, två för dag, två för månad och fyra för år. Så slutliga koden som lagras i databasen är 40 tecken lång och samma kod läggs med i e-postmeddelandet som skickas till bokaren. Avbokningslänken har koden inbakad i sig och då man klickar på den, öppnas webbläsaren till avbokning.php filen så att koden syns i adressfältet.

Nu visas ett enkelt formulär med endast ett fält; "Orsak till avbokning" och en knapp "Avboka".

Mötets datum som sattes med i koden finns med för att göra en första kontroll genast då

man anländer till avboknings formuläret. Om datumet är äldre än aktuella datumet, visas istället för formuläret texten "koden är felaktig eller föråldrad!". På detta viset kan man endast avboka ett möte som är i framtiden och denna kontroll behöver inte öppna upp kontakten till databasen.

Om man har rätt att avboka mötet, måste man fylla i orsak till avbokningen. När den är ifylld, klickar man på knappen "Avboka", varefter ett meddelande om lyckad avbokning blir synligt. Avbokningen skickar även automatiskt e-postmeddelande till bokaren och studiehandledaren angående händelsen. Fast mötet har blivit avbokat, syns det ännu som "BOKAD TID" för besökare. Detta betyder att ingen tid kan bokas på samma plats ännu. Platsen blir ledig först efter att studiehandledaren godkänt orsaken till avbokningen och raderat mötet.

Hanteringen av bokningssystemets inställningar samt listorna över alla bokningar lade jag ihop till en och samma administrerings fönster. Detta fönster öppnas från administeringspanelen och mera information om detta fås i kapitel 4 Administrering.

7 Avslutning

Det finns små korrigeringar som ännu måste göras åt bokningssystemet före det är helt färdigt. Genast när dessa korrigeringar är åtgärdade skall systemet publiceras.

Jag har fått instruktioner att skicka alla filer samt databasen till Yrkeshögskolan Novias IT personal, som sedan installerar bokningssystemet på en webbserver som körs i intranätet.

Utvecklingen av bokningssystemet har varit lärorikt och jag hoppas att det kommer till stor nytta för studiehandledaren och de studeranden.

Bokningssystemet kan ju också vidareutvecklas med nya egenskaper. Att koppla bokningssystemet ihop med Google Calendar var en egenskap jag funderade på, men som aldrig blev av. Kanske någon annan med kunskaper i PHP får möjlighet att fortsätta vidare på projektet någon gång i framtiden.

Jag vill till slut tacka Yrkeshögskolan Novia samt studiehandledaren för denna möjlighet att få utveckla ett intressant bokningssystem.

8 Källförteckning

Asleson, R & Schutta, N T. (2007). *Ajax – Tehokas hallinta*. Jyväskylä: Gummerus Kirjapaino Oy.

Augustsson, M & Folkesson, S. (2010). *Webbprogrammering med PHP*. Sundbyberg: Pagina Förlags AB.

Eriksson, U. 2008. *Test och kvalitetssäkring av IT-system*. (2. uppl.) Malmö: Hombergs i Malmö AB.

Gilmore, W J. (2006). *Beginning PHP and MySQL 5: From Novice to Professional*. (2. uppl.) Berkeley: Apress.

Novia.fi. (2014), Studiehandledning [Online] https://intra.novia.fi/studier/studerandetjanster/studiehandledning [Hämtat 27.04.2015]

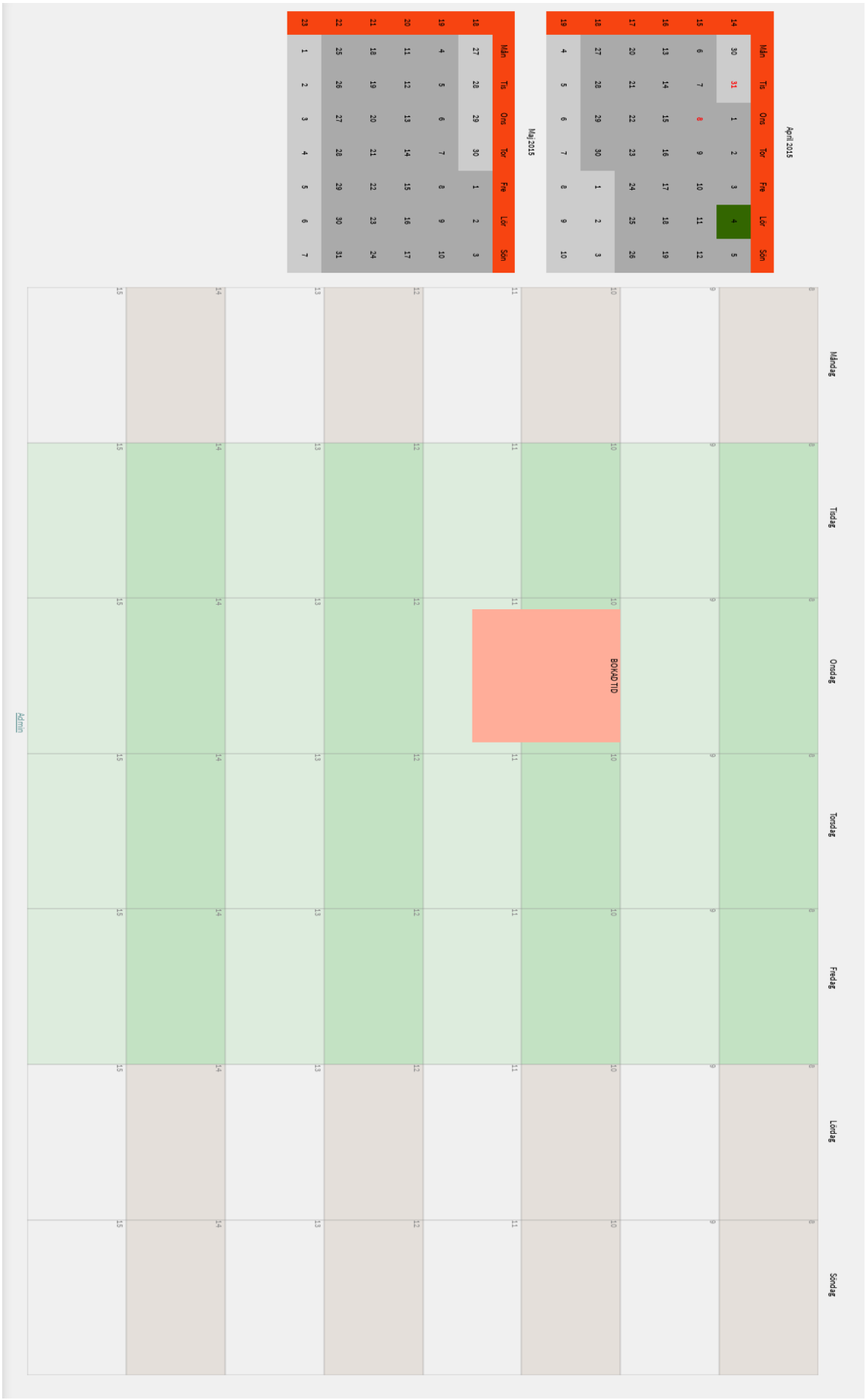
Wikipedia (2015), Apache HTTP Server [Online] http://en.wikipedia.org/wiki/Apache_HTTP_Server [Hämtat 03.05.2015]

Wikipedia (2014), jQuery [Online] http://sv.wikipedia.org/wiki/Jquery [Hämtat 03.05.2015]

Wikipedia (2014), MySQL [Online] http://sv.wikipedia.org/wiki/MySQL [Hämtat 03.05.2015]

Wikipedia (2015), PHP [Online] http://sv.wikipedia.org/wiki/PHP [Hämtat 03.05.2015]

Bilaga 1


April 2015
Mån Tis Ons Tor Fre Lör Sön
Maj 2015
Mån Tis Ons Tor Fre Lör Sön
Måndag
Tisdag
Onsdag
Torsdag
Fredag
Lördag
Söndag
BOKAD TID
Admin