



CHALMERS

Saksat.se

- En tentamensdatabas

Examensarbete inom Data- och Informationsteknik

JONATHAN GRANSTRÖM
JESPER KJELLQVIST

EXAMENSARBETE

**En tentamensdatabas för Sahlgrenska akademins
studentkår**

Hur en webbapplikation för uppladdning och visning av
tentamenstester kan utvecklas

JONATHAN GRANSTRÖM
JESPER KJELLQVIST

Institutionen för Data- och Informationsteknik
CHALMERS TEKNISKA HÖGSKOLA

Göteborg 2016

En tentamensdatabas för Sahlgrenska akademins studentkår

Hur en webbapplikation för uppladdning och visning av tentamenstester kan utvecklas

JONATHAN GRANSTRÖM

JESPER KJELLQVIST

© JONATHAN GRANSTRÖM, JESPER KJELLQVIST, 2016

Examinator: Christer Carlsson

Institutionen för Data- och Informationsteknik

Chalmers Tekniska Högskola

412 96 Göteborg

Telefon: 031-772 1000

The Author grants to Chalmers University of Technology and University of Gothenburg the non-exclusive right to publish the Work electronically and in a non-commercial purpose make it accessible on the Internet.

The Author warrants that he/she is the author to the Work, and warrants that the Work does not contain text, pictures or other material that violates copyright law.

The Author shall, when transferring the rights of the Work to a third party (for example a publisher or a company), acknowledge the third party about this agreement. If the Author has signed a copyright agreement with a third party regarding the Work, the Author warrants hereby that he/she has obtained any necessary permission from this third party to let Chalmers University of Technology and University of Gothenburg store the Work electronically and make it accessible on the Internet.

Institutionen för Data- och Informationsteknik
Göteborg 2016

FÖRORD

Detta examensarbete med tillhörande rapport gjordes som det sista momentet i Dataingenjörsutbildningen (180 högskolepoäng) på Chalmers Tekniska Högskola under våren 2016. Arbetet omfattar 15 högskolepoäng och cirka 10 veckors arbetstid.

Vi vill gärna passa på att tacka Timmy Kjellqvist och Samuel Nilsson från Sahlgrenska Akademiens Studentkår som har gjort detta examensarbete möjligt, samt varit engagerade och motiverande genom hela arbetet

Vi vill även ge ett stort tack till Joachim von Hacht som har hjälpt oss utforma vår rapport och alltid varit tillgänglig under arbetets gång för att svara på frågor och handleda.

SAMMANFATTNING

Sahlgrenska Akademiens Studentkår, förkortat SAKS, har länge velat skapa en tentamensdatabas för Sahlgrenska Akademiens studenter. Målet med arbetet var att skapa en webbaserad applikation för att göra hantering och tillgång av tentamina enklare för SAKS och Sahlgrenska Akademiens studenter. Ett insticksprogram för WordPress har skapats, som ger funktionalitet för att lägga upp kategoriserade tentamenstester och visa dem på en webbplats. WordPress används då det efterfrågades av SAKS. Arbetet genomfördes i iterationer, där grundläggande krav först uppfylldes.

Nyckelord: WordPress, webbutveckling, databas, webbapplikation

ABSTRACT

The students' union of Sahlgrenska Academy, abbreviated SAKS, has long wanted to create an exam databas for the students at the Sahlgrenska Academy. The goal of the project was to create a web-based application to make management and accessibility of exams easier for SAKS and for the students at the Sahlgrenska Academy respectively. A plugin for WordPress was created, which provides the functionality to upload categorized exams theses and display them on a website. WordPress was used at the request of SAKS. The project was carried out in iterations, where basic requirements was first fulfilled.

This report is written in Swedish.

INNEHÅLLSFÖRTECKNING

1	INLEDNING	1
1.1	Bakgrund	1
1.2	Syfte	2
1.3	Mål	2
1.4	Avgränsningar	2
2	METOD	3
3	TEKNISK BAKGRUND	4
3.1	Öppen källkod	4
3.2	Relationsmodellen	4
3.3	RDBMS	4
3.3.1	SQL	5
3.3.2	MySQL	6
3.3.3	MariaDB	6
3.3.4	phpMyAdmin	6
3.4	HTTP	6
3.4.1	Apache	7
3.5	Webbsida	7
3.6	PHP	8
3.7	JavaScript	9
3.8	Webbhotell	9
3.9	Insticksprogram	9
3.10	Tema	9
3.11	WordPress	9
3.12	Event-driven arkitektur	10
3.13	Mjukvarustack	10
3.13.1	XAMPP	10
4	KRAVSPECIFIKATION	11
5	ANALYS AV EXISTERANDE WEBBPLATS	13
6	DESIGN	14
6.1	Databasdesign	14
6.2	Grafiskt användargränssnitt	16
6.3	Arkitekturmönster och filstruktur	16
7	IMPLEMENTATION	17
7.1	Utvecklingsverktyg och arbetssätt	17
7.1.1	Lokal utvecklingsmiljö	17
7.1.2	Textredigeringsprogram	17
7.1.3	Utvecklingsdator	17
7.2	Databasen	18
7.3	Insticksprogrammet	19
7.3.1	Händelsehantering	19
7.4	Implementation av grafiskt användargränssnitt	22
7.5	Säkerhet	23

8	RESULTAT	25
8.1	Applikationen	25
8.1.1	Administrativt	25
8.1.2	Webbplatsen	27
8.2	Slutgiltiga resultat	28
8.3	Utvärdering av verktyg	28
8.4	Teknikens roll i samhället	28
9	SLUTSATS	29
9.1	Framtida inriktningar	29
	REFERENSER	30
	BILAGOR	32
	Bilaga A: Initial tidsplan	32

1 INLEDNING

1.1 Bakgrund

Sahlgrenska Akademiens Studentkår, SAKS, är en idéell organisation vilken representerar alla studenter vid Sahlgrenska Akademin, en fakultet inom Göteborgs universitet. SAKS arbetar för att öka utbildningskvaliteten, öka gemenskapen mellan utbildningarna och för en mer socialt givande studietid. Några exempel på aktiviteter SAKS har hand om är: idrottsaktiviteter, filmkvällar, spex och pubar. SAKS erbjuder även stöd och hjälp för studenter som råkar ut för problem i eller kring sina utbildningar. SAKS har dessutom möjligheter för att medverka till att förbättra utbildningarnas arbetsmiljöer.

En tentamen är ett tidsbegränsat och betyggrundande examinationsprov som ges vid slutet av en kurs på en högskola eller ett universitet. Den fungerar som ett stickprov på innehållet av kursen där studenten redovisar sina kunskaper. En tentamen kräver alltid att studenten är väl förberedd för att lyckas bli godkänd. En metod för att förbereda sig är att studera tidigare relevanta tentamenstester. Enligt offentlighetsprincipen har varje enskild person rätt att få ta del av allmänna handlingar, denna principen gäller på alla statliga högskolor. Sahlgrenska Akademin är en fakultet i ett svenskt statlig universitet. Därför faller dess tentamina under offentlighetsprincipen och ska finnas tillgänglig för alla studenter att ta del av.

I dagsläget måste en student på Sahlgrenska Akademin vända sig till studentexpeditionen för att få en gammal tentamen utlämnad. Detta är en tidskrävande och kostsam process för expeditionen men även studenterna. På grund av ekonomiska och/eller geografiska svårigheter kan vissa studenter inte hämta ut tidigare tentamina. Utöver detta måste studenterna själva bekosta utskriften. Tillgängligheten till tidigare tentamina varierar dessutom mellan de olika kurserna och mellan programmen. Det är inte alltid som en tentamen kan återfinnas. SAKS anser att det nuvarande systemet är krångligt, föråldrat och bryter mot SAKS målsättning att alla studenter skall ges samma förutsättningar. Frågan om tillgång till tentamina på Sahlgrenska Akademiens utbildningar har även fått en del uppmärksamhet utanför universitetets väggar[1]. Detta har ökat kraven på ett modernare och smidigare system.

SAKS har själva försökt utveckla ett system där studenter kan hämta ner gamla tentamenstester över Internet. Systemet är inte särskilt välutvecklat och löser inte ovan nämnda problem. De använder sig av Google Drive för att lagra gamla tentamenstester och en webbplats på ett webbhotell för att göra dem tillgängliga för nerladdning. Systemet skapar mycket administrativt arbete för kåren och gör det krångligt för studenterna att hämta ner tentamina. SAKS är inte nöjda med den nuvarande situationen.

1.2 Syfte

Arbetet syftar till att skapa ett effektivare system för hantering av, och tillgång till tentamenster.

1.3 Mål

Arbetets mål är att implementera en webbaserad applikation som löser SAKS problem med tentamenshantering.

1.4 Avgränsningar

Arbetet kommer endast involvera utbildningsprogrammen inom odontologi och hantera tentamina från deras kurser. Detta för att SAKS ska kunna testa systemet och senare besluta om de vill skala upp det till att omfatta andra utbildningsprogram på Sahlgrenska Akademin.

För att avgöra om systemet är effektivare krävs en uppföljande användarundersökning. På grund av tidsbrist kan detta inte göras. Istället kommer en skattning av effektiviteten att göras genom att jämföra det nya systemet mot det äldre.

2 METOD

Efter en genomgång med handledare och intern diskussion kom det fram till att arbetet kunde delas upp i sex större delmoment:

- Instudering av befintliga system
- Val av utvecklingsverktyg och programvara
- Analys av arbetsflöde
- Implementation av applikationen
- Sammanställning av manualer
- Skattning av systemets effektivitet

Arbetet utfördes delvis genom parprogrammering. Detta innebär att två personer arbetar tillsammans vid samma dator. En person skriver kod och den andra kontrollerar koden. Fördelar med denna metoden är till exempel att fel i koden är lättare att upptäcka och att parets delade kunskaper kommer till användning. Ett problem med denna metod är att dess framgång beror på hur bra paret kommer överens med varandra. En annan metod som användes delvis var kodgranskning. En person skrev kod självständigt och förklarade därefter koden stegvis för den andre personen. Detta för att upptäcka fel i koden. En fördel med denna metod är till exempel att framsteg inte uppehålls av en person med mindre kunskaper. Ett problem med denna metod är att en person kan fastna i sitt sätt att utveckla och tänker inte annorlunda.

Under början av projektet användes den första metoden. Senare under arbetets gång ersattes den med den sistnämnda metoden.

3 TEKNISK BAKGRUND

3.1 Öppen källkod

Öppen källkod (eng. open source) är en term som används för att beskriva programvara vars källkod är tillgänglig att läsa, använda, modifiera eller distribuera av vem som helst. Öppen källkod innebär även att licensen för programvaran är kostnads- och royaltyfri[2]. All programvara i projektet använder öppen källkod.

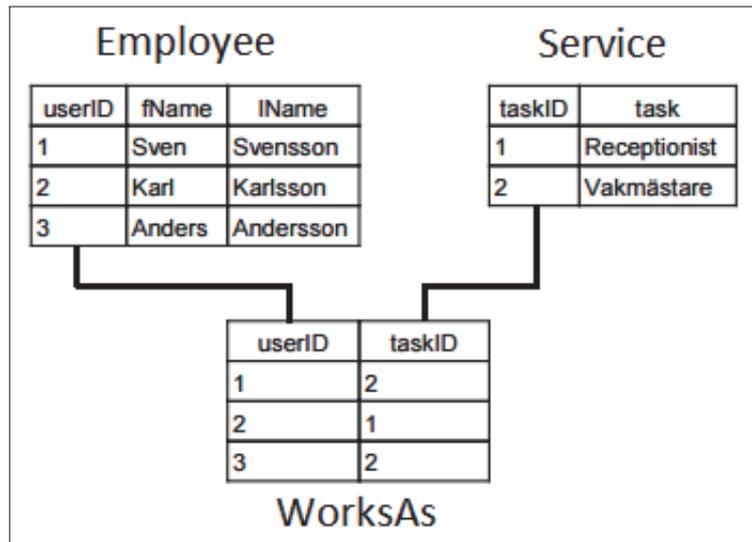
3.2 Relationsmodellen

Relationsmodellen är en teoretisk modell över hur data kan organiseras och användas i en databas. Modellen använder en struktur baserad på predikatlogik och mängdteori. Datan lagras i tupler (eng. tuples). En tupel är en ordnad mängd av värden. En uppsättning av tupler organiseras i vad som kallas en relation. Detta sätt att lagra data leder till en mindre risk för redundans i datan.

3.3 RDBMS

Ett Relational Database Management System (RDBMS) är ett databashanteringssystem som baseras på relationsmodellen. Idag är RDBMS den helt dominerande datalagringstekniken [3].

I ett RDBMS avbildas mängder som tabeller och tupler som rader i tabeller. Raderna kallas även poster. Kolumnerna i en tabell lagrar individuella element för varje post. Varje tabell och kolumn ges namn för att kunna identifiera vilken data som lagras. Namnen används för att kunna referera till data. Dessutom måste varje kolumn innehålla samma typ av data. Varje rad har en eller flera kolumner vars värden bildar en unik identifierade nyckel för posten. Data från olika tabellerna kan sammanlänkas genom att en nyckel för en tabell lagras i en annan tabell. Man kan även slå ihop nycklar från olika tabeller i en associativ tabell.

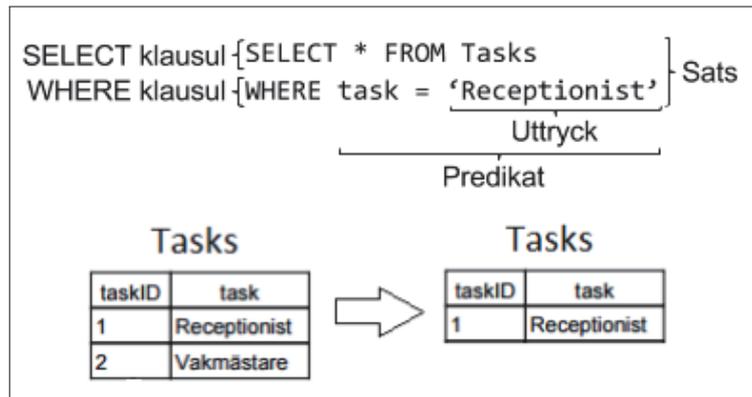


Figur 3.1: Exempel på en RDBMS struktur.

I Figur 3.1 har två vanliga tabeller och en associativ tabell avbildats, Employee, Service och WorksAs. I Employee lagras anställda och deras information. I Service lagras arbetsuppgifter. Tabellerna har varsin respektive kolumn som nyckel, userID och taskID. I den associativa tabellen WorksAs lagras nycklar från de två andra tabellerna, för att tilldela de anställda arbetsuppgifter. Enligt den associativa tabellen har till exempel Sven Svensson (userID 1) vakmästare som arbetsuppgift (taskID 2).

3.3.1 SQL

Sökspråket (eng. query language) Structured Query Language (SQL) används tillsammans med en RDBMS för att hantera och söka information i en databas. SQL är en internationell standard, ISO/IEC 9075-1:2011[4]. SQL består av tre delspråk: Datadefinition-, manipulation- och ett kontrollspråk. De tre språken används för att definiera databasens struktur, hantera data i databasen och hantera åtkomstbehörighet.



Figur 3.2: Exempel på en SQL-sats och dess resultat.

I Figur 3.2 visas en sökning i en databas med SQL. SELECT-klausulen (eng. clause) används för att specificera att data ska väljas. Ett asterisk innebär att data från alla kolumner i en tabell ska väljas. FROM specificerar att datan ska returneras från tabellen "Tasks". Den sista klausulen, WHERE, begränsar vilka rader från tabellen "Tasks" som ska returneras. Slutresultatet av sökningarna hämtar all data för arbetsuppgiften "Receptionist" från tabellen "Tasks".

3.3.2 MySQL

MySQL är ett RDBMS, det populäraste i världen [5]. Några av anledningarna att använda MySQL som RDBMS är dess prestanda och lätthet att anpassa efter lagringsbehov.

3.3.3 MariaDB

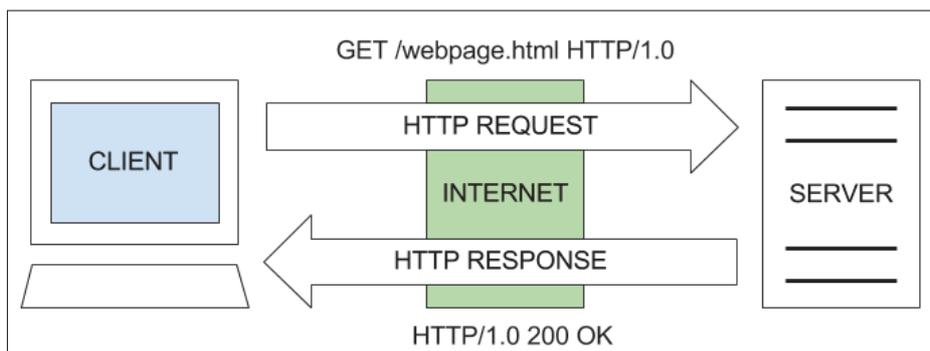
MariaDB är en vidareutveckling av MySQL. Idag används det av företag som Wikipedia och Google [6] [7]. MariaDB är kompatibelt med MySQL [8].

3.3.4 phpMyAdmin

Det webbaserade administrativa verktyget phpMyAdmin utvecklades för att hantera MySQL och MariaDB. Verktöget har ett grafiskt användargränssnitt för hantering av databaser. I gränssnittet finns verktyg för operationer som till exempel lägga till data genom "drag and drop". Dessutom finns möjligheten att skriva SQL-satser för att hantera och söka data i en databas[9].

3.4 HTTP

Datakommunikationsprotokollet Hypertext Transfer Protocol (HTTP) används för kommunikation och överföring av webbresurser mellan en webbserver och en klient, till exempel en webbläsare. En webbserver är en serverdator som kör ett webbserverprogram och har som primär funktion att skicka, lagra eller hantera webbsidor åt klienter. HTTP använder en förfrågan/svarsmodell för att fråga efter och skicka resurser, till exempel HTML-filer. HTTP är ett tillståndslöst protokoll. Det innebär att HTTP inte kräver att en webbserver behåller information om en klient över flera förfrågningar[10] [11].



Figur 3.3: En klient skickar en förfrågan om en resurs.

I Figur 3.3 kommunicerar en användare med en webbserver och begär tillgång till en resurs via ett HTTP-meddelande. Webbservern svarar med ett HTTP-statusmeddelande. Svaret innehåller en statuskod och den efterfrågade resursen.

3.4.1 Apache

Apache är ett webbserverprogram [12]. I en undersökning gjord i mars 2016 var Apache det populäraste alternativet för webbserverprogram med ca 33% av marknadsandelen [13].

3.5 Webb sida

En webbsida är ett textdokument som kan läsas och renderas (visas) av en webbläsare. En samling av flera relaterade webbsidor kallas en webbplats (eng. website). För att skapa webbsidor används märkspråk (eng. markup language) som till exempel HyperText Markup Language (HTML)[14]. HTML skrivs i HTML-dokument (.html). En webbläsare läser dokumentet, tolkar koden och omvandlar den till en läsbar webbsida som visas på en bildskärm. I Figur 3.4 visas kod för att skapa en enkel webbsida.



Figur 3.4: Exempel på HTML-kod och dess resultat.

CSS är ett komplement till HTML som beskriver utseendet och utplaceringen (eng. layout) av elementen i HTML-dokument. CSS kan även justera presentationen efter vilken typ av medium dokumentet ska skickas till. Att definiera utseendet för element direkt i HTML-dokument är en tidskrävande process, speciellt för webbplatser med många webbsidor. CSS löser detta problem. Koden skrivs istället i ett separat CSS-dokument (.css) som sedan refereras till i relevanta HTML-dokument. På så sätt kan utseendet för en hel webbplats ändras genom endast en fil [14]. I Figur 3.5 visas CSS-koden som ändrar utseendet på webbsidan i Figur 3.4. Rubriken får en ny font, textstorlek och blå textfärg. Den underliggande paragrafen får endast en ny textfärg, röd.



Figur 3.5: Exempel på CSS-kod och hur den påverkar utseendet.

3.6 PHP

PHP, en rekursiv akronym för “PHP: Hypertext Preprocessor”, är ett serverskriptspråk. PHP används för att skapa dynamiskt innehåll i webbsidor, till exempel olika formulär. PHP-kod kan läggas in i ett HTML-dokument eller skrivas i ett separat PHP-dokument (.php). Koden exekveras på en webbserver. Resultatet av exekveringen skickas tillbaka till klienten, till exempel en webbläsare, som HTML-kod [15]. Detta innebär att klienten inte ser PHP-koden utan endast den tillbakaskickade HTML-koden. I Figur 3.6 visas PHP-kod som ger webbsidan i Figur 3.4 dynamiskt innehåll, nämligen dagens datum.



Figur 3.6: Exempel på PHP-kod och det resulterande dynamiska innehållet.

3.7 JavaScript

JavaScript är ett objektbaserat skriptspråk som används för att göra webbsidor mer interaktiva. Språket körs på besökarens dator och kräver inte konstanta nedladdningar från webbsidan för att fungera. JavaScript används ofta tillsammans med HTML vars statiska definitioner av webbsidan kompletteras av JavaScripts dynamiska funktioner[16].

3.8 Webbhotell

Ett webbhotell (eng. web host service) är en webbservertjänst som gör det möjligt för användare att göra sin webbplats tillgänglig över webben (eng. World Wide Web). Tjänsten ger användare utrymme på webbhotellets webbserver. På så sätt behöver inte användaren installera och drifva en egen webbserver. Även andra möjliga tilläggstjänster kan fås genom ett webbhotell, till exempel säkerhetskopiering av data, databaser och trafikstatistik.

3.9 Insticksprogram

Ett insticksprogram (eng. plugin) ger icke fristående program ytterligare funktionalitet. Ett exempel på insticksprogram som används i applikationer är reklamblockerare i webbläsare. En bra anledning att använda insticksprogram är att funktionalitet kan tas bort eller ändras utan att själva huvudprogrammet påverkas. Arbetet skrevs som ett insticksprogram för WordPress.

3.10 Tema

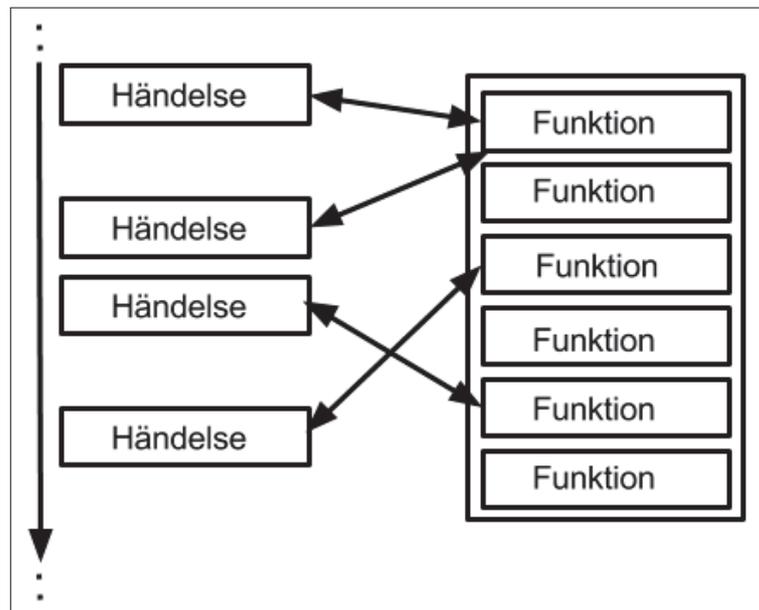
Inom datorgrafik är ett tema (eng. theme) en färdig kombination av grafiska element för att ändra stilen på ett användargränssnitt. Exempel på element är bild-, ljud- eller stilmallsfiler. Att använda färdiga teman innebär att tid inte måste spenderas med stilmall-, HTML- och PHP-filer. För arbetet används ett färdigt WordPress-tema.

3.11 WordPress

WordPress är ett webbaserat verktyg för att skapa webbplatser. WordPress riktar sig huvudsakligen mot användare som saknar erfarenhet av att utveckla en webbplats, men fungerar även som en plattform för utvecklare med mer erfarenhet[17]. Innehåll på webbplatsen för användare att ta del av, till exempel recensioner, läggs till genom innehållsinlägg. Dessa inlägg skapas av webbplatsens administratör. WordPress har stöd för färdiga teman samt insticksprogram[18] [19]. Även egenutvecklade insticksprogram och teman kan användas. På så sätt påverkas inte de kärnfiler som WordPress byggs upp av.

3.12 Event-driven arkitektur

Event-driven arkitektur (EDA) är ett mjukvaruarkitekturmönster. Applikationer som använder detta mönstret utfärdar eller lyssnar efter händelser (eng. events). Om en applikations status ändras, till exempel information uppdateras eller tas bort, så skickar den ut en händelse om förändringen. En annan applikation som lyssnar efter händelsen reagerar och utför sin uppgift. EDA är speciellt användbart när låg koppling, det vill säga beroende, mellan applikationer behövs.



Figur 3.7: Ett exempel på ett EDA mönster.

I Figur 3.7 sker olika händelser i en applikation under ett händelseförlopp. Funktioner är kopplade till dessa händelser och reagerar när dessa sker. När en händelse sker så utför de kopplade funktionerna sina respektive uppgifter. Händelseförloppet fortsätter därefter.

3.13 Mjukvarustack

En mjukvarustack är en uppsättning av programvara som tillsammans skapar en komplett plattform för en applikation. En mjukvarustack inom webbutveckling bör innehålla fyra grundläggande komponenter: ett operativsystem, en webserver, en databas och ett (eller flera) programmeringsspråk.

3.13.1 XAMPP

XAMPP är en mjukvarustack för utveckling av webbapplikationer. XAMPP står för “X (“cross”-plattform), Apache, MySQL, PHP och Perl” [20]. XAMPP kan installeras på datorer som kör antingen Linux, Windows eller OS X[21]. Den lokala Apache-webbservern kan förutom att serva statiskt innehåll även exekvera PHP-kod. MySQL ger möjlighet att använda en databas. phpMyAdmin används för att hantera databasen. XAMPP är speciellt användbart vid utveckling av en webbplats, då webbservern kan användas i testnings- och utvecklingsyfte.

4 KRAVSPECIFIKATION

Vid ett möte med handledare och styrelsemedlemmar från SAKS specificerades olika krav och önskemål. Dessa sammanställdes till en kravspecifikation.

SAKS bestämde att ett webbhotell skulle användas. Anledningen är att SAKS inte kan garantera kontinuerlig och tillförlitlig hantering, underhåll och drift av en egen webbserver. SAKS har i andra projekt använt WordPress och anser att det är ett enkelt verktyg för dem att arbeta med. Därför ville SAKS att webbapplikationen skulle utvecklas för att användas med WordPress.

SAKS ville huvudsakligen att arbetet ska ge en administratör möjligheten att lägga upp tentamina respektive svar på webbplatsen. Till en början ville de endast implementera utbildningsprogrammen inom odontologi, det vill säga tandläkare, tandsköterska och tandhygienist. Om webbplatsen fungera utifrån deras förväntningar kan stöd för de andra utbildningsprogrammen på Sahlgrenska Akademin implementeras senare. Administrativa verktyg ska finnas tillgänglig som gör det möjligt att lägga upp, ta bort, eller ändra i redan upplagd tentamina. Möjligheten att tilldela tentamen som ska läggas upp olika attribut ska finnas, till exempel kurskod, utbildningsprogram och termin. Ett önskemål från SAKS var att införa en kontroll som gör att endast elever på Sahlgrenska Akademin kan ta del av tentamina på webbplatsen. De efterfrågade därför ett sätt för administratören att godkänna användare innan de registreras på webbplatsen.

Användare av webbplatsen ska kunna hitta, visa och hämta ner tentamina. En söksida där en användare kan navigera en trädstruktur för att hitta en tentamen ska finnas. En alternativ sökmetod ger möjligheten att hitta tentamen efter olika kriterier, till exempel kurskod, program, lärare, termin eller datum. Ett önskemål var att implementera kommentarsfält på webbplatsen där användare kan dela med sig av sina åsikter. Ett uppröstningssystem ser till så att de högst uppröstade kommentarerna visas först i kommentarsfältet. Baserat på uppröstningssystemet kom idén om användarprofiler. På en sida ska en användare kunna se sin profil, information och hur många röstpoäng som införskaffats genom kommentarer. Information i en användares profil, till exempel program eller årskurs, ska kunna användas för att ge förslag på relevant tentamen på söksidan.

Tekniska krav:

- WordPress
- Webbhotell

Administrativa krav (tentamenshanteringen):

- Grundläggande krav:
 - Lägga till, ändra, ta bort tentamenstester.
 - Lägga till, ändra, ta bort svarsdokument.
- Förslag:
 - Möjligheten att godkänna/kontrollera registrerande användare.

Webbplatsen:

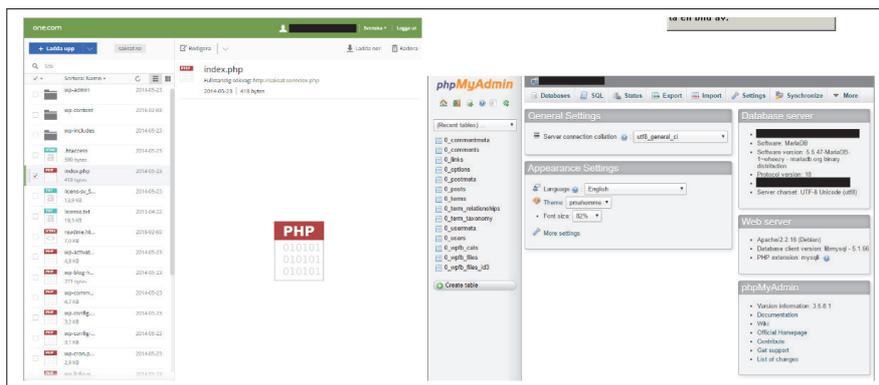
- Grundläggande krav:
 - En sida där sökning genom en trädstruktur efter en specifik tentamen kan göras.
 - En sida där sökning efter specifik tentamen baserad på olika kriterium kan göras.
- Förslag:
 - Endast inloggade/tillåtna användare kan se tentamina.
 - Ett kommentarfält för varje tentamenstest och svarsdokument.
 - Uppröstning av kommentarer.
 - Användarprofiler.
 - En förslagssida där tentamina baserad på användarprofil visas.

Arbetet delas upp i två delar: Utvecklingen av de administrativa verktygen som sköter om uppladdning, redigering och hantering av tentamina (tentamenshanteringen) och utveckling av webbplatsen som gör det möjligt för användare att ta del av upplagd tentamina.

5 ANALYS AV EXISTERANDE WEBBPLATS

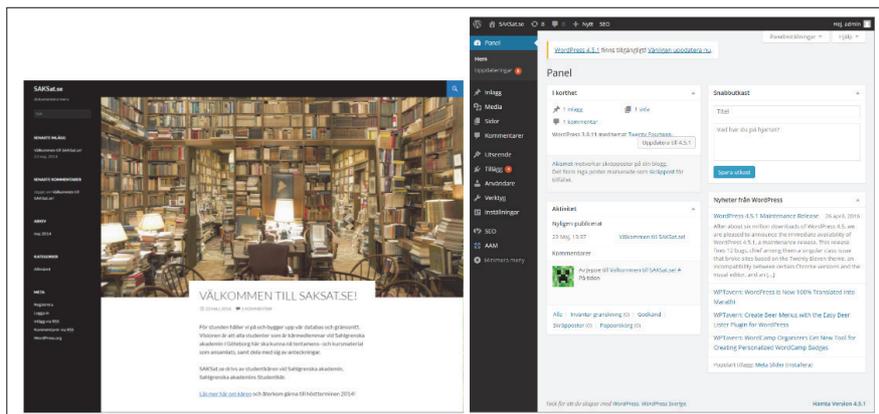
SAKS har redan en webbplats. Webbplatsen är skapad med WordPress och publicerad på Internet genom webbhotelltjänsten One.com. SAKS gav oss administrativ tillgång till webbplatsen och webbhotellet. Arbetet bygger vidare på den existerande webbplatsen.

Webbhotellet använder Apache som webbserver, MariaDB som RDBMS och PHP som server-skriptspråk. För att hantera databaser används phpMyAdmin. Det finns även en filhanteraren i vilket filer kan lagras. I filhanteraren finns de WordPress-filer som bygger upp webbplatsen. WordPress har även en egen databas i webbhotellets RDBMS (se Figur 5.1).



Figur 5.1: Filsystemet och databasen för webbplatsen via One.com.

Webbplatsen har ett utseende genom ett WordPress-tema (se Figur 5.2). För tillfället kan vilken användare som helst besöka webbplatsen. Användaren kan skriva kommentarer till inlägg gjorda av administratören. Användaren kan även registrera sig eller logga in på webbplatsen. Det finns ingen webbsida där tentamina kan sökas eller visas.



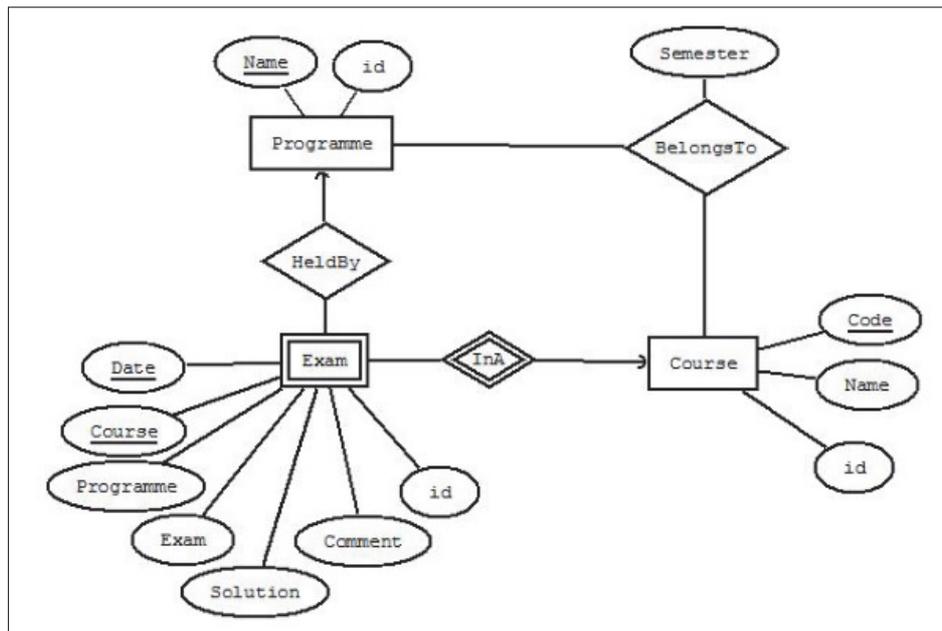
Figur 5.2: Den nuvarande ingångs- och administratörssidan.

Webbplatsen har en administrativ sida som endast är tillgänglig för administratören (se Figur 5.2). På den administrativa sidan hanteras inlägg, innehåll på webbsidor, kommentarer, med mera. Det finns också mer avancerade inställningar för webbplatsen, till exempel installation av insticksprogram.

6 DESIGN

6.1 Databasdesign

För att kunna implementera tentamenshantering på webbsidan behövs det läggas till ett par tabeller i den redan befintliga WordPress databasen. I Figur 6.1 visas ER-diagrammet över de tabeller som lades till i databasen. I Figur 6.2 visas schemat som översattes från ER-diagrammet i Figur 6.1.



Figur 6.1: ER-diagram över databastabeller skapade av insticksprogrammet.

```
Programme(Name, id)
  id unique
Course(Code, Name, id)
  id unique
BelongsTo(Programme, Course, Semester)
  Programme -> Programme.name
  Course -> Course.code
Exam(Date, Course, Programme, Exam, Solution, Comment, id)
  Course -> course.code
  Programme -> Programme.name
  id unique
```

Figur 6.2: Översatt schema för ER-diagrammet.

De tabeller som konstruerades var följande; *Programme*, *Course*, *Exam* och *BelongsTo*. Koderna för att skapa tabellen *Course* visas i Figur 6.3. Tabellerna har utöver sina egna attribut, ett extra attribut som sparar undan Wordpress inläggens unika id, vilket skapar en koppling till Wordpress-tabeller. Det är en viktig funktion som behövs för att de olika tabellerna skall kunna arbeta synkroniserat med varandra. Tabellen *BelongsTo* behöver inte ett unikt id, då den endast håller reda på vilken kurs som hör till vilket program.

```
CREATE TABLE Course (  
Code VARCHAR(10) PRIMARY KEY,  
Name VARCHAR(50),  
id BIGINT(10) UNIQUE  
);
```

Figur 6.3: SQL-kod för att skapa tabellen "Course".

Tabellen *Programme* har hand om alla utbildningsprogram som har en, eller flera kurser med tillhörande tentamen och svar. När ett utbildningsprogram läggs till på webbsidan sparas namnet undan i attributet *name*, som även är primärnyckeln för tabellen. På webbsidan görs en kontroll så att inte samma program kan läggas till flera gånger och eftersom *name* är nyckel i tabellen, så kan det inte finnas dubletter.

Varje utbildningsprogram har åtminstone en kurs länkad till sig och informationen om kurserna sparas undan i tabellen *Course*. I *Course* sparas namnet på kursen och dess kurskod och där kurskod är primärnyckeln för tabellen. Det kan finnas flera kurser med samma namn fast som går på olika utbildningsprogram, därför letar webbsidan och databasen endast efter unika kurskoder, då alla kurser har en unik kurskod på Sahlgrenska Akademin.

En kurs kan vara länkad till mer än ett utbildningsprogram och de ges i en viss termin, därför behövs det en speciell tabell som sköter relationen mellan kurs och utbildningsprogram. Tabellen i fråga heter *BelongsTo* och den kopplar ihop vilken kurs som hör till vilket/vilka utbildningsprogram. Primärnyckeln till tabellen är en kombination av nyckeln från *Course* och nyckeln från *Programme*.

Tabellen *Exam* innehåller information om alla tentamina som har lagts upp på webbsidan. Tabellen är en så kallad svag entitet, vilket innebär att den inte är unikt identifierbar endast av sina egna attribut. Tabellen måste tillsammans med en primärnyckel från en annan tabell som den har en relation till (i detta fallet tabellen *Course*) och ett av sina egna attribut, skapa en primärnyckel som är unikt identifierbar för *Exam*. Primärnyckeln för *Exam* är en hopslagning av nyckeln från *Course*, som är *code*, och attributet *date* från *Exam*, dessa attribut bildar en unik dubbelnyckel för *Exam*.

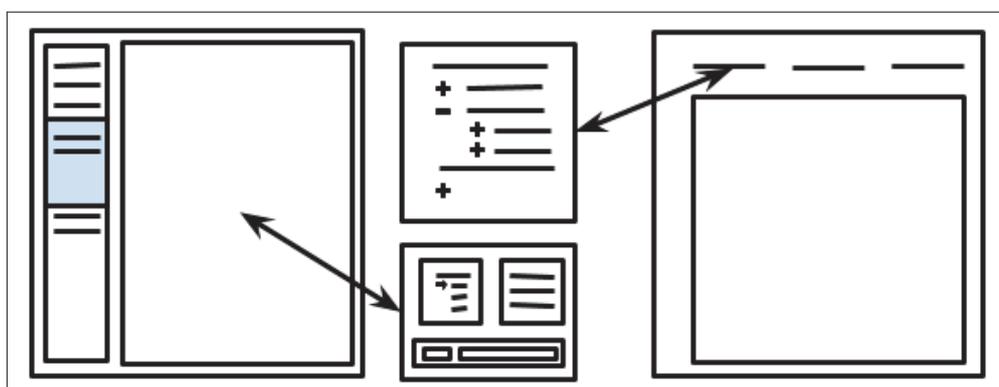
När en tentamen läggs upp behöver den kopplas till en kurskod och ett utbildningsprogram, samt till tesen och eventuellt till ett svar. Dessutom måste tentamen vara länkad till ett specifikt datum. Utöver detta kan administratören som lägger upp tentamen välja att lägga in frivilliga kommentarer. Flera tentamina kan ha samma datum, så länge de inte har samma kurskod vilket förklarar varför kurskod och datum ihop bildar den unika nyckeln för tabellen.

6.2 Grafiskt användargränssnitt

Webbplatsen hade ett grafiskt användargränssnitt genom ett installerat WordPress-tema. Gränssnittet behövdes därför endast utökas.

WordPress implementerar ett system där innehållet på administratörssidan varierar beroende på menyval. I en panel som finns på vänster sida av administratörssidan visas dessa val (se Figur 6.4). Alla val finns tillgängliga vart man än befinner sig inom sidan. Nytt innehåll läggs till genom att lägga till nya val till panelen. Vilken typ av innehåll och dess utseende bestäms genom WordPress metoder och egen skriven kod. I denna panelen lades ett nytt menyalternativ till för uppladdning av tentamina.

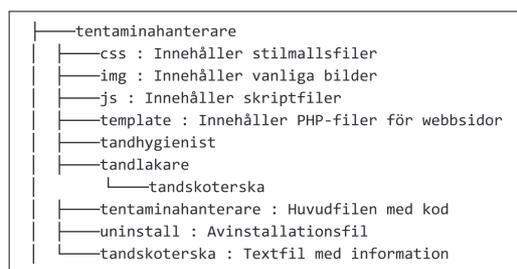
För att visa upplagd tentamina lades länkar till i webbplatsen sidhuvud (se Figur 6.4). Varje utbildningsprogram har en egen länk. Varje länk tar användaren vidare till en webbsida med en trädstruktur som visar ett programs kurser och tentamina. Dessa länkar är tillgängligt för webbplatsens användare var de än befinner sig på webbplatsen.



Figur 6.4: Skiss över administratörssidan (vänster) och webbplatsen (höger).

6.3 Arkitekturmönster och filstruktur

För att aktivera ett insticksprogram i WordPress måste en mapp och en PHP-fil finnas. Filen måste även ha samma namn som mappen. Det är denna filen som anropas när insticksprogrammet aktiveras. I detta fall är namnet på mappen och filen "tentaminahanterare" (se Figur 6.5). Insticksprogrammet är byggt enligt EDA-mönstret eftersom WordPress använder händelser för att meddela när något sker. Funktioner som lyssnar efter dessa händelser finns i samma fil som används vid aktivering av insticksprogrammet. Förutom den nödvändiga PHP-filen innehåller mappen även andra relevanta filer, till exempel stilmalls- och JavaScript-filer.



Figur 6.5: Applikationens filstruktur.

7 IMPLEMENTATION

7.1 Utvecklingsverktyg och arbetssätt

Arbetet genomfördes i iterationer. I den första iterationen uppfylldes de högst prioriterade kraven från kravspecifikationen.

De grundläggande verktygen som behövdes för att utveckla statiska webbsidor var ett textredigeringsprogram och en webbläsare. Serverskriptspråk var nödvändigt för dynamiskt innehåll på webbsidor. Det gick att använda den driftsatta miljön på webbhotellet för att utveckla webbplatsen, men detta medförde problem. Alla förändringar i filerna visades direkt på den publika webbplatsen. Även databasen som användes av webbplatsen påverkades. Detta medförde risker som till exempel förlorad data, lagring av "skräpvärden" och att databasstrukturen tog skada. En lösning var att installera en lokal utvecklingsmiljö i testnings- och utvecklingssyfte. En sådan miljö hade datorn den kördes på som värd och en kopia av databasen och alla filer som fanns på webbhotellet. För att installera den lokala miljön behövdes tre saker: En webbserver, en databas och ett serverskriptspråk. För att underlätta installationen användes ett program med en mjukvarustack.

7.1.1 Lokal utvecklingsmiljö

En miljö med samma mjukvara som webbhotellets webbserver användes. Detta för att garantera att insticksprogrammet fungerar när det används med den publika webbplatsen. Valet föll på XAMPP. Den enda skillnaden i använd mjukvara är webbhotellets MariaDB och XAMPPs MySQL som RDBMS. Detta medförde inga större risker eftersom de har hög kompatibilitet.

7.1.2 Textredigeringsprogram

Valet av textredigeringsprogram föll på Atom. Programmet har ett antal användbara funktioner som underlättar vid utveckling av en webbsida, till exempel automatisk komplettering av kod, filhanterare, hitta och ersätta ord. Programmet kan användas för att skriva HTML-, CSS- och PHP-kod och filer.

7.1.3 Utvecklingsdator

En lokal dator på Sahlgrenska Universitetet med Windows 7 som operativsystem användes. Atom installerades som textredigeringsverktyg för utveckling av webbplatsen. Även Windows-versionen av XAMPP installerades för att på så sätt få tillgång till en lokal utvecklingsmiljö.

7.2 Databasen

I PHP-filen för insticksprogrammet skrevs all kod för att lägga till och hantera tabeller i databasen på webbhotellet och för att utveckla webbplatsen. Tabellernas uppgift är att skapa en fungerande trädstruktur som dynamiskt uppdateras med tillgängliga tentamina som tillhör respektive kurs, som i sin tur hör till ett program. För att kunna implementera tentamenshantering på webbsidan behövs det läggas till tabeller i den redan befintliga WordPress-databasen.

Tabellerna skapades i ett tidigt skede av arbetet men modifierades och uppdaterades åtskilliga gånger. Detta på grund av att de svar som hade konstruerats från början inte helt överensstämde med hur WordPress egentligen fungerade.

Den planerade funktionaliteten på webbplatsen krävde mer av våra tabeller än först förväntat. Tabellerna behövde skapas på nytt med fler attribut och relationer för att kunna ge den funktionalitet som ska uppnås. Till exempel behövs det ett attribut som kopplar våra tabeller till WordPress egna, något vi inte kände till i början av arbetet.

När man tog bort ett inlägg från webbsidan, så togs det bort från WordPress-tabeller i databasen, men inte från de egenskapade tabellerna. Genom att införa id attributen i de egenskapade tabellerna, som sparade undan id från relaterade WordPress tabeller kunde hanteringen av inlägg mellan tabellerna synkroniseras. Det skapar en felsäker metod för att kunna lägga till, ta bort och ändra inlägg som kurser, tentamina och utbildningsprogram.

När ER-diagrammet var komplett översattes det till ett schema som användes vid implementeringen av tabellerna på testdatabasen som låg på den lokala servern. Därefter testades tabellerna genom att lägga in testinlägg på dem och se så att det inte fanns några buggar som hade missats. Då tabellerna fungerade som förväntat påbörjades skapandet av webbsidan och dess funktioner för att hantera de nyskapade tabellerna.

7.3 Insticksprogrammet

Insticksprogrammet installerades på WordPress genom att lägga till mappen för programmet i WordPress insticksprogram mapp. Därefter aktiverades programmet genom administratörspanelen på webbplatsen.

7.3.1 Händelsehantering

För att registrera egna funktioner som lyssnare till WordPress händelser används metoden `add_action()`. Metoden tar två inparametrar: Den händelse som en funktion ska läggas till och vilken metod som funktionaliteten hämtas från (se Figur 7.1).

```
function method_to_use () {
    ...
    // Kod för metoden
    ...
}
add_action ( 'save_post' , 'method_to_use' );
```

Figur 7.1: En exempelmetod ansluts till händelsen `save_post`”.

Ett problem med WordPress är att endast fem typer av inlägg finns: Posts, Pages, Attachments, Revisions och Nav Menus. Inget av dess alternativ har möjlighet att lägga upp innehåll på webbplatsen som arbetet kräver, till exempel PDF-filer. För att lösa detta skapades en anpassningsbar inläggstyp (eng. custom post type), som gör det möjligt att specificera vilket innehåll som ska läggas upp, med metoden `register_post_type()`. Metoden har två inparametrar. Den första inparametern är det identifierande namnet för inläggstypen. Den andra inparametern är en array med extra argument. Dessa argument specificerar till exempel inläggstypens namn i singular och plural (se Figur 7.2). En ny inläggstyp skapades för arbetet. Den gavs namnet *Tentamina*. Inlägg av denna typen kallas *tentaminsinlägg*.

```
function create_custom_post_type () {
    register_post_type(
        'example_post_type' ,
        array(
            'labels' => array(
                'name' => __('Examples'),
                'singular_name' => __('Example')
            ),
            'public' => true,
        )
    );
}
add_action ( 'init' , 'create_custom_post_type' );
```

Figur 7.2: Exempel på hur en anpassad inläggstyp skapas.

För att gruppera upplagda inlägg används klassificeringar (eng. taxonomies). På så sätt ges en överblick över hur många inlägg av en viss klass som har lagts upp. Anpassningsbara klassificeringar (eng. custom taxonomies) kan skapas för att anpassa vilka typer av klassificeringar en inläggstyp ska innehålla. En klassificering kan sedan ges olika benämningar (eng. terms). Till exempel tilldelas inläggstypen “Filmrecension” klassificeringen “Genre” och termen “Humor”. Det går då att se hur många recensionsinlägg av humorfilmer som har gjorts. Klassificeringar skapas med metoden `register_taxonomy()`. Metoden tar tre inparametrar. De första två är det identifierade namnet på klassificeringen och vilken inläggstyp klassificeringen ska tillhöra. Den sista inparametern är en array med argument som ytterligare definierar klassificeringen (se Figur 7.3). Tre stycken klassificeringar skapades för den anpassade inläggstypen med namnen *Kurser*, *Terminer* och *Program*.

```
function create_custom_taxonomy () {
    register_taxonomy (
        'example_taxonomy' ,
        'example_post_type' ,
        array(
            'labels' => array(
                'name ' => __( 'Tax Examples' ),
                'singular_name' => __( 'Tax Example' )
            ),
            'rewrite => array( 'slug' => 'genre' ),
        )
    );
}
add_action ( 'init' , 'create_custom_taxonomy' );
```

Figur 7.3: Exempel på hur en anpassad klassificering skapas.

När ett nytt inlägg av en anpassad typ skapas, eller om ett existerande ska redigeras, så ska information ofta läggas till eller ändras. WordPress har redan några standard inmatningsalternativ som visas, till exempel titeln på inlägget. Om man vill ha anpassad inmatningsalternativ, till exempel en betygsskala, så används metalådor (eng. meta boxes). Metoden `add_meta_box()` gör det möjligt att lägga till anpassade metalådor. Metoden tar fyra inparametrar. Dessa ger metalådan ett identifierande namn, en titel, namnet på den metoden som ger metalådan sitt innehåll och namnet på inläggstypen som metalådan ska tillhöra (se Figur 7.4). Lådorna visas när ett nytt inlägg skapas eller när ett inlägg ska redigeras.

```
function add_custom_meta_box () {  
  
    add_meta_box (  
        'custom_meta_box_id' ,  
        __( 'Custom Meta Box' ) ,  
        'custom_meta_box_style' ,  
        'example_post_type'  
    );  
  
}  
...  
function custom_meta_box_style () {  
    $output = '  
        <div>  
            <h1>This is a custom meta box</h1>  
            <p>It is displaying some text</p>  
        </div>';  
    echo $output;  
}  
add_action ( 'add_meta_boxes' , 'add_custom_meta_box' );
```

Figur 7.4: Exempel på hur en metalåda skapas.

Sex stycken metalådor skapades för den anpassade inläggstypen. Tre lådor skapades automatiskt när de tre klassificeringarna skapades. De togs bort eftersom de tillät flera val att göras. Till exempel kunde ett inlägg tilldelas flera kurser. De ersattes med metalådor som innehöll rullgardinsmenyer där endast ett val är tillåtet. De sista tre metalådorna gör det möjligt att ge ett inlägg skrivdatum, kommentar och tentamenstes- och svarsdokument.

Ett problem med WordPress är att dess metalådor inte tillåter uppladdning av PDF-filer som standard. Det går att införa denna möjligheten genom koden i Figur 7.5. Koden uppdaterar metalådans `<form>`-element och ser till så att inga karaktärer i dess värde teckenkodas.

```
function update_edit_form() {  
    echo 'enctype="multipart/form-data";'  
}  
add_action('post_edit_form_tag', 'update_edit_form');
```

Figur 7.5: Kod för att uppdatera metalådors `<form>`-tagg.

7.4 Implementation av grafiskt användargränssnitt

Det går att skapa anpassningsbara sidor (eng. custom pages) för WordPress. Detta används när en utvecklare vill att inlägg ska visas på ett annorlunda sätt, till exempel i tabellform, än hur WordPress visar inlägg som standard. Det går att skapa sidor genom administratörspanelen på webbplatsen men det går också att göra i kod.

För att ge en sida ett utseende används en mallfil (eng. template file). Detta är en separat PHP-fil som tilldelas en sida genom administratörspanelen. Filen kan innehålla PHP-, HTML- och JavaScript-kod. I Figur 7.6 visas ett exempel på en enkel mallfil. Den viktigaste delen i mallfilen är Template Name: Custom Pagesom berättar för WordPress att filen är en mall och att namnet på den är "Custom Page". Resten av koden är upp till utvecklaren som bestämmer vad som ska visas på den anpassade sidan.

```
<?php
/*
 * Template Name: Custom Page Template
 */
?>

<?php get_header(); ?>

This text will be displayed on the custom page this template is linked to!

<?php get_footer(); ?>
```

Figur 7.6: Exempel på grundläggande kod för en mallfil.

Tre stycken anpassade sidor skapades, en för varje utbildningsprogram. På så sätt påverkar inte ändringar i en sida de andra. Varje mallfil innehåller kod för en trädstruktur. Informationen som visas i trädstrukturen hämtas från databasen, både från WordPress-tabeller och de egenskapade. Trädstrukturen uppdateras automatiskt efter vilken tentamen som lagts till vilket utbildningsprogram respektive kurs. När en tentamen läggs till på webbplatsen läggs dess information till i både de egenskapade tabellerna och WordPress-tabeller. WordPress-tabellerna använder den undansparade informationen om tentamina för att skapa inlägg på webbplatsen. I tabellerna vi skapade används informationen för att kunna skapa en fungerande trädstruktur. PHP-koden för trädstrukturen hämtar data från tabellerna vi skapat och skriver ut det på respektive utbildningsprograms sida. Till exempel, så läggs det till en tentamen på webbplatsen som har kurskoden "TM900". Kursen går på tandhygienist programmet. Tentamen läggs nu automatiskt till i trädstrukturen för tandhygienistprogrammet, under kursen "TM900". Inlägget har länkar till tentamenstes respektive svarsdokument som har sparats undan på webbservern, samt stöd för en kommentar från administratören. Det sorteras efter skrivdatum och har formatet YYYY-MM-DD.

Det går att genomföra tilldelningen av mallfiler i kod genom att lägga till ett filter med metoden `add_filter()`. Ett filter tillåter en utvecklare att ändra data som används av WordPress under körning (eng. runtime). Exempelmetoden i Figur 7.7 kallas innan WordPress tilldelar sidor deras mallar. Den ersätter en mallfil med en fil som skapats av utvecklaren. En filtermetod skrevs liknande exemplet. På så sätt kunde varje utbildningsprogram tilldelas en webbsida med en egenutvecklad mall automatiskt.

```
function set_custom_template ( $template ) {
    // Check if it is the correct page for the template
    if (is_page ( 'example_post_type' ) ) {
        // Get the custom template file from the plugin directory
        $new_template = plugin_dir( 'custom-page-template.php' );
        // If file could be found, return it
        if ( $new_template != '' ) {
            return $new_template;
        }
        // If not, notify the user and return the standard template file
    } else {
        write_log("Error: Custom template file could not be found!");
        return $template;
    }
    // If it isn't the correct page, do nothing and just return
    else {
        return;
    }
}
add_filter ( 'template_include' , 'set_custom_template' );
```

Figur 7.7: Exempel på filtermetod för tilldelning av en anpassad mallfil.

7.5 Säkerhet

För att säkerställa att endast administratörerna kan använda sig av funktionaliteten i insticksprogrammet genomförs olika säkerhetskontroller i koden, bland annat kontroll av användarbehörighet (se Figur 7.8) och nonce-värden. Ett nonce-värde är ett numeriskt värde som används endast en gång. Visa metoder har kodstycken som kontrollerar användarens roll (eng. user role) och ser till så att han eller hon är en administratör.

```
if (current_user_can('activate_plugins') && is_admin()) {
    // Kod som ska exekveras
}
else {
    // Kod som ska exekveras
}
```

Figur 7.8: Kod för att kontrollera en användares roll.

Nonce-värden ser till att webbplatsen inte missbrukas. Värdet innehåller slumpmässigt valda siffror och sparas i ett gömt fält. I Figur 7.9 skapas ett nonce-värde och sparas i metalådan för ett inläggs kurskod.

```
wp_nonce_field(plugin_basename(__FILE__), 'course_code_nonce');
```

Figur 7.9: Gömt nonce-fält för kurskodens metalåda.

Ett exempel på en metod som kontrollerar ett nonce-värde ges i Figur 7.10. När en administratör försöker tilldela ett inlägg en kurskod görs först en kontroll av nonce-värdet. Metoden kontrollerar först att ett nonce-värde för kurskodens metalåda finns. Om ett värde finns så kontrolleras det mot nonce-värdet från händelsen \$_POST. Om dessa två värden stämmer överens innebär det att tilldelningen av en kurskod skedde genom metalådan på administratörssidan. Metoden informerar då att nonce-värdet är korrekt och returnerar true.

```
function correct_nonce($nonce) {
    if(!isset($_POST[$nonce]) ||
        !wp_verify_nonce($_POST[$nonce],
            plugin_basename( __FILE__ ))) {
        return false;
    } else {
        write_log($nonce . ' är korrekt! ');
        return true;
    }
}
```

Figur 7.10: Exempel på metod för kontroll av nonce-värde.

Nonce-värden ser till att endast auktoriserade användare kan utföra vissa handlingar på webbplatsen. Ett exempel är om en utomstående person kopierar URL:en för att uppdatera ett inlägg och byter ut kurskoden mot en annan. Ett nonce-värde ser till så att denna förändringen endast kan ske genom metalådan på administratörssidan.

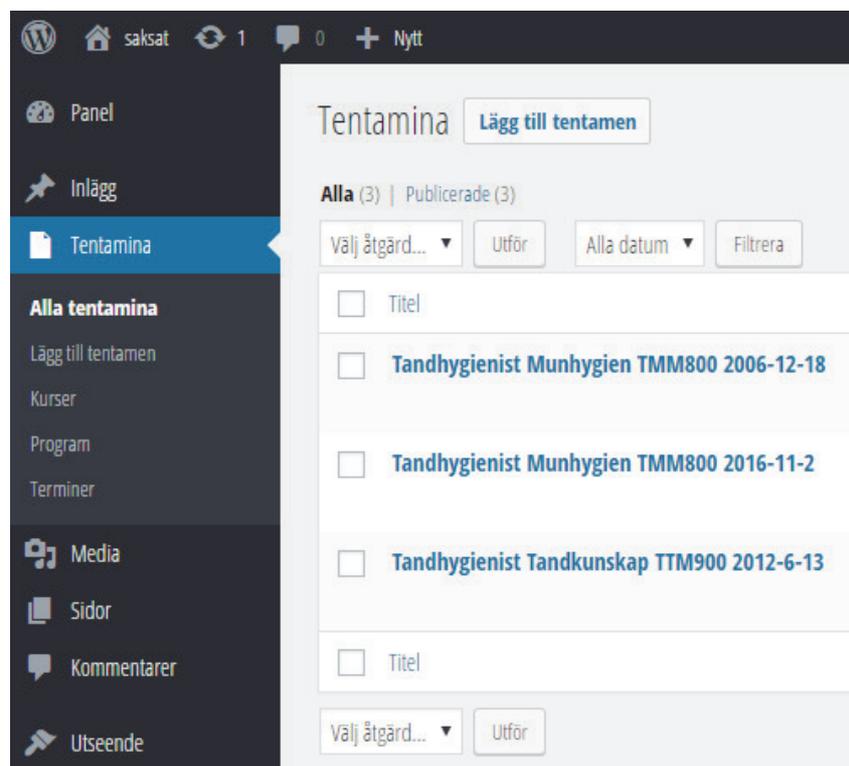
8 RESULTAT

8.1 Applikationen

Arbetet resulterade i ett insticksprogram som ger administratören ytterligare funktionalitet för att lägga upp och hantera tentamina. Insticksprogrammet utvecklar även webbplatsens utseende för att göra det möjligt för användare att ta del av upplagd tentamina.

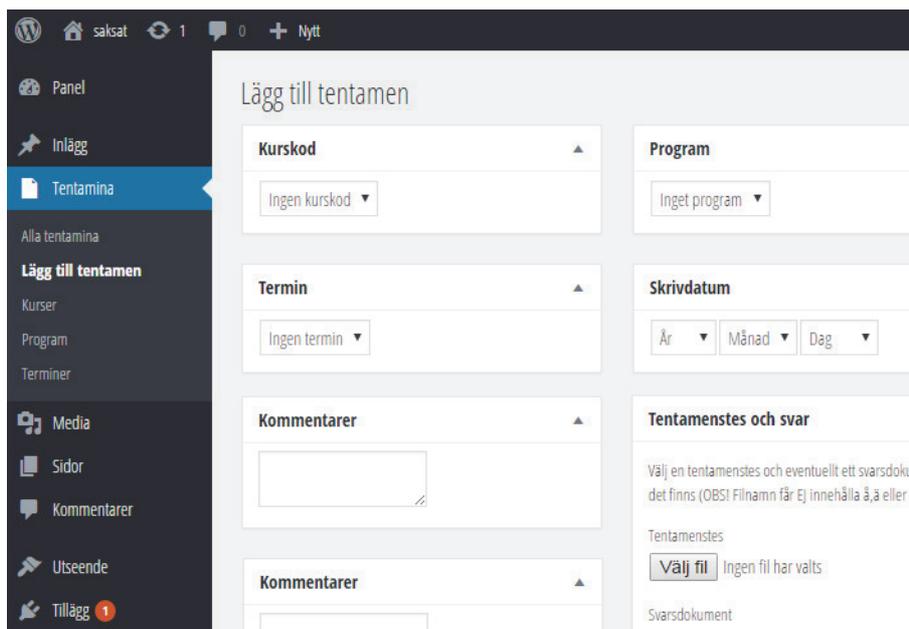
8.1.1 Administrativt

På webbsidan för administratören syns den skapade inläggstypen (se Figur 8.1). I panelen till vänster syns huvudmenyalternativet “Tentamina”. Om man klickar på detta alternativet så visas ytterligare fem undermenyalternativ. Administratören kan välja att visa redan upplagd tentaminainlägg eller lägga till ett nytt tentaminsinlägg. Det går även att hantera de tillhörande tre klassificeringarna. Klickar man på några av dessa fem alternativ så visas nytt innehåll på administratörssidan relevant till det valda alternativet.



Figur 8.1: Administratörspanelen med “Tentamina”-alternativet.

Klickar man på “Alla tentamina” (se Figur 8.1) så visas en lista över alla upplagda tentaminsinlägg. Det går att välja, redigera, ta bort eller visa en eller flera inlägg. Klickar man på ett inlägg så visas mer information, till exempel dess tentamenstes.



Figur 8.2: Undermenyalternativet “Lägg till tentamina”.

På “Lägg till tentamen” visas innehåll för att lägga upp en ny tentamen (se Figur 8.2). Kurskod, termin, program, tentamenstes och skrivdatum måste väljas. Det går att lägga till ett svarsdokument och skriva en kommentar, men detta är inte obligatoriskt.

Om ett tentamensinlägg redan har ett tentamenstes- eller svarsdokument så visas namnen på dessa dokument i metalådan “Tentamenstes och svar”. Innan ett inlägg läggs upp på webbplatsen görs en kontroll att alla obligatoriska val har gjorts. Även en kontroll att tentamenstes- och svarsdokument är av filtypen PDF görs. Om obligatoriska val inte har gjorts eller filtypen av ett dokument inte är ett PDF så visas ett felmeddelande. Om alla nödvändiga val har gjorts korrekt så sparas informationen för tentamensinlägget i WordPress-tabeller och insticksprogrammets egna tabeller i databasen. Tentamenstes- och svarsdokumentet läggs upp i filhanteraren på webbhotellet.

De tre undermenyalternativen “Kurser”, “Program” och “Terminer” i panelen gör det möjligt för administratören att lägga till, ta bort eller redigera benämningar för respektive klassificering. Innehållet som visas på administratörssidan för dessa tre undermenyalternativ är lika varandra. Alla klassificeringar kräver att ett namn skrivs in när en ny benämning ska skapas, till exempel namnet på ett program under klassificeringen “Terminer”. Men en skillnad finns. En benämning under klassificeringen “Kurser” behöver förutom namnet på kursen även en kurskod. Om ingen kurskod skrivs in så visas ett felmeddelande när administratören försöker spara den nya benämningen. Om alla nödvändiga val har gjorts korrekt så läggs benämningen och dess information upp i WordPress tabeller i databasen. Benämningen visas därefter i en lista tillsammans med andra existerande benämningar för sin respektive klassificering.

8.1.2 Webbplatsen

I webbplatsens sidhuvud (se Figur 8.3) visas tre separata länkar som tar en användare vidare till en webbsida för respektive utbildningsprogram. Varje sida innehåller en trädstruktur med tentamina för utbildningsprogrammet. Strukturen innehåller information som kurskod och kursnamn. Det går att expandera och kollapsa så att alla tentamina visas eller göms. Genom att klicka på en kurs visas eller göms endast tentamina som tillhör den kursen. När en kurs har expanderats visas information om varje specifik tentamen som kursen har. Varje tentamen har ett skrivdatum, en länk till tentamenstes- och ett eventuellt svarsdokument, samt en kommentar.



Figur 8.3: Trädstrukturen för programmet “Tandhygienist”.

8.2 Slutgiltiga resultat

Den slutgiltiga produkten uppnådde nästan alla de grundläggande kraven. Kontroll av användare påbörjades men kunde inte slutföras i tid. Att skapa denna kontrollen tog mer tid än uppskattat. Arbetet på möjligheten att söka efter tentamen blev nästan avklarat. Det fanns inte tillräckligt med tid till att testa denna funktionaliteten. Insticksprogrammet är ändå inom förväntan. Under arbetets gång har kunskap från vår utbildning använts och utvecklats. Även ny kunskap har införskaffats.

8.3 Utvärdering av verktyg

Atom rekommenderas som textredigeringsprogram för webbutveckling. Det fungerade bra att arbeta med programmet. Inga buggar eller allvarliga fel stöttes på och programmet fungerade felfritt utan några kraschar eller liknande. Programmet användes för HTML-, PHP-, CSS- och JavaScript-kod vilket gör det till ett universellt verktyg.

XAMPP har fungerat bra under arbetets gång. Arbetet med insticksprogrammet kom igång snabbt då all nödvändig mjukvara installerades samtidigt. Det enda problemet med XAMPP uppstod under installationen. Apache ville använda portar som redan användes av Skype. Genom att ändra vilka portar Skype använde löstes problemet. Inga andra problem uppstod efter detta.

Användning av phpMyAdmin i den lokala och globala webbservern har fungerat väl under arbetets gång. phpMyAdmin var smidigt vid hantering av databasen och när tabeller skapades. Istället för att skriva SQL-kod för hand för att skapa tabeller, attribut och inlägg går det att göra detta genom menyalternativ som till exempel "Skapa ny databas" eller "Lägg till tabell". Detta förkortade tiden det tog att utveckla databasen.

8.4 Teknikens roll i samhället

Insticksprogrammet gör det möjligt att lägga upp tentamen på webben vilket gör dem mer lätt-tillgängliga. Alla studenter på Sahlgrenska Akademin har då möjlighet att ta del av tentamina, trots avstånd till skolan, handikapp, med mera. Insticksprogrammet skapar även ett alternativ till att hämta ut tentamina i pappersformat. Detta kan minska mängden pappers som används för utskrivning av tentamina och på så sätt miljön.

9 SLUTSATS

Eftersom inget arbete inom webbutveckling hade gjorts av gruppen innan detta arbetet så gjordes en generös tidsplan. Till en början ansågs det att lite tid behövdes till instuderingen av programmeringsspråk. Anledningen var gruppens bakgrund inom programmering. När SAKS meddelade ett par veckor in i tidsplanen att ett webbhotell och WordPress skulle användas så uppstod visa avvikelser från tidsplanen. Tid till att upprätta en server hoppades över och istället lades tid till inläring av WordPress. Inläringen av WordPress tog längre tid än först uppskattat, men annars följdes den ursprungliga tidsplanen.

Arbetets mål och syfte har uppnåtts. Ett insticksprogram har skapats för SAKS WordPress-webbplats som tillåter uppläggning och hantering av tentamina. Hantering blir enklare eftersom insticksprogrammet gör det möjligt att kategorisera tentamina för bättre översikt. Insticksprogrammet utvecklar även webbplatsen så att Sahlgrenska Akademiens studenter kan ta del av upplagd tentamina.

En skattning av insticksprogrammet effektivitet kunde inte utföras på grund av tidsbrist. Det går därför inte att med säkerhet uppskatta hur mycket insticksprogrammet underlättar för SAKS och Sahlgrenska Akademiens studenter. Om insticksprogrammet jämförs mot det nuvarande systemet, att hämta ut tentamina från studentexpeditionen, är insticksprogrammet och dess funktionalitet ett effektivare system, då det gör det möjligt att ta del av tentamina över webben.

9.1 Framtida inriktningar

Det finns funktionalitet som kan skapas för insticksprogrammet. En söksida som ett alternativ till att navigera en trädstruktur för att hitta en tentamen kan implementeras. Användare av webbplatsen och administratörer kan då söka efter en specifik tentamen istället för att navigera igenom en trädstruktur. Att implementera en form av innehållsbegränsning för användare av webbplatsen, till exempel att endast inloggade användare kan se tentamina, kan utvecklas. På så sätt är det lättare att se till så endast Sahlgrenska Akademiens studenter kan ta del av upplagd tentamina. Istället för att visa en tentamen i en ny flik i webbläsaren, kan ett sätt att visa en tentamenstes och dess svarsdokument direkt i trädstrukturens webbsida implementeras. Möjligheten för användare att kommentera på tentamensinlägg kan utvecklas. Ett kommentarsfält gör det möjligt för användare att dela med sig av till exempel funderingar angående frågorna i en tentamenstes.

Mer funktionalitet för administratören kan också vidareutvecklas. Statistik över hur webbplatsen används, till exempel hur många gånger en tentamen har hämtats ner, kan implementeras. På så sätt kan administratören få en uppfattning över hur webbplatsen används.

REFERENSER

- [1] F. Levison. (2016) 'lämnade inte ut offentliga handlingar – universitetet jo-anmält'. (Använd 2016-05-28). [Online]. Available: <http://www.test.org/doi/>
- [2] O. S. Initiative. (2007) 'the open source definition'. (Använd 2016-05-29). [Online]. Available: <https://opensource.org/osd>
- [3] DB-Engines. (2016) 'db-engines ranking'. (Använd 2016-04-12). [Online]. Available: <http://db-engines.com/en/ranking>
- [4] ISO. (2011) 'iso/iec 9075-1:2011 - information technology – database languages – sql – part 1: Framework (sql/framework)'. (Använd 2016-05-29). [Online]. Available: http://www.iso.org/iso/home/store/catalogue_ics/catalogue_detail_ics.htm?csnumber=53681
- [5] DB-Engines. (2016) 'mysql system properties - editorial information provided by db-engines'. (Använd 2016-04-19). [Online]. Available: <http://db-engines.com/en/system/MySQL>
- [6] S. J. Vaughan-Nichols. (2013) 'google quietly dumps oracle mysql for mariadb'. (Använd 2016-04-19). [Online]. Available: <http://www.zdnet.com/article/google-quietly-dumps-oracle-mysql-for-mariadb/>
- [7] A. Feldman. (2013) 'wikipedia adopts mariadb'. (Använd 2016-04-19). [Online]. Available: <https://blog.wikimedia.org/2013/04/22/wikipedia-adopts-mariadb/>
- [8] MariaDB Foundation. (2016) 'about mariadb'. (Använd 2016-05-20). [Online]. Available: <https://mariadb.org/about/>
- [9] phpMyAdmin. (2016) 'about'. (Använd 2016-05-20). [Online]. Available: <https://www.phpmyadmin.net/>
- [10] R. Fielding and J. Reschke. (2014) 'hypertext transfer protocol (http/1.1): Message syntax and routing'. (Använd 2016-05-20). [Online]. Available: <https://tools.ietf.org/html/rfc7230>
- [11] Techopedia. (2016) 'stateless protocol'. (Använd 2016-05-20). [Online]. Available: <https://www.techopedia.com/definition/697/stateless-protocol>
- [12] The Apache Software Foundation. (2016) 'about the apache http server project'. (Använd 2016-05-20). [Online]. Available: https://httpd.apache.org/ABOUT_APACHE.html
- [13] Netcraft. (2016) 'march 2016 web server survey'. (Använd 2016-03-29). [Online]. Available: <http://news.netcraft.com/archives/2016/03/18/march-2016-web-server-survey.html>
- [14] W3C. (2016) 'html & css'. (Använd 2016-05-22). [Online]. Available: <https://www.w3.org/standards/webdesign/htmlcss>
- [15] The PHP Group. (2016) 'what is php?'. (Använd 2016-05-22). [Online]. Available: <https://secure.php.net/manual/en/intro-what-is.php>
- [16] Chapman, Stephen. (2016) 'what is javascript?'. (Använd 2016-06-04). [Online]. Available: <http://javascript.about.com/od/reference/p/javascript.htm>
- [17] WordPress. (2016) 'about wordpress'. (Använd 2016-05-23). [Online]. Available: <https://wordpress.org/about/>

- [18] iThemes Media. (2016) 'what is a wordpress theme'. (Använd 2016-05-23). [Online]. Available: <https://ithemes.com/tutorials/what-is-a-wordpress-theme/>
- [19] iThemes Media . (2016) 'what is a wordpress plugin'. (Använd 2016-05-23). [Online]. Available: <https://ithemes.com/tutorials/what-are-wordpress-plugins/>
- [20] STANDS4. (2016) 'what does xampp stand for?'. (Använd 2016-05-23). [Online]. Available: <http://www.abbreviations.com/XAMPP>
- [21] Apache Friends. (2016) 'about'. (Använd 2016-05-20). [Online]. Available: <https://www.apachefriends.org/about.html>

BILAGOR

Bilaga A: Initial tidsplan

Tidsplan som sammanställdes tillsammans med handledare under projektets planeringsfas

Uppgift	Lv 1	Lv 2	Lv 3	Lv 4	Lv 5	Lv 6	Lv 7	Lv 8	Lv 9	Lv 10
Planeringsrapport	■									
Bakgrund och problem		■								
Undersök och bestäm mjukvara		■	■							
Upprätta server		■	■							
Upprätta databas			■	■	■					
Teknisk bakgrund			■	■	■	■				
Upprätta hemsida					■	■	■			
Genomförande			■	■	■	■	■	■	■	■
Testa serveranslutningen			■	■						
Testa databasen (lägg till/ta bort tenta)					■	■				
Testa hemsidan (hämta/hitta tenta)							■			
Sammanfattning/abstract										■
Diskussion/slutsats									■	■
Rapportdesign (sidnummer, bilder, etc)									■	■
Adminverktyg för hemsidan							■	■		
Instruktionsmanual									■	■
Kodkommentering/upprepning									■	■
Gå igenom rapportreferenser										■
Innehållsförteckning, ordlista, etc										■