Datavetenskap

Staffan Nilsson och Mattias Pehrsson

Online självvärderingskunskapstest

Examensarbete, C-nivå

2004:17

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

Staffan Nilsson

Mattias Pehrsson

Godkänd, 2004-06-09

Handledare: Donald F. Ross

Examinator: Stefan Lindskog

Sammanfattning

I denna C-uppsats redovisas ett examensarbete som utförts inom ämnet Datavetenskap vid Karlstads Universitet. Arbetet har utförts under vårterminen 2004 vid Universitet. Vi fick i uppdrag att designa ett webbaserat kunskapstest av vår handledare Donald F. Ross.

Kunskapstestet är ett test för en student att testa sin förståelse och kunskap för en viss kurs. Frågorna till testet skapas av en kursansvarig för en kurs.

Bakgrunden till projektet var att vi själva skulle använda oss av ett sådant test. Vi ville även lära oss mer hur databaser fungerar över Internet. För att utveckla ett sådant test använde vi oss av flera verktyg. Verktygen vi använde oss av var PHP, HTML, JavaScript och MySQL.

Det huvudsakliga krav vi ställde på produkten var att det skulle vara enkelt att använda.

Resultatet blev att vi fått fram en funktionell prototyp som är fullt duglig att användas för tester på Internet.

I uppsatsen redovisas arbetets alla faser samt de resultat som uppnåtts.

Design and Implementation of a Web Based Self-Assessment Test

Abstract

This report describes a Bachelor's Project at the Department of Computer Science, Karlstad University. The dissertation took place during the spring term 2004 and was located at the University. Our assignment, given by our supervisor Donald F. Ross, was to design a web based knowledge test.

The knowledge test is a test for a student to test their understanding for a specific course. The questions for the test are created by the teacher who is responsible for a course.

The background to the project was that we ourselves would use such a test. We also wanted to learn more about how a database works over Internet.

To develop such test we used several tools. The tools we used were PHP, HTML, JavaScript and MySQL.

The main requirement was to create a product that is easy to use.

The result was a functional prototype that is fully suitable to use for tests on the Internet.

In the dissertation we describe all the phases of the project and the result we achieved.

Tack

Under vårt arbete, i skapandet av en prototyp på ett kunskapstest, har vi stött på olika typer av problem. Ett stort tack vill vi rikta till vår handledare Donald F. Ross för att han har varit pådrivande i projektet. Vi vill också passa på att tacka Monique Pehrsson som testat och lämnat synpunkter på systemet och uppsatsen. Sedan vill vi rikta ett stort tack till Joakim Norlinder och Ola Bull som givit råd beträffande programmeringen.

Innehållsförteckning

Figurförteckning

1. Inledning

Denna uppsats handlar om ett självvärderingskunskapstest, detta kommer vi fortsättningsvis kalla enbart kunskapstest, och hur detta projekt har utvecklats. Vårt system konstruerades från grunden trots att det finns många liknande system på Internet. I kapitel 2 ger vi en bakgrund till projektet och redovisar varför vi tog vissa beslut och hur vi trodde att projektet skulle fortgå.

Vi går även igenom vilka problem som finns och vad man skall ta hänsyn till gällande design, verktyg osv. Det som krävs för ett kunskapstest, som skall finnas på Internet, är att det är enkelt. Bakom användargränssnittet finns en databas som lagrar all information om hur kunskapstestet ser ut och vad studenten svarat i testet. Detta för att studenten ska får reda på sitt resultat på testet. När en prototyp för ett kunskapstest skall utvecklas så måste man ha ett verktyg för att skapa sidor på Internet och en databas som lagrar data. Sidorna som skapas måste skapas dynamiskt och då räcker det inte med HTML som är det sidbeskrivningsspråk som används för att göra hemsidor.

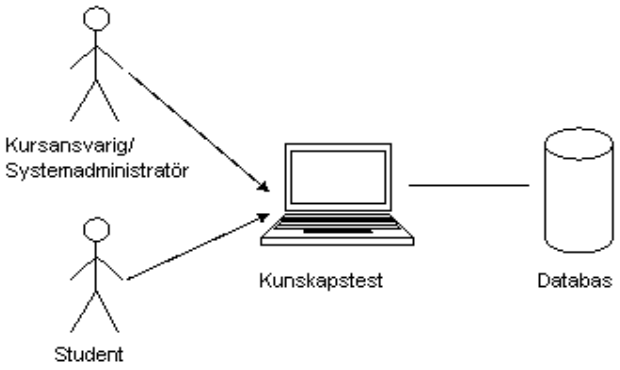
Kapitel 3 ger en beskrivning av vilka språk och verktyg som valdes för projektet, varför vi valde dessa verktyg och hur vi gick tillväga. Där går vi också igenom hur prototypen ser ut. Till Prototypen valdes skriptspråket PHP för att skapa dynamiska sidor och en MySQL-databas för att lagra data.

Prototypen är uppbyggd i två delar. Den ena för studenten och den andra för kursansvarige och systemadministratören. Studenten kommer till en förstasida där denne blir presenterad en text om vad det är för system. När studenten sedan klickar sig vidare så kommer denne till en sida där man får välja inom vilket ämne studenten ämnar göra sitt test. Efter det så får studenten välja vilket test och hur många frågor man vill svara på. Detta sker genom att studenten får välja mellan 5, 10, 15, 20 respektive alla frågor i ett test. Om man inte väljer alla så kommer systemet att slumpa fram det antal frågor man valt från testet men även om man väljer alla kommer frågornas nummerordning att slumpas. När studenten, efter genomfört test, rättar testet så blir denne presenterad antal rätt och vilka frågor man svarat rätt respektive svarat fel på.

Studenten behöver alltså inte logga in, utan vem som helst kan göra testet. Detta kunskapstest skall alltså enbart ses som en övning för studenten. Den kursansvarige och systemadministratören däremot måste logga in. Detta sker på deras startsida som samtidigt är

en inloggningssida. Här har vi skilt på systemadministratören och den kursansvarige. Systemadministratören kommer till en sida där denne kan lägga till kursansvariga som skall kunna skapa ett test. Den kursansvarige kommer till en sida där man kan skapa ett test eller redigera ett befintligt test. Utförligare beskrivning kommer i kapitel 3.

I kapitel 4 går vi igenom de säkerhetsaspekter som finns i systemet och då främst mot databasen och inloggningen av systemadministratören och den kursansvarige. I kapitel 5 redovisar vi resultatet vi kom fram till samt utvärderar detta. Till sist i kapitel 6 så sammanfattar vi hela projektet. Figur 1.1 visar en översiktsbild över systemet med de två olika användartyper som skall finnas i systemet.



Figur 1.1: Översiktsbild av systemet

2. Bakgrund

I samband med att vi skulle skriva en c-uppsats för vår examen i Datavetenskap vid Karlstad Universitet, ville vi utveckla ett kunskapstest online. Detta innebär att en kursansvarig ska kunna skapa ett test som en student ska kunna testa sina kunskaper på, liknande en dugga på Internet.

Vi hade från början tänkt vidareutveckla ett redan befintligt kursutvärderingssystem skapat av Daniel Geschwind[1] och Simon Ogolla[1]. Efter att ha diskuterat med vår handledare/uppdragsgivare valde vi istället att satsa på ett system där man kan testa sina kunskaper. Detta berodde huvudsakligen på att det redan fanns planer på att införskaffa ett redan färdigt kommersiellt kursutvärderingssystem så vi ansåg inte att det fanns något värde i att vidareutveckla ett till likadant system. Däremot finns det inget system där man kan testa sina kunskaper.

Att vi valde just att skapa ett system där man kan testa sina kunskaper och se hur mycket man kan av det som den kursansvarige anser vara väsentligt beror till stor del på att vi själva skulle använda oss av ett sådant system. Vi är därför övertygade om att även andra studenter och även lärare skulle uppskatta och använda ett sådant system. Vi vill även lära oss mer om hur man använder databaser över Internet.

Vi kommer även att testa och utvärdera befintliga test som finns på Internet såsom; konsumentverkets test om produkter och miljön[10], test om kunskaper för att hålla Sverige rent[11], test om kunskap inom aktiemarknaden[12] och ett IQ test[13]. Detta för att skapa oss en bättre uppfattning om vad som är ett bra respektive mindre bra kunskapstest.

2.1 Diskussion av projektet

Vi planerar att utveckla ett system från grunden som skall vara användarvänligt för båda användarparterna dvs. studenten och den kursansvarige/systemadministratören. Med användarvänligt menar vi att en person med viss erfarenhet av datorer samt Internet ska uppfatta systemet lätt att använda. Designen är väldigt viktig i ett system. Har systemet en dålig design så kommer förmodligen systemet inte att användas i den utsträckning det skulle ha gjorts med en bra design.

De komponenter som ska ingå i systemet är två typer av aktörer som interagerar med ett gränssnitt och en databas. Aktörerna är främst en kursansvarig samt studenterna för kursen men även andra användare kommer att kunna utnyttja kunskapstestet. Detta beror på att vi inte kommer att använda oss av någon typ av inloggning eller säkerhet för studenterna, vilket innebär att även andra användare än just studenter kan testa sina kunskaper. Gränssnittet kommer att vara en HTML-sida med PHP-script som interagerar mot databasen. Som databashanterare kommer vi använda oss av MySQL för att skapa tabeller och för att ställa frågor till databasen. Att vi valt PHP och MySQL beror på att vi använt verktygen tidigare samt att de är fria programvaror. Databasen kommer att lagra all den information som är nödvändig såsom användarnamn, lösenord, svar, kommentarer, frågor, rubrik och ämne.

Vid framställningen av systemet kommer vi att arbeta med prototyper som vi kan vidareutveckla, testa och utvärdera. Ett kunskapstest som är användarvänligt med en lättläst text och bra design anser vi bli välanvänt och därför är det av stor vikt att kunskapstestet följer dessa kriterier.

Systemet kommer att vara uppbyggt utifrån två utgångspunkter, den ena utgångspunkten med hänsyn till den kursansvarige och systemadministratören, den andra för studenten. Den kursansvarige kommer givetvis ha större rättigheter att påverka systemet eftersom det är den kursansvarige som skapar och redigerar ett test och därför måste ha rättigheter att ändra i databasen. Den enda rättigheten en student har är att utföra test. Studenten kommer dock att ha möjligheten att utföra testet hur många gånger som helst. Vi beskriver närmare de olika utgångspunkterna nedan i prototypbeskrivningen.

Vidare så tycker vi att det ska bli både intressant och lärorikt för oss att få större kunskap om design, webbaserad programmering och databashantering. Vi hoppas även att systemet skall komma till användning i framtiden. Kunskapstestet som vi ska utveckla kan ligga till grund för vidareutveckling av ett större färdigt kunskapstest med duggor, mindre tentor på Internet, men främst kommer vi att satsa på att göra ett kunskapstest. Om det blir tid över kommer vi givetvis påbörja en prototyp för duggor.

2.2 Design

Vad är design? Enligt kursen Software Engineering[4] så är design den kreativa processen att omvandla ett problem till en lösning. I designen är systemet skapat och format så att det tillgodogör alla krav. Design är mer än att göra "snygga webbsidor". Design handlar framför allt om funktion. Funktionell design är bra design och funktionell design är oftast vacker och

användbar. God design måste förstås av dess användare och dessutom svara mot användarens behov, om inte har designern misslyckats i sitt uppdrag. Det finns ett oundvikligt men svårdefinierbart samband mellan god funktion och god estetik. Men vad är vackert? Vad är fult? Vad är funktionellt? För detta finns det inga fullständiga svar på. Förutsättningarna för hur design bedöms skiftar. Individuell smak skiftar ännu mer. Men vissa grundregler finns. I systemet skall det vara lätt att göra rätt och svårt att göra fel och saker skall inte vara krångligare än vad de är.

Det finns några aspekter man bör tänka på som utvecklare:

- Det är för användaren man bygger systemet
- VEM är användaren?
    - Vem som helst?
    - Utbildning?
    - Språkkunskaper?
- VAD?
    - Definiera uppgiften som skall lösas
- VARFÖR?
    - Tillämpningen skall underlätta att lösa den givna uppgiften, annars fyller den ingen funktion

Vårt mål för projektet är att följa dessa aspekter och designa systemet så att det passar de flesta användare. Vi har dock tänkt oss att göra en svensk version av prototypen till att börja med. Användaren skall känna igen sig.

2.3 Krav

Kraven kommer att visa hur vi tänkt oss systemets design och funktionalitet. Eftersom vi inte ännu har tillräckliga kunskaper om hur vi ska designa systemet så kan kraven bli ändrade lite under utvecklingsarbetet samt projektets gång.

2.3.1 Generella krav

Systemet ska konstrueras så att det blir enkelt att använda sig av. Samtidigt krävs att användaren har viss datorvana. Den kursansvarige ska på ett snabbt och enkelt sätt kunna

skapa frågor med flervals svar. När den kursansvariga skapat ett test kommer testet att lagras i en databas.

2.3.2 Specifika krav

Endast den kursansvarige och systemadministratören ska kunna logga in på en databas för att skapa frågor samt svar till testuppgifterna. Densamme är även ansvarig för att publicera testet på Internet.

2.4 Pedagogiska aspekter

Hur ska man på bästa sätt visa att en student fått x rätt utav y uppgifter? Det finns tre möjligheter som vi kommer att testa i systemet.

De tre funderingar vi har är:

- Rätt svar efter varje fråga
- Rätta svaren efter ett visst antal frågor
- Rätta svaren efter man gjort färdigt hela testet

Att redovisa efter varje fråga innebär att man måste skicka iväg svaret efter varje fråga och kontrollera mot rätt svar i databasen. Detta kan vara ett störande moment om man sitter hemma vid en vanlig modemuppkoppling. Dock kan man koncentrera sig mer på en fråga i taget och om man svarar fel kan man först läsa om det och lösa frågan så den blir rätt innan man går vidare i testet. Om man redovisar efter ett antal frågor så kan det bli svårt att koncentrera sig på varje enskild fråga. Man blir kanske nöjd om man svarat t.ex. fyra rätt utav fem och då inte bryr sig om att försöka lösa det man svarade fel på. Att ett test går snabbare att utföra kan upplevas av somliga som mer givande. Om vi bestämmer oss för att redovisa resultatet efter det att man gjort färdigt hela testet så blir fördelar och nackdelar ungefär desamma som då man redovisar svaren efter ett antal frågor. Innan vi bestämmer hur just vi ska redovisa svaren till ett test kommer vi att testa och utvärdera andra test som finns tillgängliga på Internet. Vilka test och hur vi gjort utvärderingen kommer vi att redovisa i kapitel 2.5.

Det finns några förslag på hur svaren kan redovisas för användaren.

- Grönt R eller rött √ efter frågan
- En kommentar om hur testet gått
- Redovisa alla svar tillsammans t.ex. 5/10 rätt
- Popup ruta som visar antalet rätt samt en kommentar om hur det har gått

Dessa är bara några alternativ som vi upptäckt efter att ha testat olika kunskapstest på Internet. Vi kommer under projektets gång fortsätta testa andra system för att utvärdera vilket sätt som vi anser vara det bästa sättet att redovisa antalet rätt för det test som vi skapar.

Vi har ingen form av statistik där en kursansvarig får reda på hur studenterna klarar av testerna. Detta beror på att man inte kan erhålla någon vettig statistik eftersom vi inte har någon form av säkerhet. Vem som helst kan utför testen och då blir inte statistiken relevant för den kursansvarige. För att kunna redovisa statistik på testerna måste vi ha en inloggning sida även för studenterna. Detta är inte något som vi planerar att utföra i vår första prototyp utan det får bli något som man kan bygga vidare på om det blir tid över. Det kommer att läggas in en räknare på testet som indikerar om kunskapstestet blir använt.

Det är efter iakttagelser från andra system (se kapitel 2.4) och fackfolk (se kapitel 5.2) som vi kommer att forma testen så att de även blir tilltalande ur en pedagogisk synvinkel.

2.5 Kulturella aspekter

En annan viktig detalj att ta hänsyn till är olika kulturella aspekter. En bock anses t.ex. som fel svar hos oss men kan ha helt motsatt betydelse hos andra. Hänsyn till sådana aspekter bör man ta vid utveckling av större system som har stor spridning.

2.6 Befintliga system

Det finns en mängd Internetbaserade kunskapstest som har helt olika utförande och funktion. Först tittade vi på konsumentverkets webbtest[10] om ekologiska livsmedel. Där har man valt ett 1, X eller 2 system där man rättar testet på slutet. När man rättat testet så får man reda på antalet rätt samtidigt som man bredvid varje fråga får ett grönt R eller en röd bock som visar hur resultatet blev på just den frågan.

Ett annat test vi provade var Håll Sverige Rent[11] där man får svara på 10 frågor om miljön. När man svarat klickar man på rätta och då får man reda på hur många rätt man fått ex. 5 av 10 rätt dock ej vilka frågor man svarat rätt på.

Nästa test vi tittade på var ett test för aktiesparare[12] där man får svara på 10 frågor om börsen. Varje fråga visas en och en i ett popup fönster. När man är klar får man rätta det.

Det sista testet var ett IQ test[13] från USA där man ska svara på 40 frågor. Frågorna visas på två sidor med 20 frågor på varje sida. När man gjort färdigt testet får man reda på vilket IQ man har. Efter att vi provat och utvärderat dessa test har vi bildat oss en egen uppfattning om vad vi anser vara bra design av ett test på Internet. De slutsatser vi kommit fram till är att det lämpar sig bäst att visa alla frågor på en och samma sida och att man rättar testet på slutet efter att man svarat på alla frågor.

2.7 Syfte och målgrupp

Syftet med skapandet av ett kunskapstest har varit att underlätta för studenter vid inlärningsprocessen av vad som är väsentligt i en kurs. Systemet kan också vara till stor hjälp för en kursansvarig som inte får tillräcklig med frågor från sina studenter och därför har svårt att veta hur mycket studenterna förstår. Den kursansvarige kan då hänvisa till testet där varje student enskilt kan testa sina kunskaper och sin förståelse och utifrån detta test komma med frågor till den kursansvarige.

Målgruppen för detta system är både studenterna och den kursansvarige men även allmänheten som vill testa sina kunskaper inom ett visst ämne.

2.8 Kapitelsammanfattning

Vi ska skapa ett system där en kursansvarig kan skapa ett eller flera test till en kurs. Testen ska vara till hjälp för studenter som vill testa hur mycket de har förstått under föreläsningarna samt ett hjälpmedel när man läser på inför en tentamen. Att vi valde att skapa ett kunskapstest beror på att vi själva saknar ett sådant hjälpmedel och vi tror att det kommer att utnyttjas flitigt av kommande studenter, speciellt vid inlärning inför tentamen. Systemet kommer att vara konstruerat med hänsyn till användarna, kursansvariga, systemadministratör och studenter. Det kommer att ha ett användarvänligt gränssitt samt en databas som lagrar information. Säkerhetsmässigt kommer vi att använda oss av en inloggning med användarnamn och lösenord för den kursansvarige men inte någon form av säkerhet för

studenter. Detta innebär att vem som helst kommer att kunna utföra testen. Därför kommer inte någon statistik att redovisas eftersom den kommer att vara helt irrelevant för den kursansvarige. Målgrupperna för detta system är den kursansvarige och dess studenter. En viktig aspekt som vi tar hänsyn till vid utformandet av systemet, är att testa flera andra likvärdiga system för att bilda oss en uppfattning om vad som är användarvänligt och pedagogiskt önskvärt. Vi började med en bred prototyp, dvs. ingen underliggande funktionalitet utan enbart grafiskt design. Detta gjorde vi för att få en bättre känsla för hur systemet ska se ut när det är färdigt samt ett underlag att diskutera förbättringar utifrån.

3 Verktyg

I detta kapitel går vi igenom de verktyg som vi valt för projektet och alternativ på andra verktyg.

3.1 Val av verktyg för detta projekt

Det finns flera anledningar till varför vi har valt språken/verktygen PHP och MySQL. Den största anledningen var att servern vi använde oss av på universitetet stöder PHP och MySQL. En annan anledning är att dessa språk hör till de vanligaste inom respektive användningsområde. Ytterligare en anledning var att vi redan hade baskunskaper i dessa språk. Dessa anledningar gjorde att vi beslöt att använda oss av server-side skriptspråket PHP och databashanteraren MySQL, förutom HTML. Men för uppgiftens skull hade det inte spelat någon roll om vi hade valt någon annan databashanterare t.ex. Access eller några andra server-side skriptspråk som ASP[17] eller CGI[18]. ASP är Microsofts sätt att baka in programkod i HTML-sidor. Common Gateway Interface är mest känt under sin förkortning CGI och är en enkel teknik för att göra HTML-dokument på Internet interaktiva. Standarden beskriver hur externa program kan interagera med webbservrar för att tillhandahålla dynamiskt innehåll.

3.2 HTML

HTML står för "Hyper Text Markup Language". På svenska blir det ungefär "Markeringsspråk för hypertextning". En hypertext är en text som innehåller länkar. Med länkar kan man nämligen knyta ihop olika sidor efter deras innehåll.

HTML är en bra början för den som vill skapa webbsidor. Oavsett vilka andra språk du senare kommer att lära dig eller redan behärskar kommer du alltid att komma i kontakt med html när du skall presentera information på Internet. Att lära sig html kräver egentligen inga som helst förkunskaper eller speciell mjukvara.

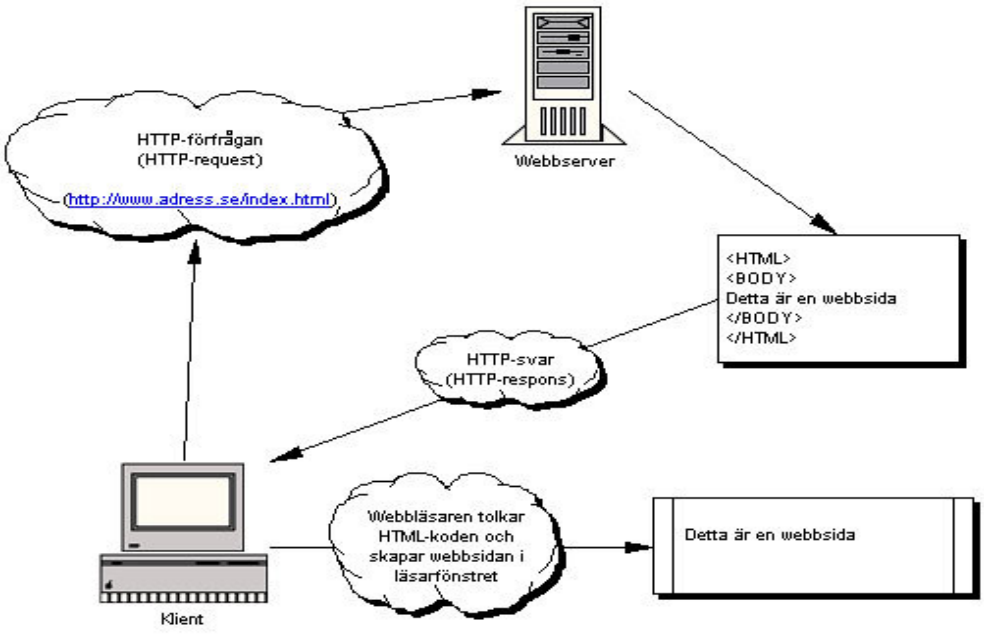
HTML är ett markeringsspråk (även benämnt sidbeskrivningsspråk), dvs. inget renodlat programmeringsspråk. Tanken bakom detta språk är att ett html-dokument ska kunna läsas/tolkas av alla webbläsare oberoende av om besökaren har t.ex. en Mac eller en Pc under

förutsättning att man kodar enligt gällande rekommendationer. Man strukturerar sidan i olika element (exempel tabeller, stycken och figurer). Till hjälp har man olika markeringar så kallade taggar. Som exempel kan vi nämna en lista. För att markera listan använder man starttaggen <UL>, sedan skriver man listan med hjälp av listtaggarna <LI> respektive </LI> och avslutar listan med sluttaggen </UL>. Varje element har alltså en starttagg <tagg> och oftast en sluttagg </sluttagg>. HTML är rena textfiler, som har ändelsen ”.html” eller ”.htm”. Det är sådana sidor som webbläsaren kan tolka och visa på skärmen.

Vad händer med HTML-sidan?

När man efterfrågar (besöker) en HTML-sida händer följande, sett ut serverns perspektiv (figur 3.1).

- Servern läser av anropet från webbläsaren.
- Servern letar upp efterfrågad fil i sitt filsystem.
- Servern skickar denna till webbläsaren som sedan tolkar och visar sidan



Figur 3.1: HTML-beskrivning

3.3 PHP

PHP är ett HTML-inbäddat server-side skriptspråk vilket innebär att programkoden behandlas på en server, till skillnad från JavaScript som kör programkoden i webbläsaren hos klienten. PHP används framför allt när det gäller dynamiska webbtjänster när man har en databas som innehåller information. Förkortningen PHP kommer ursprungligen från Personal Home Page tools, men står idag för PHP Hypertext Preprocesser. Språkets grundare var Rasmus Lerdorf[15]. 1994 när han skrev språket var det för att följa och räkna antalet besökare på sin webbsida. Sedan dess har det kommit ut nya versioner. Andra versionen av språket kom ut 1997. Nu finns det en fjärde version. Viktigt är också att inse att PHP inte gör någonting utan HTML. I webbsammanhang fungerar PHP så att det på ett eller annat sätt genererar HTML-kod. Du kan omöjligen skapa en HTML-sida enbart med hjälp av PHP, utan du skriver ut HTML-kod med PHP. Man kan säga att PHP och andra liknande serverspråk används när HTML inte räcker till.

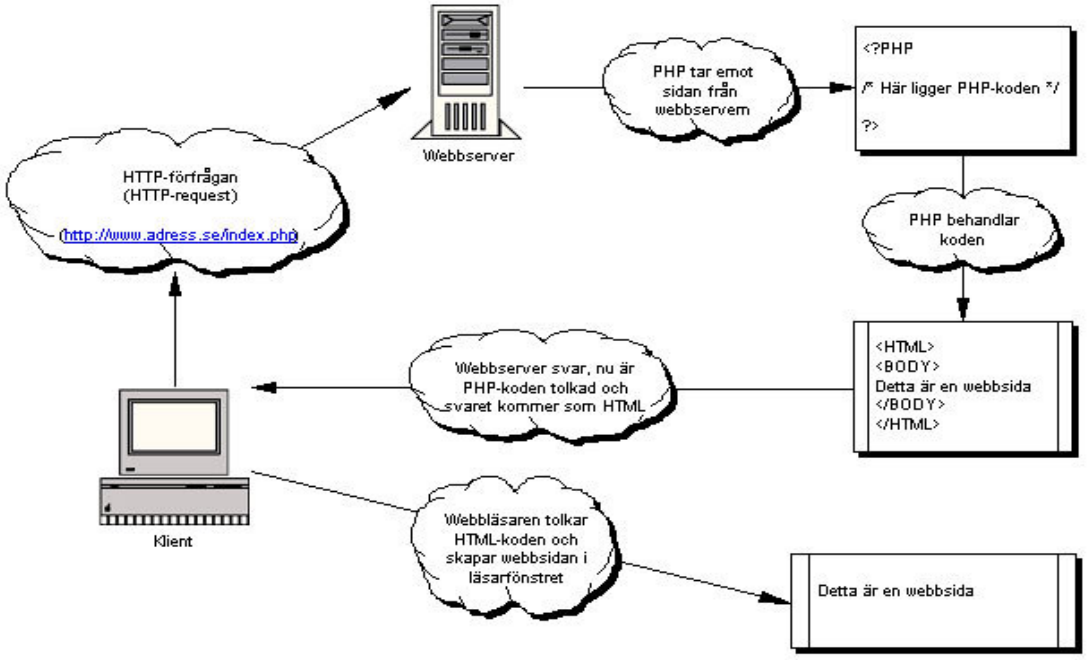
Som vi tidigare skrev så kan man skapa dynamiska webbplatser med hjälp av PHP. Språket har också stöd för flera olika typer av databassystem däribland ett antal SQL-databaser. Några av de stora fördelarna med PHP är att det är lätt att använda, syntaxen är väldigt lik C och Perl. Det finns bra verktygslådor som hela tiden utvecklas, dokumentation finns online, programvaran är fri och dessutom är PHP plattformsoberoende. Genom att koppla databaser till PHP får man möjlighet att skapa dynamiska sidor. Man kan lagra bestående data och manipulera dessa via PHP-skript. Översiktligt kan man säga att PHP möjliggör att webbsidor byggs upp av programkod i samma stund som de hämtas av besökaren.

Vad händer med PHP-sidan?

Skillnaden mellan PHP/HTML och ren HTML är att kod exekveras på servern då PHP används. Detta lägger till ett steg i serverns arbete. Detta händer när man efterfrågar en PHP-sida, sett ut serverns perspektiv (se figur3.2):

- Servern läser av anropet från din webbläsare.
- Servern letar upp efterfrågad fil i sitt filsystem.
- Servern exekverar PHP-koden i denna fil.

Servern skickar den resulterande HTML-koden tillbaka till webbläsaren som sedan tolkar och visar sidan för användaren.



Figur 3.2: PHP-beskrivning

Från figurerna 3.1 och 3.2 kan man utläsa att det som klient inte är någon skillnad på att besöka en HTML- eller PHP-sida. Det man får tillbaka i båda fallen är HTML-kod som webbläsaren tolkar och uppvisar.

3.3.1 Grunderna i PHP

När man skriver PHP-kod så gör man det i en textfil, oftast i samband med HTML-kod. Man måste då tala om för servern var PHP-koden börjar och var den slutar, dvs. där HTML-koden tar vid. Detta kan göras på flera sätt:

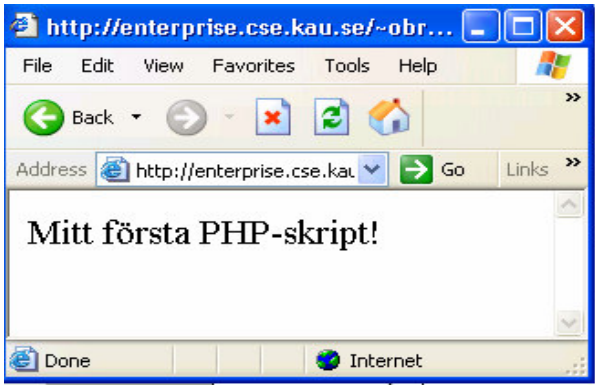
<? /*PHP-kod*/ ?> //Efter dessa tecken kan man skriva in en kommentar

<?php /*PHP-kod*/ ?> /*Mellan dessa tecken kan man skriva in en kommentar*/

<scriptlanguage="php"> /*PHP-kod*/ </script>

Följande exempel och fakta är hämtade ur boken *”PHP programmering 2: a upplagan”* [3]. Ett typiskt PHP-skript kan se ut så här.

```
<?
   echo ”Mitt första PHP-skript!”;
   ?>
```



Figur 3.3: Mitt första PHP-skript

Kommandot echo skriver ut vanlig text på skärmen i detta fall det som står innanför dubbelt citattecken (figur 3.3). Man avslutar varje rad med ett semikolon.

Variabler inom PHP markeras med ett dollartecken framför variabelnamnet $Namn.

I variabeln $Namn kan man nu lagra bokstäver, siffror och understrykningstecken. PHP avgör själv av vilken datatyp variablerna ska vara. Man kan dock göra en typkonvertering för att med säkerhet behandla en variabel som just ett tal eller en sträng ex. $Tal = 4; $Tal = (int) $Tal;.

Det är också viktigt att veta att det blir en viss skillnad i utskriften om man använder dubbelt eller enkelt citattecken runt om strängen (figur3.4).

```
<?
$Namn =  Staffan;
echo ’ Hej $Namn<br>’;
echo ”Hej $Namn”;
?>
```

Figur 3.4: Variabler och Citattecken i PHP

Med variablerna kan man sedan utföra aritmetiska operationer (+, -, *, /), jämförelse operationer (<, >, !=, ==, >=, <=) och logiska operationer (!, ||, &&, and, or, xor).

Vektorer eller arrayer skapar man om man har flera variabler som hör ihop på ett eller annat sätt. Man kan då behandla variablerna i vektorn som en grupp (figur 3.5).

```
<?
$Namn[] = ”Mattias”;
$Namn[] = ”Staffan”;
echo $Namn[0]  ."<br>";
echo $Namn[1];
?>
```

Figur 3.5: Vektor, array i PHP

Mattias finns lagrad först i vektorn på index nr 0 och Staffan är lagrad på index nr 1.

Man kan nu istället för att namnge alla variabler med olika variabelnamn använda sig av samma variabelnamn med ett index nr $Namn[0], $Namn[1] etc.

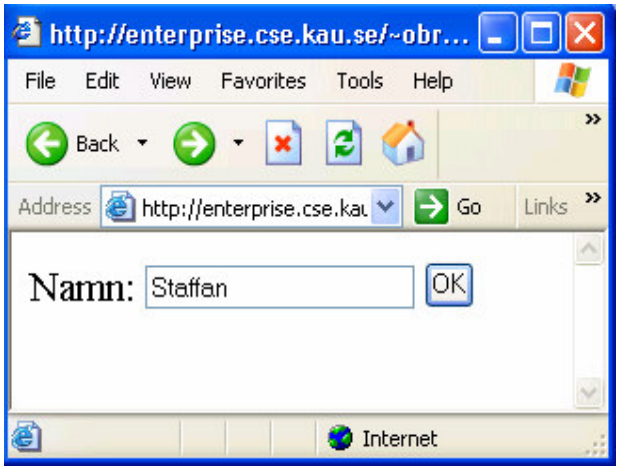
PHP använder sig som många andra programmeringsspråk också av sekvens, selektion och iteration.

Sekvens betyder helt enkelt att kommandoraderna exekverar i den ordning som de ligger det vill säga uppifrån och ned. Selektion står för urval vilket betyder att du kan få valda delar av koden utförda. Iteration betyder upprepning, vilket innebär att vissa delar av koden upprepas till dess att ett speciellt uttryck blir sant eller falskt.

Formulär är en viktig del av webbprogrammering. För att få information från användaren har man ofta ett formulär på sidan. Formuläret kan vara på en vanlig HTML-sida men den sida som tar emot formuläret ska vara en skriptsida av något slag, i vårt fall PHP.

Figur 3.6 visar hur man skickar ett formulär från en HTML-sida och figur 3.7 visar hur man tar emot formuläret på PHP-sidan.

```
<html>
<body>
<form action="sida2.php"
method="post">
Namn: <input type="text"
name="namn">
<input type="submit" value="OK">
</form>
</body>
</html>
```



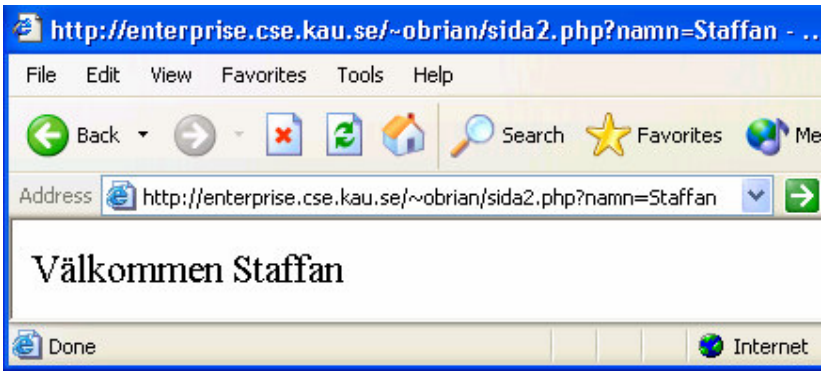
Figur 3.6: Formulär HTML

När man klickar på knappen OK skickas formuläret till den sida som står efter action.

```
<?
echo "Välkommen " . $_POST['namn'];
?>
```

Figur 3.7: Formulär PHP

För att skicka formulär mellan sidor kan man använda sig av method=”POST” eller method=”GET”. När man använder GET så läggs formulärdatan till i URL-strängen (figur3.8).



Figur 3.8: Formulär med GET

Det är därför oftast bättre att använda sig av POST som inte visar någon information i URL-strängen. En fördel med GET är att man kan skicka värden till en sida utan att använda sig av formulär. HTML-koden i den sida som skickar kan se ut så här: Sida 2. Sida 2 blir här en länk till sida2.php och man kommer nu åt namnet Staffan genom $_GET[’namn’].

3.4 MySQL

MySQL är en databashanterare som lämpar sig bra att köra tillsammans med bland annat PHP. En förutsättning för att köra MySQL är att man har någon form av webbserver som

klarar MySQL installerad. Exempel på sådana webbservrar är Apache eller Microsoft Internet Information Server. MySQL är den mest spridda databashanteraren med öppen källkod. Språket utvecklas av det svenska företaget MySQL AB. Det är en förhållandevis enkel databashanterare som är fokuserad på snabbhet och säkerhet. PHP har en uppsättning funktioner avsedda speciellt för MySQL.

SQL står för "Structured Query Language" och är ett frågespråk för att bearbeta och hämta data ur en eller flera databaser. Det är ett relationsbaserat system. Relationsbaserade system sparar data i separata tabeller hellre än att spara all data i en fil. På så sätt får man snabbare åtkomst och större flexibilitet.

Databasservern lagrar data i databaser och databasklienter använder SQL för att bearbeta och hämta data ur dessa databaser. En databasserver innehåller förutom databaser även autentiseringsmekanismer för att hantera ägande och åtkomsträttigheter till de olika databaserna.

3.4.1 SQL-frågor

För att kunna lagra data i en databas så måste man först logga in sig på den server som databasen finns lagrad på. Detta sker med kommandot @mysql_connect($host, $user, $password), $host innehåller namnet på URL-adressen till databasen, $user innehåller användarnamnet och $password innehåller lösenordet. När man sedan är inloggad måste man ställa frågor till databasen för att få fram rätt tabell, kolumn och rad.

Denna fråga tar fram användarnamn och lösenord ur tabellen User och lagrar dem i variablerna $ID_namn och $Pass. Lägg märke till att man kan ha samma variabel namn som det som står i tabellen. Givetvis måste man sätta ett dollartecken framför variabeln.

```
<?
$query =”SELECT ID_namn, Password FROM User
WHERE ID_namn = '{$_POST['NAMN']}'
AND   Password = '{$_POST['PASSWORD']}'";.
$result = mysql_query($query);
$User = mysql_fetch_array($result);
$ID_namn = $User[‘ID_namn’];
$Pass = $User[’Password’];
?>
```

3.5 JavaScript

JavaScript är ett scriptspråk som i grunden utvecklats av Netscape under namnet Livescript och sedan förändrades i samarbete med Sun Microsystems till ett mer javaliknande scriptspråk. Det har samma syntax som Java och påminner om objektorienterad programmering.

JavaScript är ett skriptspråk som körs hos användaren, vilket gör att det krävs att användaren har en webbläsare som klarar JavaScript. Detta är dock inga problem då i stort sett alla webbläsare idag klarar av detta. JavaScript körs med andra ord på användarsidan, inte serversidan som exempelvis PHP gör. Koden som skall köras laddas alltså först ner av användaren och tolkas sedan på klientsidan (användarsidan). Detta innebär att användaren som utsätts för skriptet även kan se koden till skillnad mot då exempelvis PHP används. Vi har använt oss sparsamt av JavaScript. Dock har vi t.ex. använt det för kontroll vid inmatning i formulär, eftersom detta lättast kan göras direkt i webbläsaren. Vi använder oss av JavaScript även för popup-rutor som visas när man klickar på hjälp i programmet. JavaScripts-koden skrivs in direkt i HTML-koden mellan HEAD-taggarna enligt följande exempel:

```
<HTML>
<SCRIPT LANGUAGE =”JavaScript”
        kod skriven I Javascript
</SCRIPT>
        resten av HTML-sidan
</HTML>
```

Funktioner är vanliga i JavaScript och kan skrivas in var som helst i HTML-koden. Vi har till exempel en funktion som heter check_formfield() som kontrollerar att alla fält i formuläret är ifyllda.

```
function check_formfield()
{

    var ID_field = document.logginForm.NAMN.value;
    var Course_field = document.logginForm.PASSWORD.value;
    var ADDSUBMIT;

        if ((ID_field != “”) && (Course_field != “”))
        return true;
```

```
        else

    {
    alert("Du måste fylla i Användarnamn och Lösenord");
    return false;
    }

}
```

När formuläret är ifyllt väljer användaren att skicka iväg informationen genom att klicka på en knapp. Om något fält inte är ifyllt skickas ingen information till servern utan användaren får ett felmeddelande. Detta sker genom att knappen ”Tryck” är kopplad till funktionen check_formfield().

```
<input type="submit" onclick="return check_formfield()" name="submit" value="Logga in">
```

3.6 Kapitelsammanfattning

Det finns ett stort antal språk ute på marknaden idag för att skapa dynamiska webbsidor och även många olika databasspråk. Vi valde PHP som programspråk och MySQL som databas till vår prototyp. Förutom HTML och JavaScript så valde vi PHP som serverspråk och MySQL som databasspråk. Detta var ett naturligt val med hänsyn till våra tidigare kunskaper och på grund av tilldelad serversituation. Vi har också märkt att dessa två program kompletterar varandra ypperligt då det finns specialfunktioner i PHP avsedda för MySQL. En annan anledning är ju att det är fria programvaror. När vi väl fått grepp om språken så var de inte så svåra att använda och det finns mycket referenser på Internet när man stöter på problem.

4 Prototypens design och implementation

För att utveckla webbsidor och program för Internet krävs ett språk, HTML, som kan läsas av webbläsaren. För att presentera enbart statisk information på en webbsida räcker det med enbart HTML. Förr var webben till största delen statisk, HTML-designers skrev ihop webbsidor antingen för hand i en editor eller i en HTML-redigerare och publicerade sedan sidorna. Där låg sedan informationen orörd tills det blev dags att uppdatera sidan. Då laddades sidan hem, redigerades återigen för hand eller via HTML-redigeraren och laddades sedan upp igen. När innehållet var statiskt fungerade detta utmärkt men de senaste åren har webben mer och mer gått mot ett dynamiskt innehåll. Då krävs att man använder ytterligare ett språk gemensamt med HTML, exempelvis PHP som Online Kunskapstestet är utvecklat med. PHP gör det möjligt att hämta information från databaser för att presentera den på webbsidan. Tack vare PHP finns det inget behov att gång på gång redigera statiska HTML-dokument eftersom man kan manipulera databasen som man hämtar information ifrån. Extremt viktigt är det dock att inse att med enbart PHP-kod fås en tom sida utan text och bilder. Därför måste HTML-kod finnas med för att något visuellt skall kunna presenteras för den som besöker webbsidan. I webbsammanhang fungerar PHP så att det på ett eller annat sätt genereras HTML-kod. Online kunskapstest innehåller även JavaScript.

Vi kommer att beskriva PHP mer grundligt och ingående än JavaScript och MySQL eftersom PHP utgör stommen i systemet.

4.1 Prototypbeskrivning

Prototypen består av de delar som man vill att ett färdigt kunskapstest skall kunna göra. I våran prototyp skall vi använda oss av kursen Datastrukturer & Algoritmer, vid institutionen för datavetenskap på Karlstads universitet, för att skapa ett test med frågor och svar. I prototypen kommer vi även att efterlikna den mall som Karlstad Universitetet använder för typsnitt och färger, då internetsidor skapas.

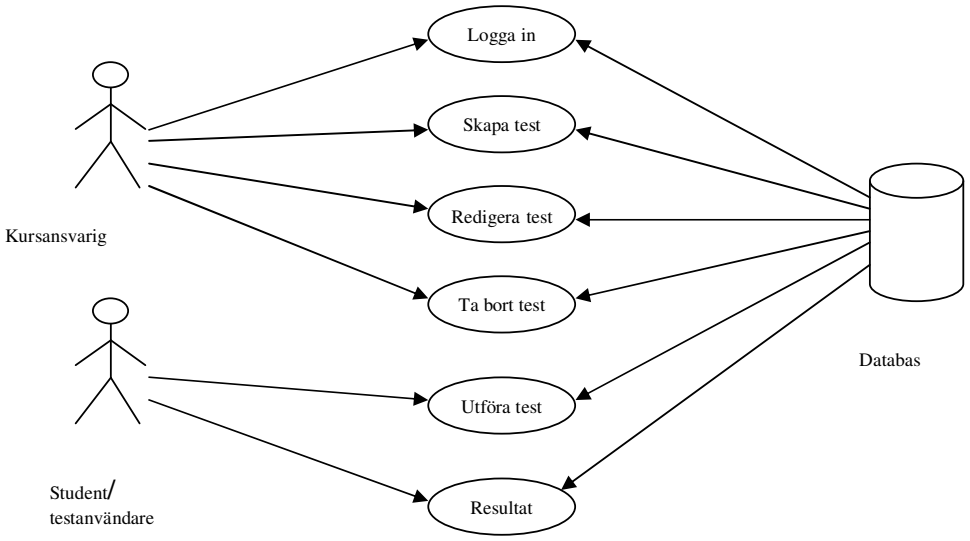
4.1.1 Användargränssnitt

Som vi tidigare diskuterat ska användargränssnittet vara användarvänligt. Gränssnittet skall vara tydligt, lätt att läsa och använda, så att inga manualer krävs. En användare ska uppleva

miljön som behaglig, enkel och välbekant. För knappar och andra komponenter som kommer att ingå i systemet kommer vi att använda oss av Windows-standarden. Vi kommer att följa metaforer, dvs. användaren kan använda tillämpningen utan tidigare erfarenheter av det. När det gäller designen så har vi den uppfattningen att användaren skall känna igen sig och bara genom att titta på systemet veta hur man skall gå tillväga.

Felmeddelanden kommer att visas om man inte har fyllt i formulären korrekt. Detta kommer vi att göra med hjälp av JavaScripts direkt i formuläret.

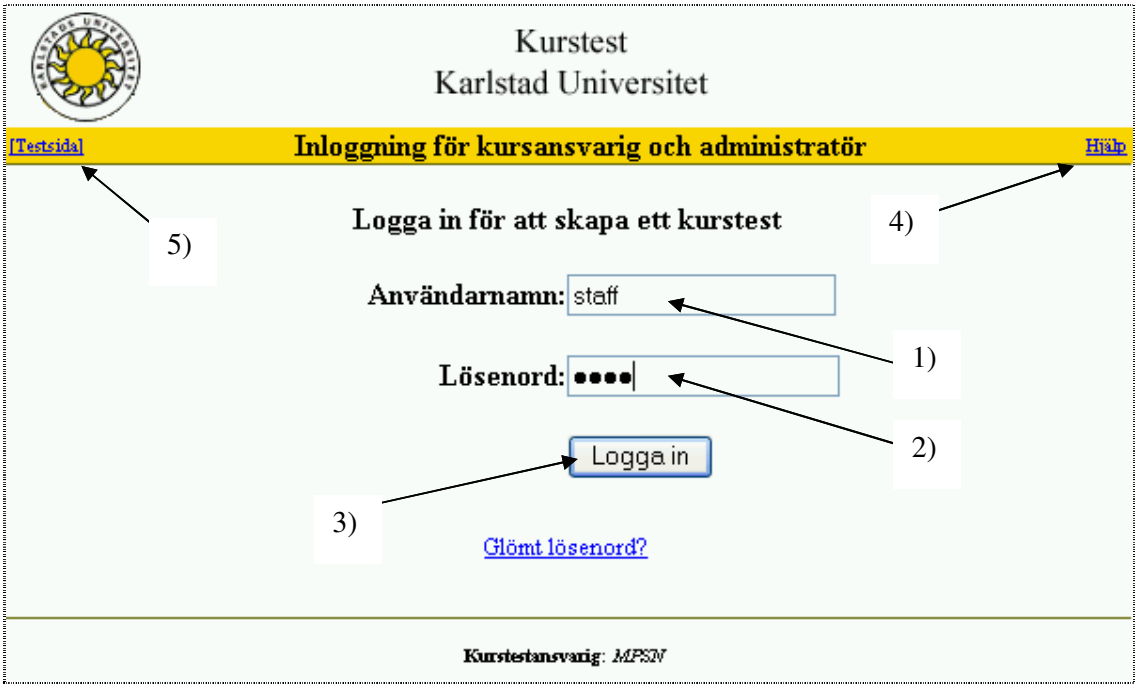
Figur 4.1 visar de användarfall som vi planerar att systemet skall kunna hantera.



Figur 4.1: Use-case diagram

4.1.1.1 **Kursansvarig och systemadministratör**

Prototypen är på så sätt att den kursansvarige och systemadministratören går in på en startsida (Figur 4.2) och där loggar in på systemet. Det gör de genom att skriva in sitt användarnamn (1) och lösenord (2) i inloggningsformuläret. För att skydda lösenordet, syns det bara asterixer i textfältet när man skriver in det (2). När användaren gjort detta så kan de logga in genom att trycka på ”Logga in”-knappen (3). Det finns även en hjälplänk (4) på denna sida om problem uppstår. Från denna inloggningssida kan man även komma till studentens startsida (5).



Figur 4.2: Inloggningssida

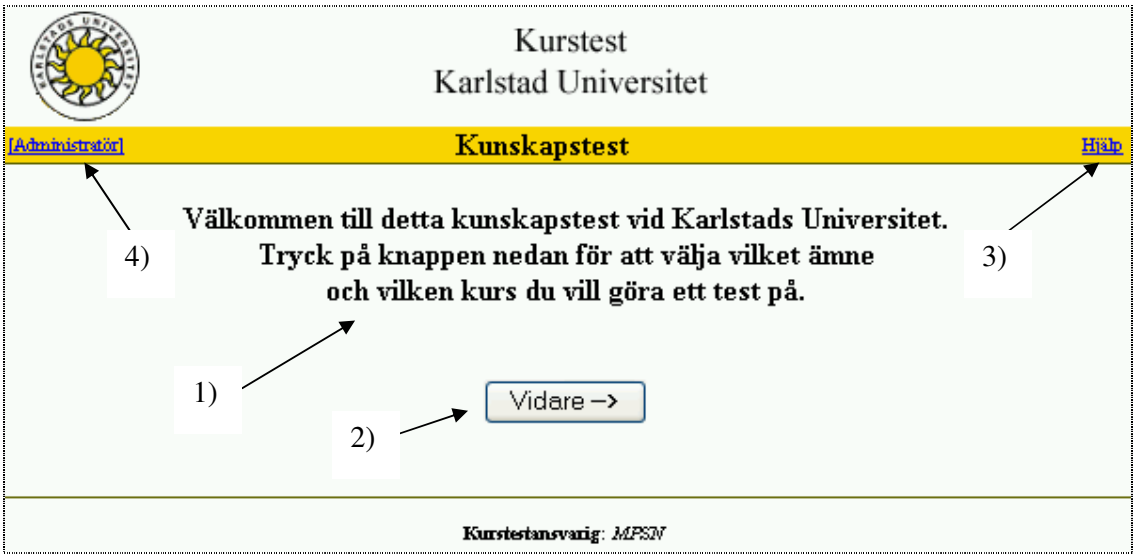
Väl inne i systemet kan den kursansvarige skapa och redigera ett test. Prototypen är designad på sådant sätt att man först skapar en rubrik på testet och sedan frågor med flervalsalternativ, d v s med ”radio” knappar. Designen av prototypen tillåter inte fritextsvar då dessa är alltför komplicerade att analysera. Den kursansvarige skriver in en fråga, de olika svarsalternativen samt rätt svar. Dessutom finns det möjlighet att referera till var man kan läsa om en fråga med en textruta som heter ”Kommentar”. Denna möjlighet finns för att studenten ska kunna tillgodogöra sig information om en felbesvarad fråga. Det finns även en möjlighet att förhandsgranska testet så att man får en överblick och en chans att upptäcka eventuella felaktigheter. När den kursansvarige är färdig med testet och trycker på knappen spara så lagras rubriken, frågorna, rätt svar på varje fråga samt eventuellt en kommentar med läsanvisningar vid fel svar, i en databas. Det finns även alternativ för redigering eller borttagning av ett redan befintligt test.

När systemadministratören loggar in kommer denne till en annan sida, där denne kan lista användare, lista tabeller, redigera användare samt administrera kurser. I ”redigera användare” kan administratören lägga till, ta bort användare, ändra lösenord samt lista alla användare.

4.1.1.2 Student

Studenten kommer att kunna utföra ett test, på den kurs studenten läser, skapat av den som är kursansvarig. Detta test kommer studenten att kunna utföra hur många gånger som helst. För

studenten kommer det inte att finnas någon säkerhetsanordning som inloggning, detta för att testet endast är till för studentens räkning och inte ingår i någon tenta eller slutbetyg. Om man vill att testet ska ingå i kursen för examination måste man skapa någon form av säkerhet av typen inloggning. Till testet kommer studenten via en länk från kurshemsidan. Startsidan (figur 4.3) innehåller en välkomsttext (1) och en knapp (2) för att börja testet. Vidare så finns där en hjälplänk (3) och en administratörslänk (4), där användaren kommer till inloggninssidan för kurstestadministaratören.



Figur 4.3: Studentens startsida

När man gjort klart testet trycker man på skicka som skickar iväg svarsformuläret till databasen där det kontrolleras. Databasen skickar sen resultatet till studenten som får reda på antal rätt samt en kommentar, vid varje fråga där man svarat fel, om var studenten kan skaffa sig den information som behövs för att lösa uppgiften. I ”kommentarer” kan man skriva vanlig text eller hänvisa via en länk. För att lägga till en länk krävs dock lite html-kunskaper som vi beskriver under hjälplänken (se 3) i figur 4.3).

4.1.2 Databas

Systemet ska presentera rätt svar och läsanvisningar om man svarat fel vid testen. Denna information visar hur många rätt samt fel en användare av kunskapstestet har fått. Svaren hämtas från en databas, genom olika frågor man ställer mot den, för att sedan presenteras för användaren. Databasen kommer även att innehålla information om kursansvarigas användarnamn och lösenord. När en kursansvarig gör ett test behöver han/hon inte från början veta exakt hur många frågor som testet ska innehålla utan testet kommer att vara dynamiskt.

Databasen innehåller fem tabeller som vardera har en primärnyckel samt en främmande nyckel som binder samman alla tabellerna. Några av de attribut som databasen ska innehålla är användarnamn, förnamn, efternamn, lösenord, frågor, svar, kommentarer, rubriker och kursnamn.

4.1.3 Säkerhet

Säkerheten är viktig för alla system Extra viktigt är det när man använder sig av Internet för att kommunicera med databaser. Den säkerhet vi kommer att använda oss av är främst en inloggning av en kursansvarig till databasen. Eftersom den testprototyp vi framställer kommer att vara tillgänglig för alla som vill prova, kommer det inte vara någon form av säkerhet för användaren på Internet. Om vi ska bygga vidare på detta system och vidareutveckla ett system där studenterna kan göra duggor, mindre tentor på Internet, måste vi skärpa säkerhetskraven även för användaren.

4.1.4 Hjälpanvisningar

Vidare så har vi på varje sida där vi tycker att användaren kan behöva hjälpanvisningar om systemet infört en hjälplänk. Detta har vi gjort med hjälp av JavaScript, så att användaren får upp en liten popup-ruta när denne klickar på länken. I denna ruta har vi skrivit anvisningar om just den sida som användaren är på. Vi har försökt förklara så utförligt som möjligt om problem som vi tror kan dyka upp.

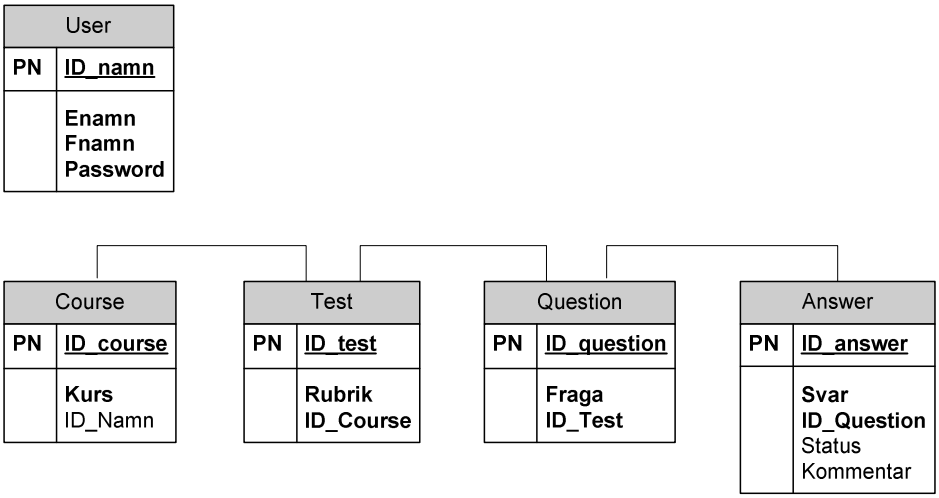
4.2 Databasbeskrivning

Med ordet databas menas att man har en samling relaterade data, som modellerar en del av världen, t.ex. ett företag och dess verksamhet samt att informationen finns lagrad och inte försvinner när man avslutar ett program eller stänger av datorn. Några fördelar med att använda sig av en databas är att flera personer samtidigt kan få tillgång till samma information. Databashanteraren ser till så att inga krockar sker om två personer samtidigt skulle ändra på samma information. Med hjälp av databaser kan man minska redundansen, förbättra säkerheten och ha flera olika gränssnitt.

4.2.1 Kunskapstestets databas beskrivning

Databasen har vi utformat på ett enkelt sätt med hjälp av fyra tabeller kopplade till varandra, plus en tabell med all information om användarna. Varje tabell har minst tre kolumner, en primärnyckel med ett unikt ID nummer, en kolumn där information lagras som t.ex. frågor

och svar. Det finns också en kolumn med en främmandenyckel som innehåller ett ID nummer, detta ID nummer är en primärnyckel i en annan tabell. T.ex. finns i tabellen Course en kolumn ID_course med ett unikt ID nummer. Samma ID nummer återfinns i kolumnen ID_Course i tabellen Test. På så vis kan man koppla samman tabellerna med hjälp av databashanteraren MySQL. För att förtydliga och underlätta arbetet med databasen har vi använt oss av ett databasmodelldiagram (figur 4.4) som visar alla tabeller samt kopplingen mellan dem. Diagrammet visar att tabellen User inte har kopplats samman med tabellen Course, detta är något som bör göras i en framtida vidareutveckling av systemet (se kapitel 6.3).



Figur 4.4: Databasmodell diagram

PN står för primärnyckel. Främmandenyckeln i tabellerna är de som också börjar med ID_. Att några attribut har fet stil beror på att dessa måste innehålla ett värde och de får inte vara NULL. Figur 4.4 visar att tre stycken attribut som inte behöver innehålla ett värde. Status visar vilket svar som är det rätta. Kommentarer behöver inte lämnas utan är frivilliga. ID_Namn är skapat endast för vidareutveckling av systemet.

Databasens tabeller är User, Course, Test, Question och Answer. När Systemadministratören lägger till användare så läggs ID_namn, Fnamn, Enamn och Password till i User-tabellen. Om användaren går in och skapar en ny kurs så läggs ID_course och Kurs till i Course-tabellen osv. Figur 4.5 visar en illustration över tabellerna i databasen och dess innehåll som vi använder. Man kan följa och se hur tabellerna kopplas samman från ”ID_course DAVB01” ända ner till svaren på frågan ”Vad är en mängd” från ”Test 1”, ”on” i status fältet berättar att svars id ”ID_A_00001” är rätt svar.

User

ID_namn	Fnamn	Enamn	Password
einar	Mattias	Pehrsson	ella
staf	Staffan	Nilsson	stafnils

Course

ID_course	Kurs	ID_Namn*
DAVB01	DSA	
DAVC09	C++	

Test

ID_test	Rubrik	ID_Course*
ID_T_001	Test 1	DAVB01
ID_T_002	Test 2	DAVB01

Question

ID_question	Fraga	ID_Test*
ID_Q_0001	Vad är en mängd	ID_T_001
ID_Q_0002	Vad är en datatyp	ID_T_001

Answer

ID_anwer	Svar	ID_Question*	Status	Kommentar
ID_A_00001	En oordnad samling av unika entiteter som har en gemensam egenskap	ID_Q_0001	on	Se kapitel 4.5
ID_A_00002	En samling ordnade unika element	ID_Q_0001		
ID_A_00003	En samling oordnade icke unika element	ID_Q_0001		
ID_A_00004	En ordnad samling av entiteter som har en gemensam egenskap	ID_Q_0001		

Figur 4.5: Databasens tabeller

_ = *Primärnyckel*

* = *Kandidatnyckel*

4.3 Kapitelsammanfattning

När det gäller designen så försökte vi få fram ett så enkelt och lättförståeligt utseende och funktionalitet som möjligt på prototypen. Skulle produkten bli mer komplicerad så finns risken att den inte kommer att användas. Vi försökte även använda oss av färger för att efterlikna universitetets webbplats.

En sak som vi märkte och som kan vara ett stort problem vid utveckling av Internetapplikationer är att olika webbläsare inte visar sidorna likadant. En sida som ser bra ut i Internet Explorer kan se väldigt konstig ut i Netscape till exempel. Detta beror på att de olika webbläsarna tolkar koden lite olika. Man bör alltså under utvecklingens gång testa applikationen i flera olika webbläsare och försöka anpassa applikationen så gått det går.

5 Säkerhetsaspekter

Säkerheten är viktig för alla system. Extra viktigt är det när man använder sig av Internet för att kommunicera med databaser. Det finns åtminstone två sätt att komma åt information som är lagrad på en databas. Den ena vägen är genom det system som använder sig av databasen. Den andra vägen är att använda sig av ett program som kan kopiera, flytta och redigera filerna på en databas t.ex. WINSCP[16].

5.1 Databassäkerhet

I databasen lagras alla användare, deras lösenord samt all annan information såsom kurs, rubriker, frågor, svar, status och kommentarer. Databasen som systemet använder sig av måste alltså även den vara säker. Man bör skapa databasen själv för att se till att man har säkerhet även runt den.

Kunskapstestet finns på en server som Karlstad Universitet tilldelade oss med ett användarnamn och lösenord så att man kan logga in sig på servern. För att sedan kunna lagra information så måste det finnas en plats på servern som kan lagra och hantera data dvs. en databas. Rättigheterna att utföra säkerhetsmässiga åtgärder på servern ligger utanför vår behörighet. Det enda säkerhets mässiga vi gjort för att skydda filerna på databasen är att ändra skriv- och läsrättigheterna till enbart läsrättigheter.

5.2 Systemsäkerhet

I systemet har man olika användare. Den säkerhet vi använder oss av är främst en inloggning av en kursansvarig till databasen och inloggning av systemadministratören som root i databassystemet.

Vi har kommit fram till att studenten inte behöver logga in sig överhuvudtaget, så där krävs det ingen säkerhet. Detta därför att studenten bara skall testa sina kunskaper och att testet inte kommer att vara en del av examinationen av kursen. Vi har konstruerat testet så att studenten enbart har knappar att trycka på och rullgardinsmenyer att välja från. Att vi valt att göra så beror på att studenten inte ska ges tillfälle att skriva in något olämpligt som ska lagras i databasen, t.ex. länkar.

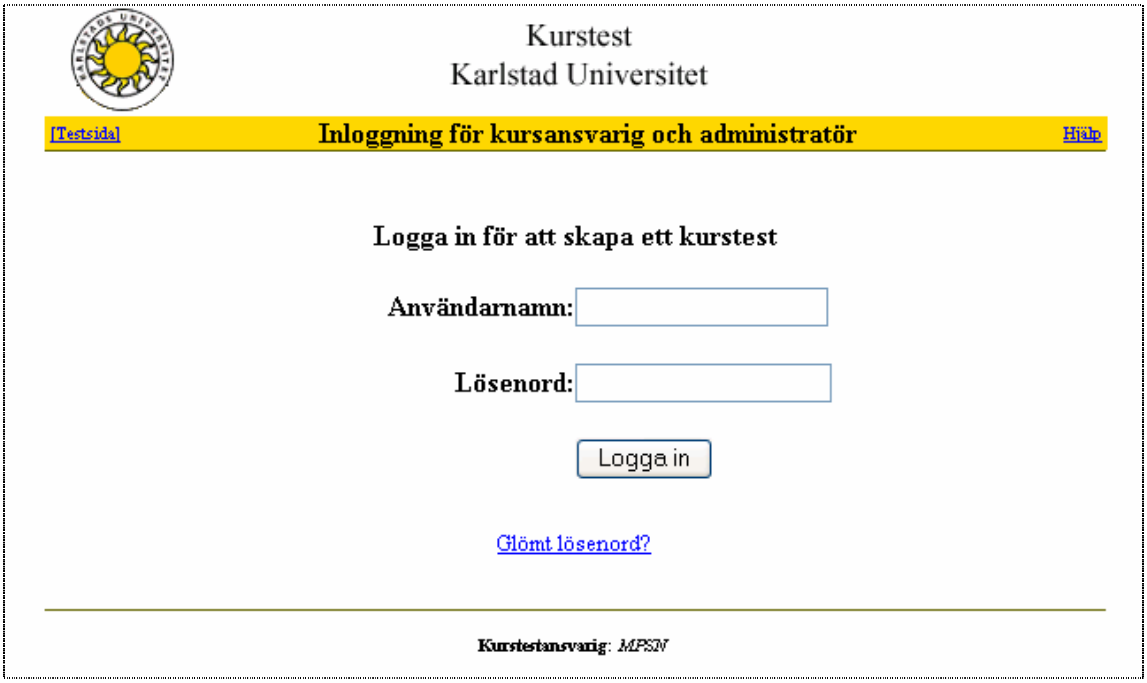
I systemet skall det finnas en systemadministratör som har hand om vilka kursansvariga som skall kunna använda systemet och ger dessa användarnamn och lösenord så att de kan använda systemet. Det är endast den ”rootinloggade” systemadministratören som kan lägga till en ny kurs. Dom rättigheterna som den kursansvarige har är att, kunna skapa ett test med frågor och svar samt redigera ett befintligt test.

För att komma in i systemet så krävs alltså användarnamn och lösenord. När man har loggat in sig i systemet tar sedan varje ny sida emot användarnamn som säkerhetsåtgärd. Det är sessioner som sparar vem som är inloggad när man hoppar mellan sidorna. Det sker en jämförelse mot databasen varje gång en ny sida laddas för att se om användarnamnet är giltigt. Om inte användarnamnet är giltigt så loggas man ut och kommer till inloggningssidan samt ett felmeddelande skrivs ut. På de sidor som administratören enbart har rättigheter till sker ytterligare en kontroll så att en inloggad kursansvarig inte ska kunna komma till de sidorna genom att skriva in en root sida i adress fönstret i webbläsaren. Detta sker genom att kontrollera att den som kommer till sidan är inloggad som root och inte har något annat användarnamn. Systemadministratören kan inloggad komma till alla sidor genom att skriva in sidoadress i adress fönstret i webbläsare.

5.3 Systemsäkerhetsvyer

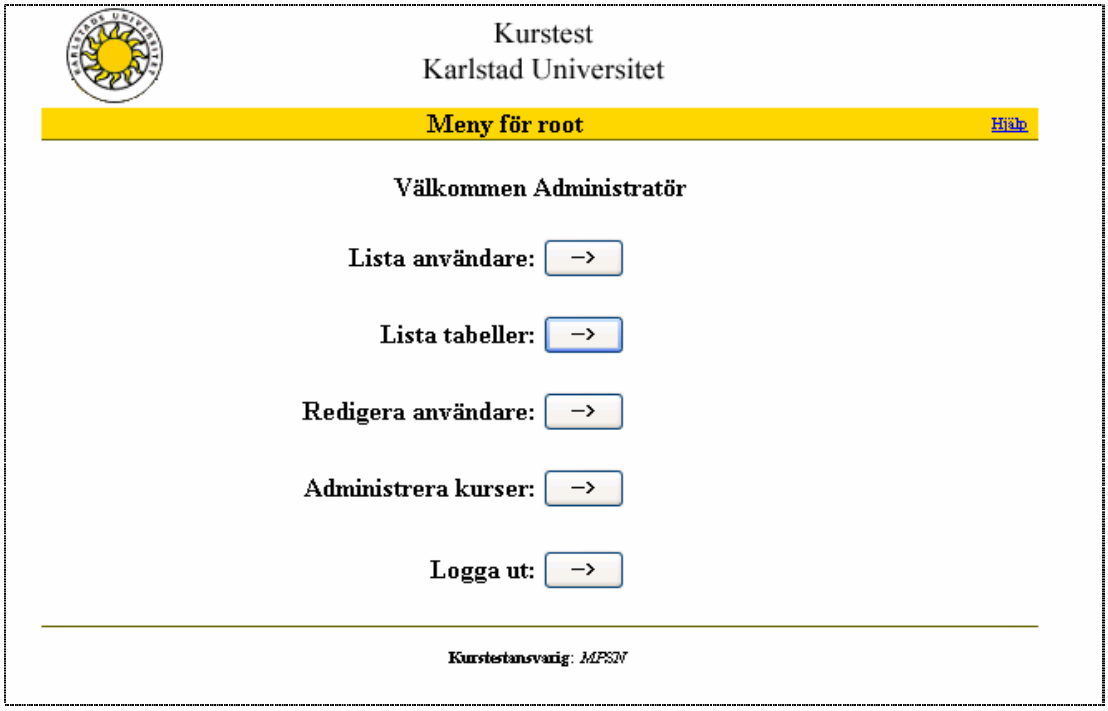
Systemadministratören har rättigheter till allt och kan komma åt all information i databasen. Systemadministratören kan dock inte skapa, ta bort eller redigera test från systemadministratör gränssnittet utan måste som ovan nämnts skriva in sidoadressen i adressfönstret i webbläsaren. Figur 5.1 visar inloggningssidan för systemadministratör och kursansvarig. Figur 5.2 visar Systemansvariges startsida och Figur 5.3 visar Kursansvariges startsida.

Inloggningssida:



Figur 5.1 Inloggningsvy

System administratörens (root) startsida:



Figur 5.2: Systemansvariges startsida

Kursansvariges startsida:

Figur 5.3: Kursansvariges startsida

6 Resultat och utvärdering

Det kunskapstest som vi utvecklat och implementerat är än så länge en prototyp. Med prototyp menas framförallt att produkten kan ha vissa svagheter som måste rättas till om man vill fortsätta att utveckla den till ett helt färdigt kunskapstest. Från första början var det meningen att vi skulle vidareutveckla ett online kursutvärderingssytem, men vi valde efter viss betänketid att göra ett kunskapstest. Från första början var det lite svårt att greppa hur det skulle se ut och vilka verktyg som skulle användas, men efterhand så löste det sig. Vi har valt att utvärdera resultatet kring följande områden:

- Språk
- Användargränssnitt
- Design
- Databas
- Säkerhet
- Pedagogisk synvinkel

6.1 Prototyp

6.1.1 Språk

Prototypen är skriven i PHP och MySQL. Om man tidigare har programmerat i C/C++ och har lite kunskap om HTML så är det inte svårt att komma igång med PHP. När man börjar med PHP märker man snabbt likheter med C/C++. Dock finns en del olikheter, som till exempel att i PHP görs ingen skillnad mellan olika typer av variabler. I MySQL ställer man vanliga SQL-frågor. Har man arbetat med databaser förut så är det inte svårt med MySQL. Vi kände att våra kunskaper inom dessa språk utvecklades under projektets gång.

6.1.2 Användargränssnitt

Användargränssnittet är utvecklat för att vara så enkelt som möjligt. Man behöver inte någon speciell kunskap för att klara av att använda systemet. Vi har eftersträvat detta i vår prototyp och tycker att vi lyckats uppnå ett bra resultat. Det finns tre olika gränssnitt: ett för studenten, ett för kursansvarig samt ett för administratören. En student som använder systemet har enbart knappar och rullgardinsmenyer att trycka på och välja från. Det finns enbart två val för studenten vid varje ny sida, antingen tillbaka till föregående sida eller vidare till nästa sida.

Detta plus en beskrivande text underlättar användandet för studenten. Vi valde att göra användargränssnittet för studenten på detta vis för att förenkla användandet av systemet. Dessutom för att inget olämpligt ska lagras i databastabellerna genom att t.ex. skriva html-kod i formulär. Den kursansvarige har ett annat gränssnitt med text fält som måste fyllas i och fler valmöjligheter. Därför är detta gränssnitt lite besvärligare att använda men det finns beskrivande texter och hjälpmenyer till hands som underlättar arbetet. Systemadministratören har rättigheter att redigera användare samt administrera kurser. Gränssnittet är utvecklat från en rootmeny som visar alla rättigheter som administratören har.

6.1.3 Design och funktionalitet

För att uppnå en hög användbarhet krävs att systemets design och funktionalitet är väl genomtänkta och väl fungerande. Vi har försökt att eftersträva detta under hela projektets gång. Vår prototyp, med avseende på design och funktionalitet, kräver dock lite mer tester för att anses som komplett. Testet har vissa mindre bra egenskaper. Systemet tillåter att flera kan logga in som ”root” samtidigt. Vi har valt att visa texten för rätt eller fel svar alltid efter det första alternativet i svaret. Detta för att vi inte kom fram till någon bra lösning inom tidsramen.

6.1.4 Databas

Databasen kan troligtvis konstrueras på ett annat sätt än enligt just vår lösning. Tabellerna i databasen har en begränsning. Vi har valt att man inte kan lagra mer än 99 olika kurser, 999 olika tester, 9999 olika frågor samt 99 999 olika svar och kommentarer. Detta får inte ta för stor plats på servern. Systemet sparar id nummer genom att titta på det högsta numret och sedan fortsätter id numreringen. Detta kan ställa till problem när man tar bort test eftersom det kan bli stora luckor i tabellen där inga id nummer finns lagrade.

6.1.5 Säkerhet

Den enda säkerhetsåtgärderna som vi har utfört i databasen är att ändra skrivrättigheter på filerna så att endast den som är inloggad på databasen kan utföra några ändringar. Säkerheten i systemet ligger vid inloggning av Systemadministratör och kursansvarig.

6.1.6 Användarsynvinkel

Ur användarsynvinkel så valde vi att redovisa svaren på frågorna som testpersonen gör efter fullgjort test. Vi har även lagt till en hjälplänk på vissa sidor (De sidor som vi anser behöver

det) för att förklara hur man skall använda dem. Vi har försökt att göra systemet så att det är lätt att förstå. För överblick av systemet (se bilaga A).

6.2 Testning och utvärdering av designen

Till hjälp vid testningen av vår produkt har vi haft Monique Pehrsson som arbetar som användbarhetskonsult vid Sigma Information Design AB[14] i Karlstad. Monique testade kunskapstestet och lämnade synpunkter på förbättringar och tillägg (se bilaga D).

Två studenter Joakim Norlinder och Ola Bull har även testat systemet och lämnat deras åsikter om förbättringar. Bland annat ansåg de att vi skulle ha grön text vid rätt och röd text vid fel. Vidare så tyckte de att det skulle finnas en möjlighet att lägga till en fråga till ett redan befintligt test vid redigeringen av testen. De tyckte även att man skulle kunna ändra på rubriken på ett befintligt test. Vi åtgärdade det som gällde färgen på text vid rätt och fel samt funktionaliteten att kunna lägga till en fråga till ett redan befintligt test men inte möjligheten att kunna ändra rubriken på ett test. Att vi valde att inte låta en rubrik ändras berodde på att det var tidsödande att göra samt att vi ansåg att funktionalitet inte var så viktig. När man skapar ett nytt test så kan man ta bort testet direkt om man inte är nöjd med rubriken.

Uppdragsgivaren Donald F. Ross har även testat och utvärderat systemet och lämnat önskemål på förbättringar och tillägg i systemet.

6.3 Framtida utvecklingsarbete

Prototypen som vi har utvecklat kan ligga till grund för framtida utvecklingsarbete. I vår prototyp kan det dock finnas brister i koden och den är lite ostrukturerad. Detta är dock inget unikt för en prototyp. Våran prototyp kan vara till hjälp för fortsatt arbete till ett komplett system med inloggning även för studenten. Man kan då utveckla och implementera ett större säkerhetssystem i systemet så att man kan ha prov och duggor som är en del av examinationen i en kurs. En annan aspekt är att införa statistik i systemet.

Som systemet ser ut idag kan man införa statistik. Man kan till exempel ha statistik på vilka frågor som studenterna har svårast för osv.

Vidare så kan man också göra en variant av prototypen på Engelska också. Detta genom att användaren på startsidan får välja vilket språk (Svenska eller Engelska) som man vill ha.

I tabellen Course finns det en kolumn som heter ID_Namn denna kolumn används inte utan är till för att man ska kunna ta bort rättigheterna att kunna ändra andra kursansvarigas

test. Tiden räckte inte till för att färdigställa denna funktion som vi hade med i planeringen av produkten utan att vi fick begränsa oss till att enbart lägga till en kolumn för att i framtiden kunna använda sig av ID_Namn.

Uppdatering av tabellerna så att inga tomma id nummer i tabellerna finns är något som borde åtgärdas då detta kan bli ett framtida problem.

Knappen ”Ta bort fråga” i redigera.php är ej implementerad till fullo utan det krävs att man tar reda på rätt id nummer för frågan som ska tas bort.

6.4 Kapitelsammanfattning

Prototypen byggdes upp från grunden med ett skriptspråk som heter PHP. Språket var lätt att sätta sig in i, mycket på grund av dess likhet med C/C++, som vi tidigare har programmerat med en hel del. För att kommunicera med databasen så användes MySQL, som också var lätt att förstå sig på eftersom vi i en tidigare kurs fått erfarenhet av databashantering. Användargränssnittet är enkelt så att det skall vara lätt att göra rätt och svårt att göra fel. Systemet måste enligt våran mening vara mer användarvänligt och ej för tidskrävande för att det skall komma i bruk och användas. I databasen lagras alla användare och deras lösenord.

7 Projektsammanfattning

Sammanfattningsvis så har det varit ett intressant arbete att utveckla ett kunskapstest. Ett webbaserat kunskapstest kan vara mycket användbart för en student vid inläsning till tentamen till exempel. Det är även ett användbart pedagogiskt verktyg för kursledaren.

7.1 Detta projekt

Detta projekt gick ut på att utveckla en prototyp för ett kunskapstest på Internet vid Karlstads universitet. I mån av tid skulle vi lägga till någon form av statistik. T.ex. statistik över hur testpersonerna löst varje uppgift, dvs. hur många i procent som har rätt på respektive uppgift eller hur testpersonerna klarat av varje test genom att föra statistik över testresultatet.

7.1.1 Prototyp

De funktioner som vi har i vår prototyp är de mest väsentligaste när de gäller ett kunskapstest. Den kursansvarige och systemadministratören måste logga in. Den kursansvarige kan skapa eller redigera ett test, medan systemadministratören även kan redigera kurstestansvariga. Studenten kan göra valt test. Det finns en hel del funktioner som kan läggas till. Vi kan säga att prototypen är tillfredställande men att det kan göras vissa tillägg.

PHP som vi har valt att programmera i, är väldigt lätt att sätta sig in i, speciellt med tanke på dess likhet med C++, och man kan göra mycket konstruktivt med det. Det har även en del inbyggda funktioner för att kommunicera med MySQL-databasen.

7.1.2 Design och funktionalitet

Designen i vår prototyp är framtagen så att skall vara tilltalande, men ändå enkel. Inga skrikiga färger eller animeringar finns med. Vi har valt färgerna för att användaren skall känna igen sig från Karlstads Universitets hemsida. Funktionaliteten som vi skapat är lätt att förstå och använda. Vi har dock valt att implementera en hjälplänk på de sidor vi tycker behöver det. Vi har även försökt att få design och funktionalitet att inte skilja sig från sådan funktionalitet som en användare är van vid. En annan aspekt som vi försökt ta hänsyn till är att designen och funktionaliteten skall vara anpassad till olika webbläsare, inte bara Internet Explorer.

7.1.3 Säkerhet

Säkerheten i vår prototyp är inte så stor då vi beslutade att studenten inte behöver logga in. Däremot så behöver ju både systemadministratören och den kursansvarige logga in så där har vi implementerat säkerhet gällande inloggning. Även databasen har krävt säkerhet.

8 Slutsatser

En av de aspekter vi ställdes inför vara att avgränsa arbetet, då både vi själva och handledaren kommit på tillägg och ändringar i systemet under arbetets gång. Arbetet med implementationen tog längre tid än beräknat.

Från början var det tänkt att vi skulle vidareutveckla ett befintligt system som utfördes av två tidigare studenter vid Karlstads Universitet. Efter diskussioner med vår handledare och efter hänsynstagande till egna önskemål beslutade vi oss för att utveckla ett eget system.

I efterhand så har vi kommit fram till att vi skulle ha lagt ner mer tid åt förarbete än vad vi gjorde, detta då vi kom fram till många ändringar i systemet under arbetets gång. Vi har upptäckt att det är väldigt viktigt att strukturera upp arbetet innan själva implementationen. Med det menar vi att man bör använda en kodstandard för programmeringen och en stilmall för hur knappar, färger, teckensnitt osv. skall vara. Att strukturingen av arbetet inte blev fullständigt beror till stor del på att vi inte har så stor erfarenhet av större projektarbete. Under projektets gång har vi lärt oss en hel del om databaser, PHP, MySQL, JavaScript och vilka problem som kan uppstå när man arbetar med dessa.

Vi anser att vi nått vårt mål med en funktionell prototyp som är fullt duglig att användas för tester på Internet. Det finns dock lite tillägg och förbättringar som kan och borde göras för att få ett system som är komplett.

Referenser

Böcker och Uppsatser

[1] Geschwind, Daniel; Ogolla, Simon ; Kursutvärderingssystem på Internet, *Karlstad Universitet, Datavetenskap 2002.*

[2] Spolsky, Joel ; User Interface Design for Programmers, Apress 2001.

[3] Ek, Jesper; Overgaard, Jörgen & Eriksson, Ulrika ; PHP Programmering, Pagina Förlags AB 2:a upplagan (2003).

[4] Kompendium i kursen Software Engineering (DAV C19) vid Karlstads Universitet.

[5] Jacobson, Ivar, Booch, Grady & Rumbaugh, James ; The Unified Software Development Process.

Webbadresser

[6] http://www.phpportalen.net/ (040608)

[7] http://www.atiger.pp.se/sida.html (040608)

[8] http://www.javascript.nu/ (040608)

[9] http://www.webbkunskap.com/ (040608)

[10] http://kunskapstest.konsumentverket.se/user/default.asp?testid=7 (040608)

[11] http://www.hsr.se/sa/node.asp?node=245 (040608)

[12] http://www.aktiespararna.se/frameset.asp?page=http://www.aktiespararna.se/utbildning/studiecirklar_och_kurser/kunskapstest/kunskapstest.asp (040608)

[13] http://web.tickle.com/tests/uiq/ (040608)

[14] http://www.id.sigma.se (040608)

[15] http://lerdorf.com/ (040608)

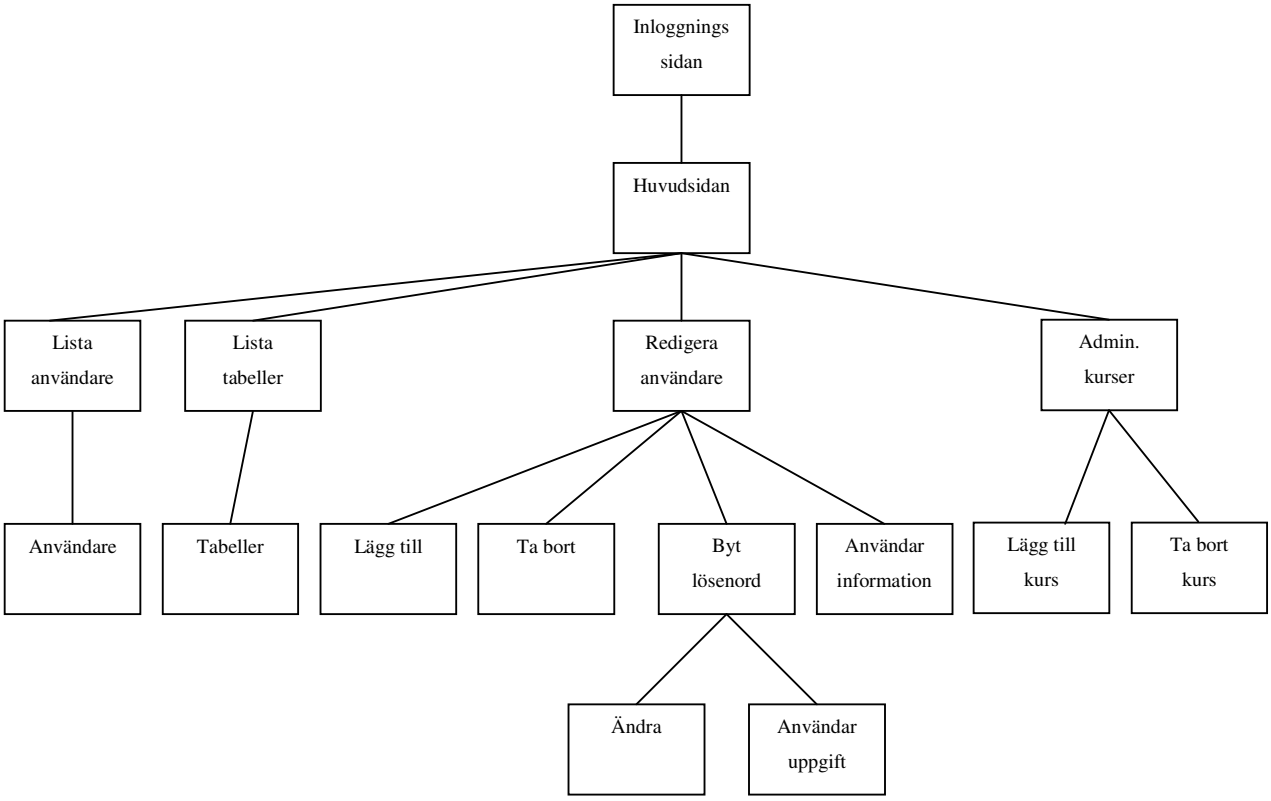
[16] http://winscp.sourceforge.net/eng/about.php (040608)

[17] http://www.aspsidan.nu/ (040608)

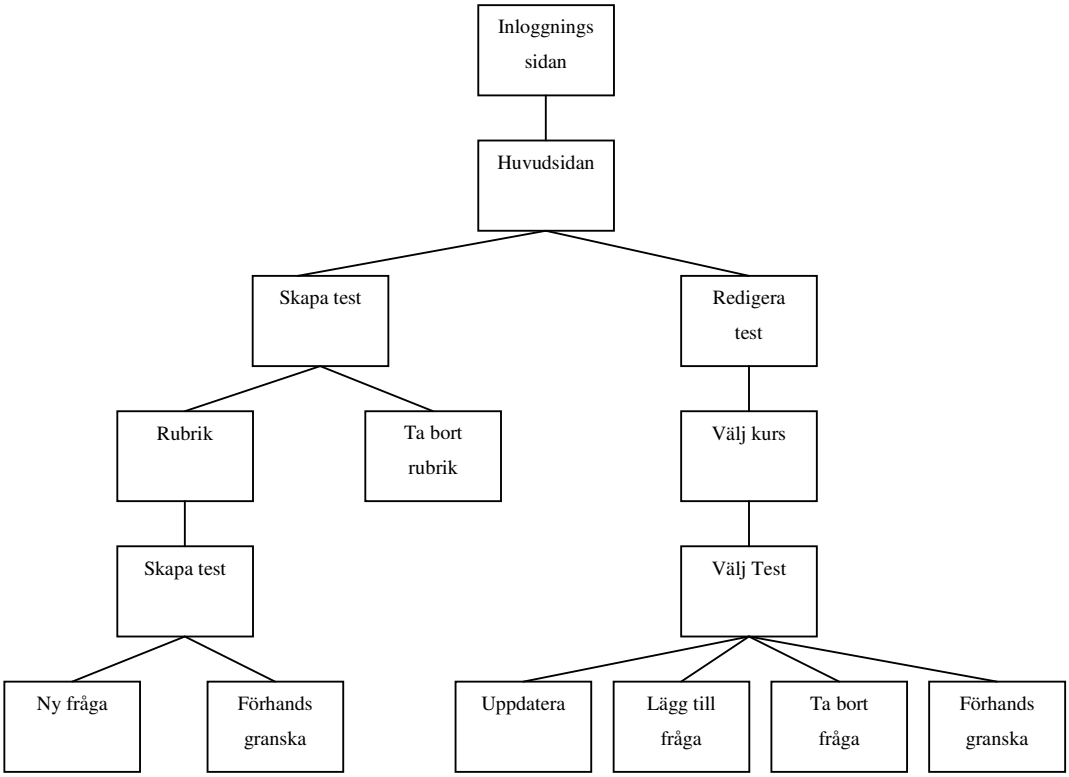
[18] http://cgi.resourceindex.com/ (040608)

A Systemöversikt

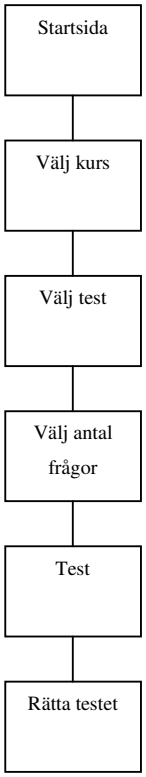
Systemansvarig (root):



Kursansvarig:



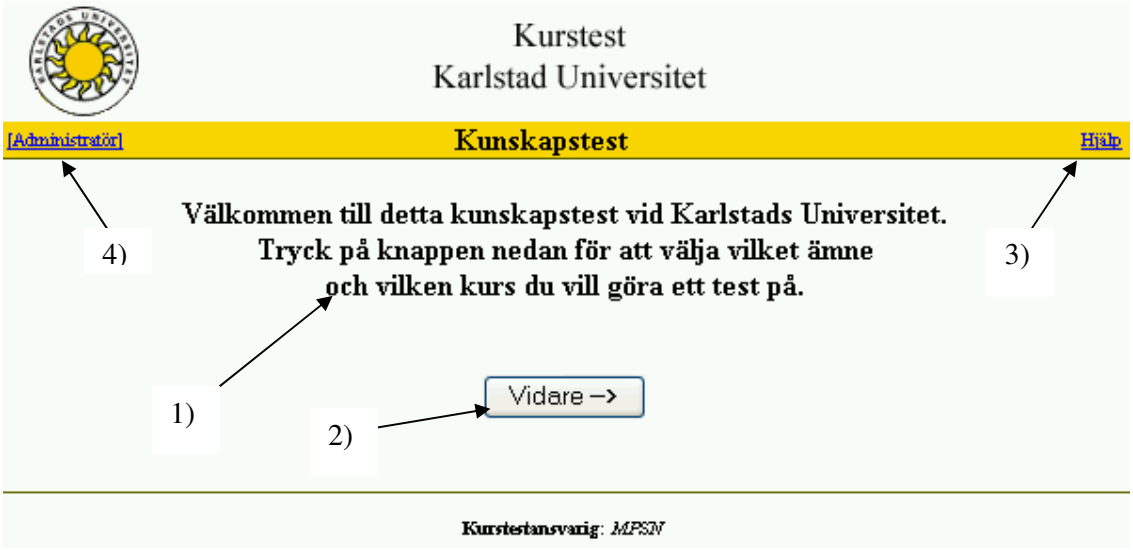
Student:


Startsida
Välj kurs
Välj test
Välj antal frågor
Test
Rätta testet

B Användarmanual

B.1 Student

B.1.1 Startsida



Till testet kommer studenten via en länk från kurshemsidan. Startsidan innehåller en välkomsttext (1) och en knapp (2) för att gå vidare och börja testet. Först måste dock ämne väljas (B1.2). Vidare så finns där en hjälplänk (3) och en administratörslänk (4), där användaren kommer till inloggninssidan för kurstestadministaratören.

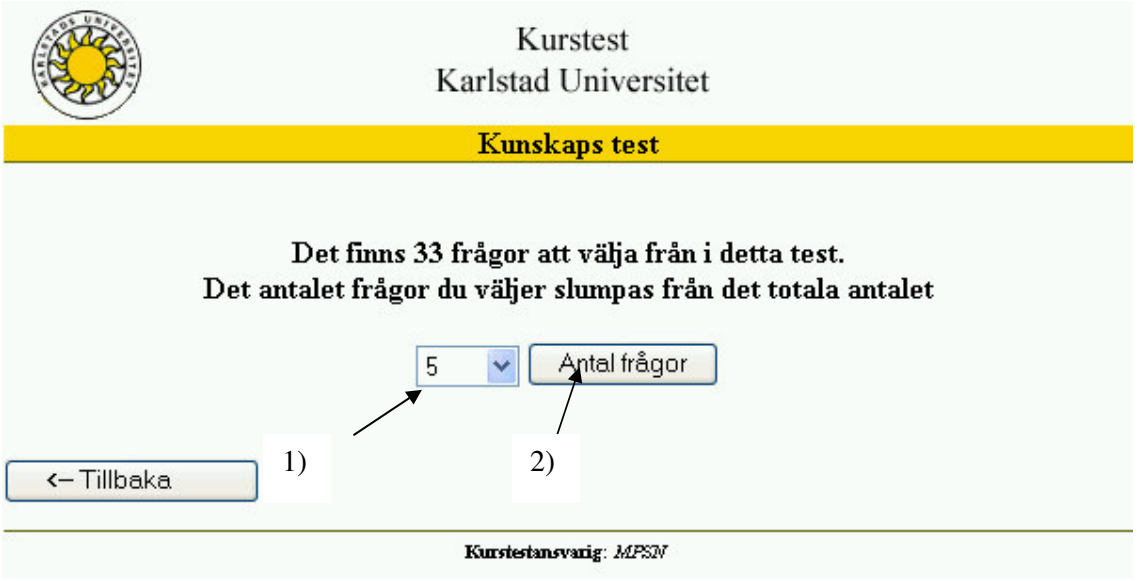
B.1.2 Välj ämne

Studenten väljer vilken kurs (1) och trycker på knappen ”Välj ämne” (2).

B.1.3 Välj test

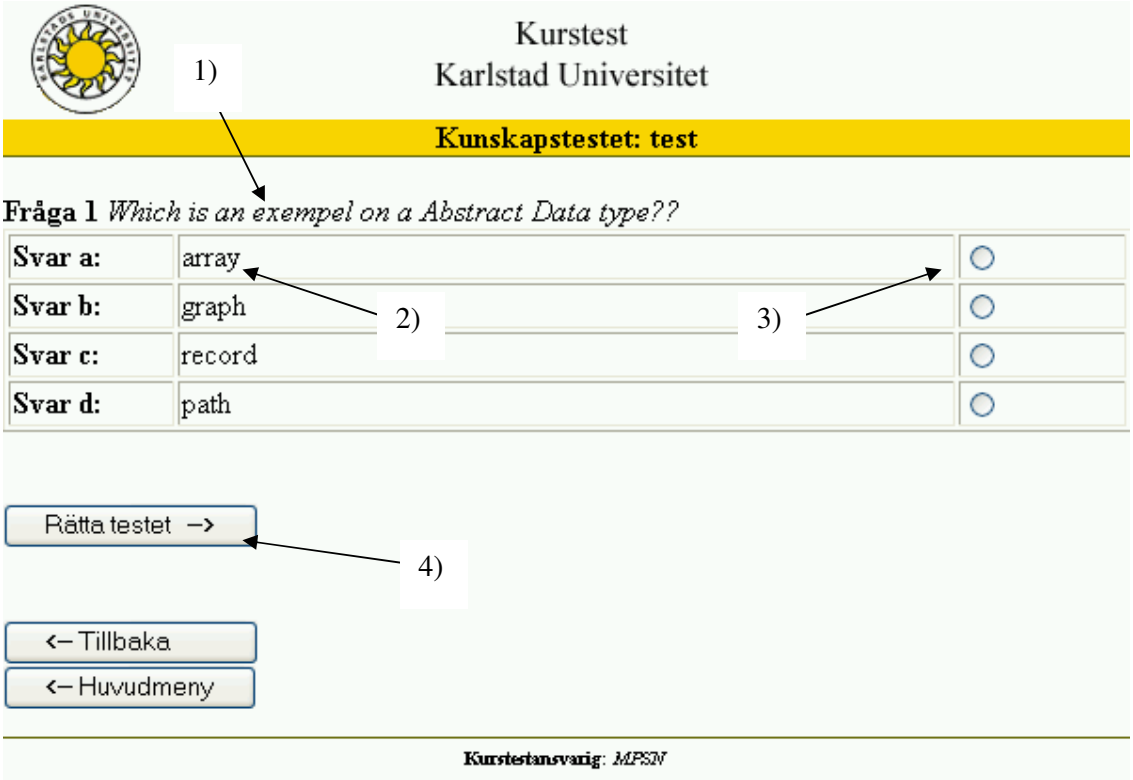
Studenten väljer ämne (1) och trycker på knappen ”Välj test”(2).

B.1.4 Antal frågor



Testdeltagaren väljer antalet frågor som denne önskar besvara (1). Man kan välja mellan 5, 10, 15, 20 samt alla frågor. Dessa slumpas sedan från befintligt test när man trycker på knappen ”Antal frågor”.

Fråga (Exempel)



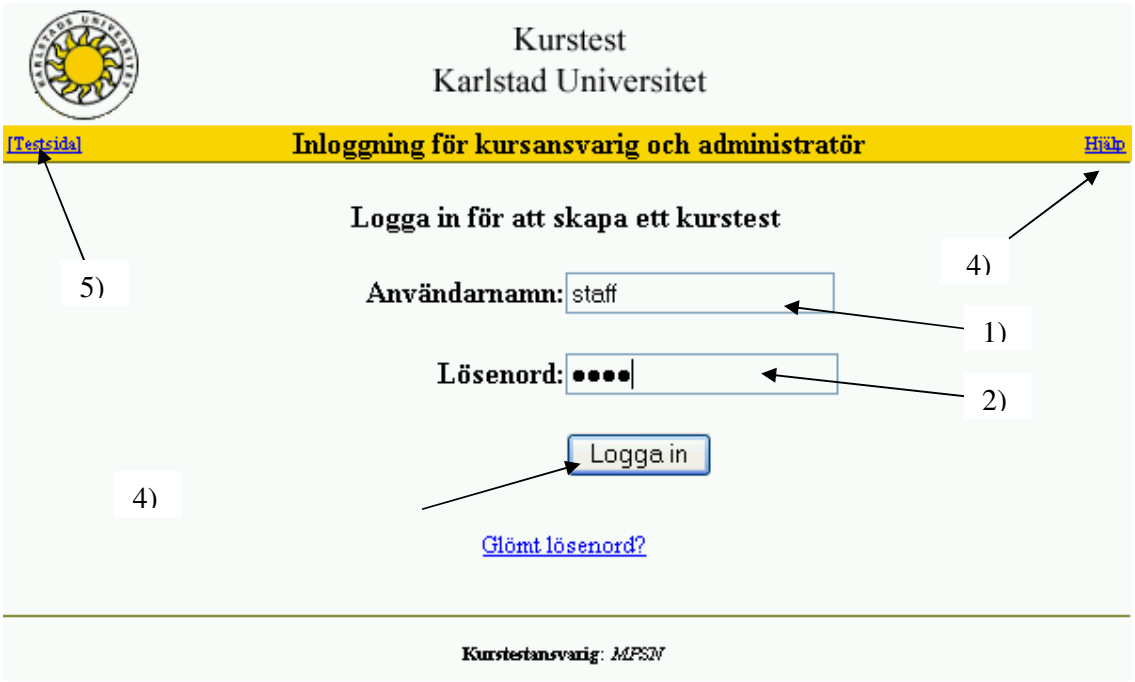
Testdeltagaren kommer till testet och svarar på frågor (1) genom att kryssa i rätt alternativ (2) med motsvarande ”radio”-knapp (3). När testet är klart trycker man på knappen ”Rätta testet” (4) för att få reda på resultatet.

Svar (Exempel)

Kurstestdeltagaren får reda på om svaret på frågan/frågorna är rätt eller fel (1) och hur många rätt man fick (2). Om man svarat fel på en fråga så får man en kommentar (3).

B.2 Kurstestansvarig

B.2.1 Startsida (Inloggningssida)



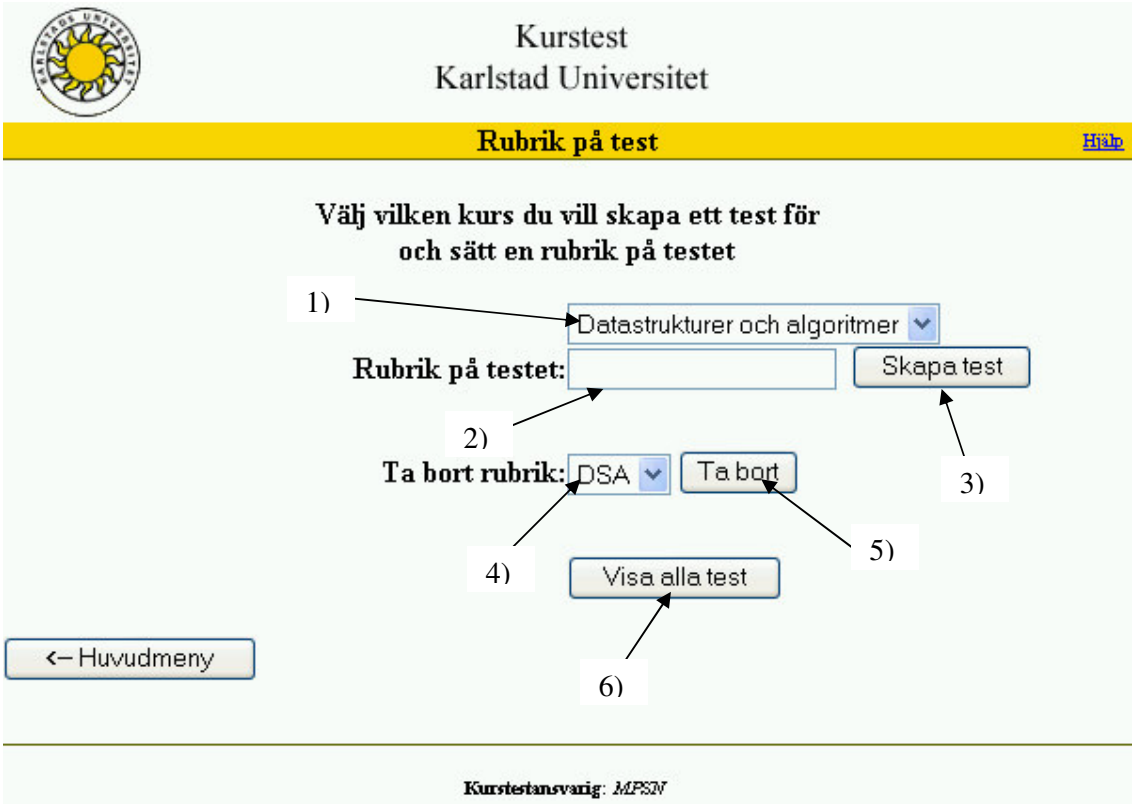
Den kursansvarige går in på startsidan och loggar där in på systemet. Det gör han/hon genom att skriva in sitt användarnamn (1) och lösenord (2) i inloggningsformuläret. När användaren gjort detta så kan han/hon logga in genom att trycka på ”Logga in”-knappen (3). Det finns även en hjälplänk (4) om problem uppstår. Från denna inloggningssida kan man även komma till studentens startsida genom att trycka på länken ”Testsida” (5).

B.2.2 Huvudmeny

Den kursansvarige kan i huvudmeny välja att skapa test (1) eller redigera test (2). Det finns även en ”Logga ut”-knapp (3).

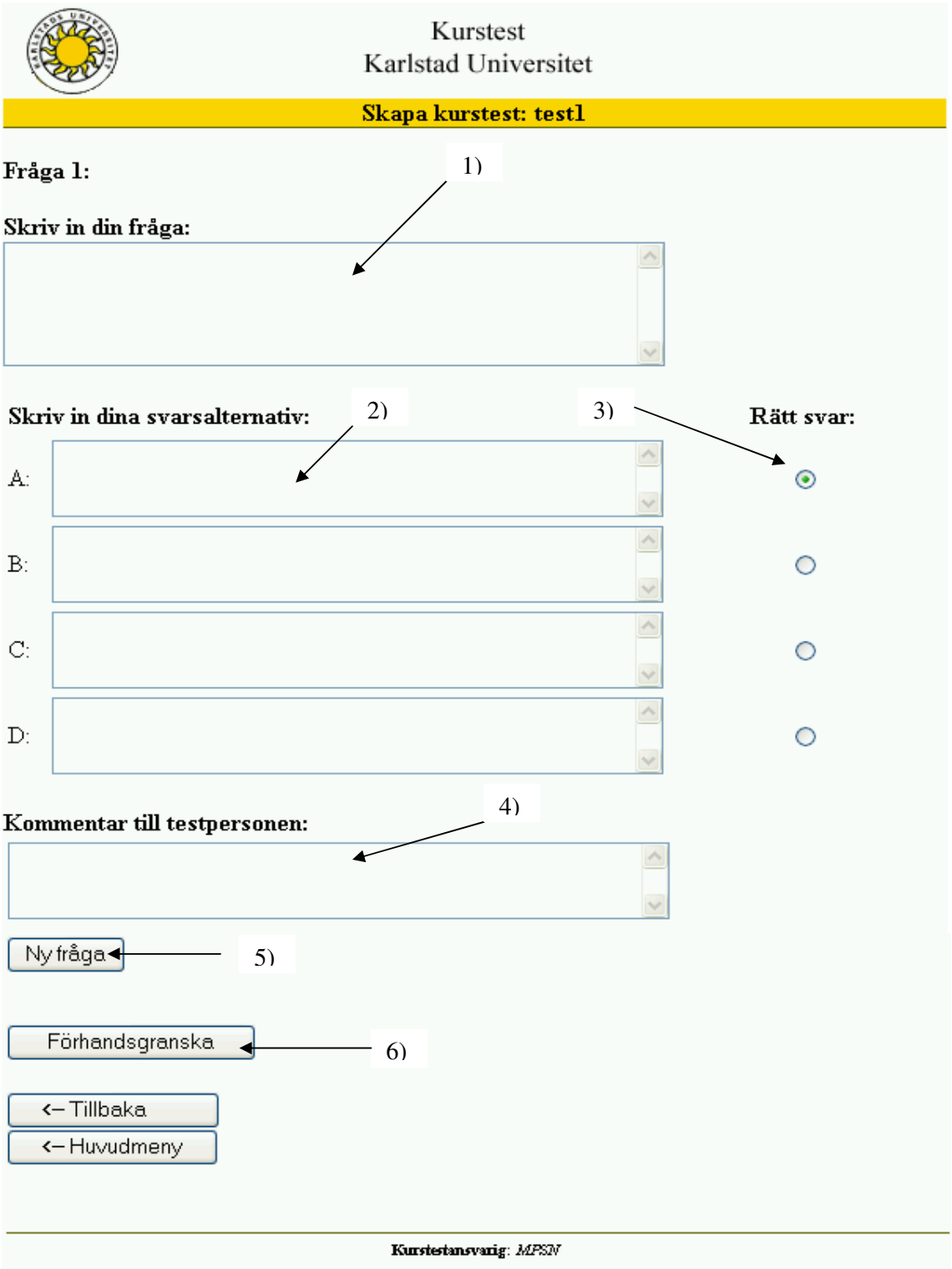
B.2.3 Skapa test

B.2.3.1 Rubrik



Här kan den ansvarige välja kurs (1) och sedan sätta rubrik (2) på testet och trycka på knappen ”Skapa test” (3). Man kan även välja att ta bort rubrik (4) och trycka på knappen ”Ta bort” (5) för att radera ett test. Man kan också lista alla befintliga test (6).

B.2.3.2 Skapa test



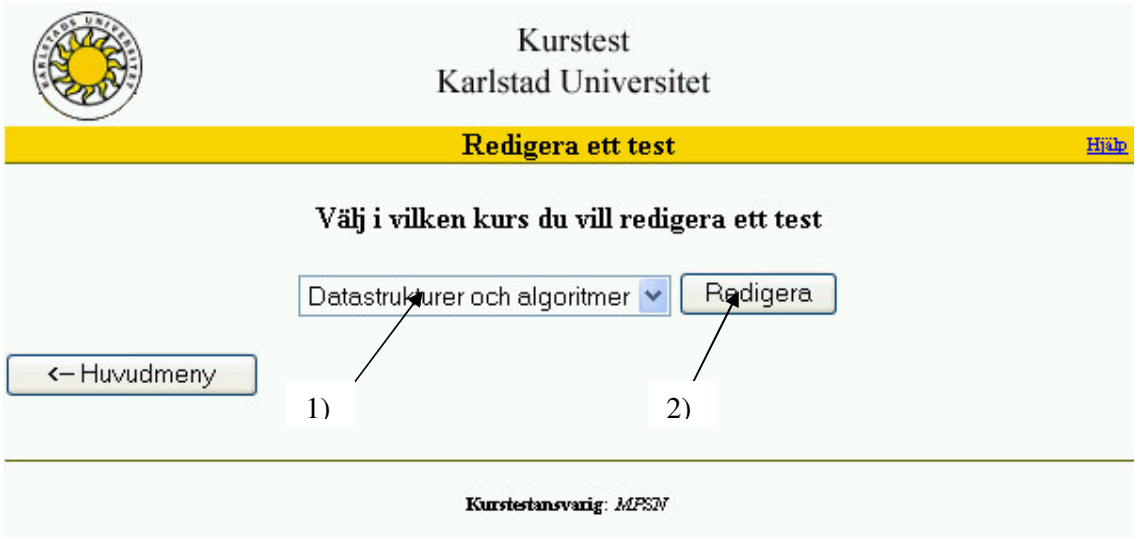
Test skaparen skriver först in frågan (1). Han/hon skriver sedan in 4 svarsalternativ (2) och markerar rätt svar i rätt ”radio”-knapp (3). Vidare så kan man skriva in en kommentar (4) till testpersonen vid fel svar. För att skapa ny fråga så trycker man på knappen ”Ny fråga” (5). När man är klar med testet så trycker man på knappen förhandsgranska (6).

B.2.3.3 Förhandsgranska

Exempel på förhandsgranskning.

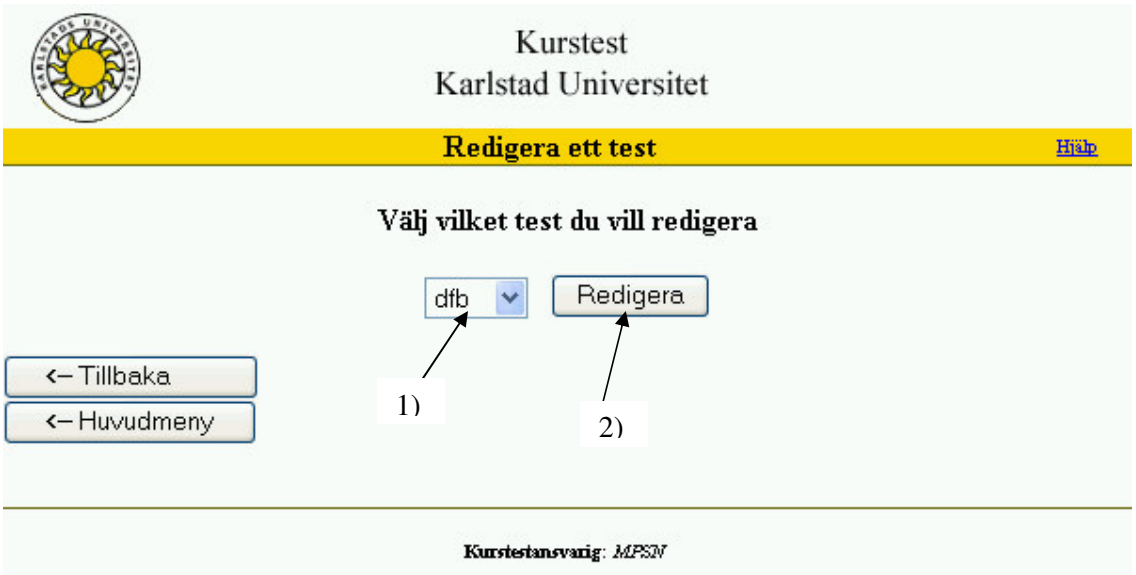
B.2.4 Redigera test

B.2.4.1 Kurs



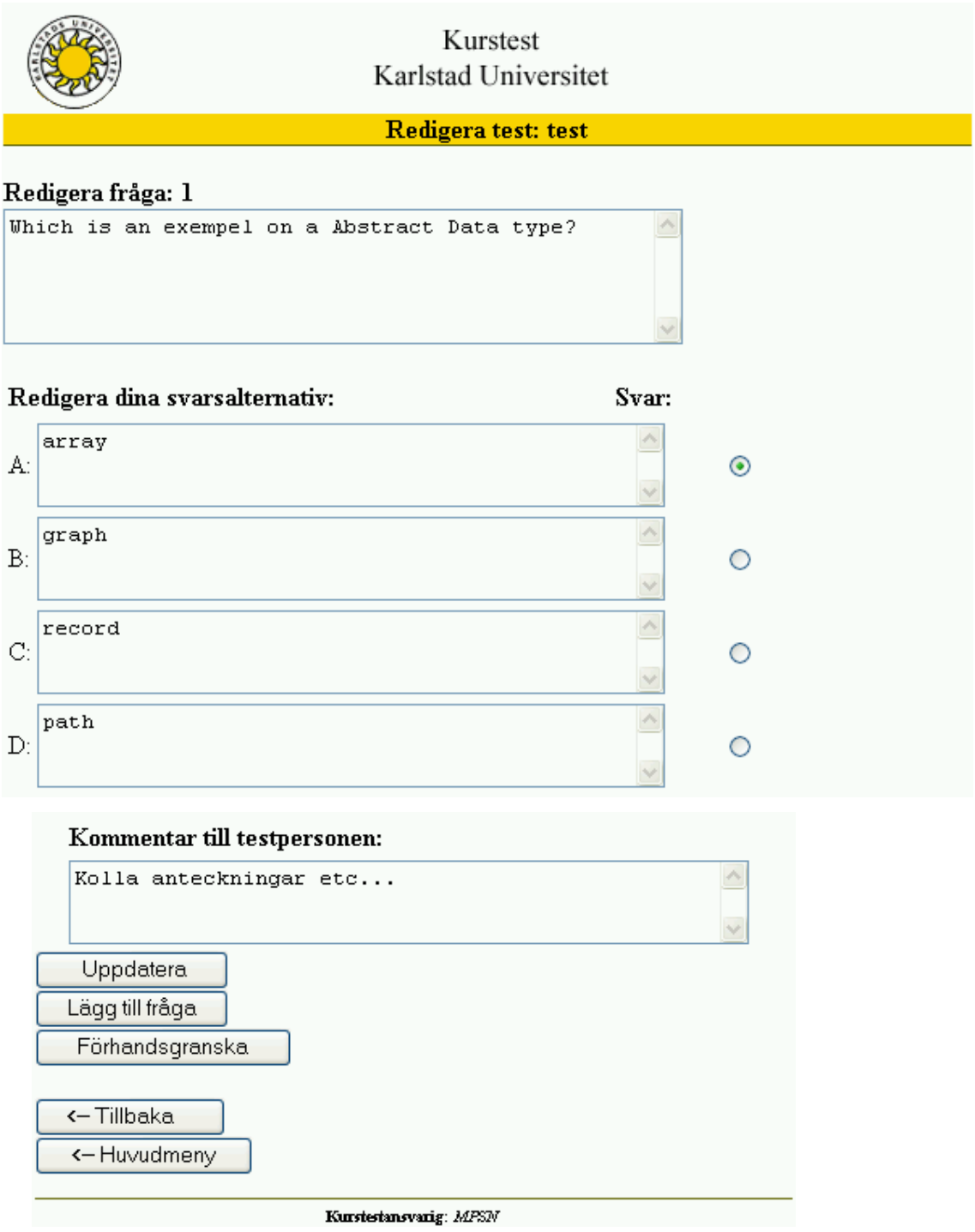
För att redigera ett test så väljer man först kurs (1) och trycker på knappen ”redigera” (2).

B.2.4.2 Test



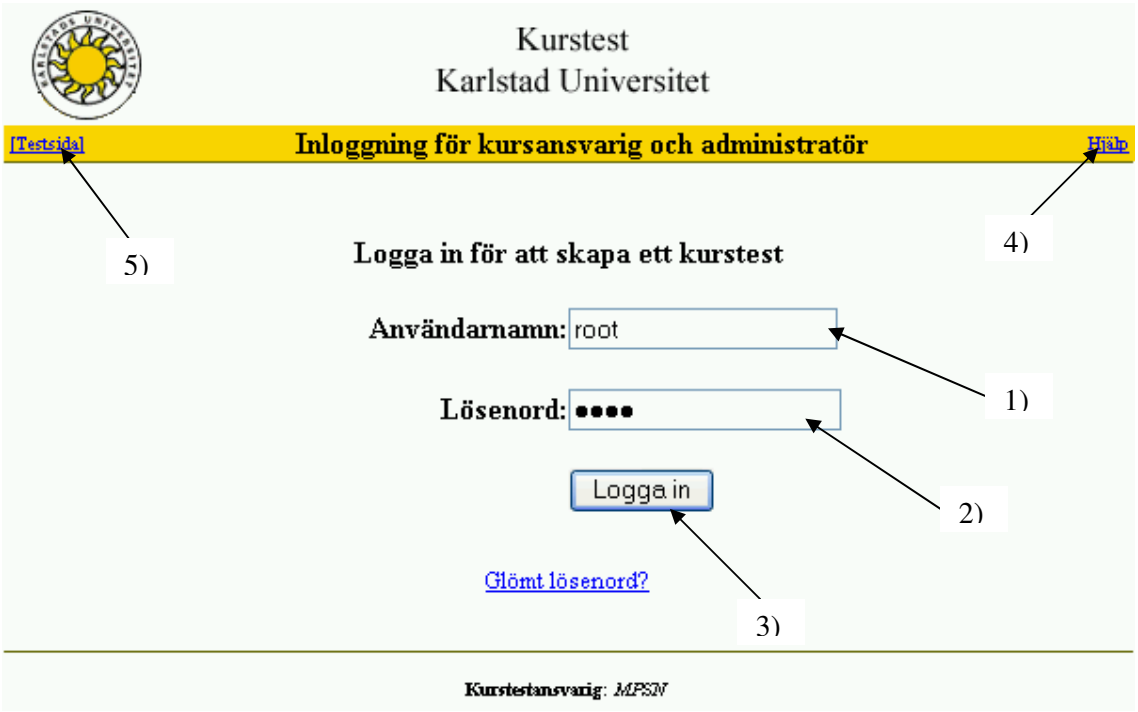
Sedan väljer man vilket test (1) man vill redigera och trycker på knappen ”Redigera” (2).

Redigera test



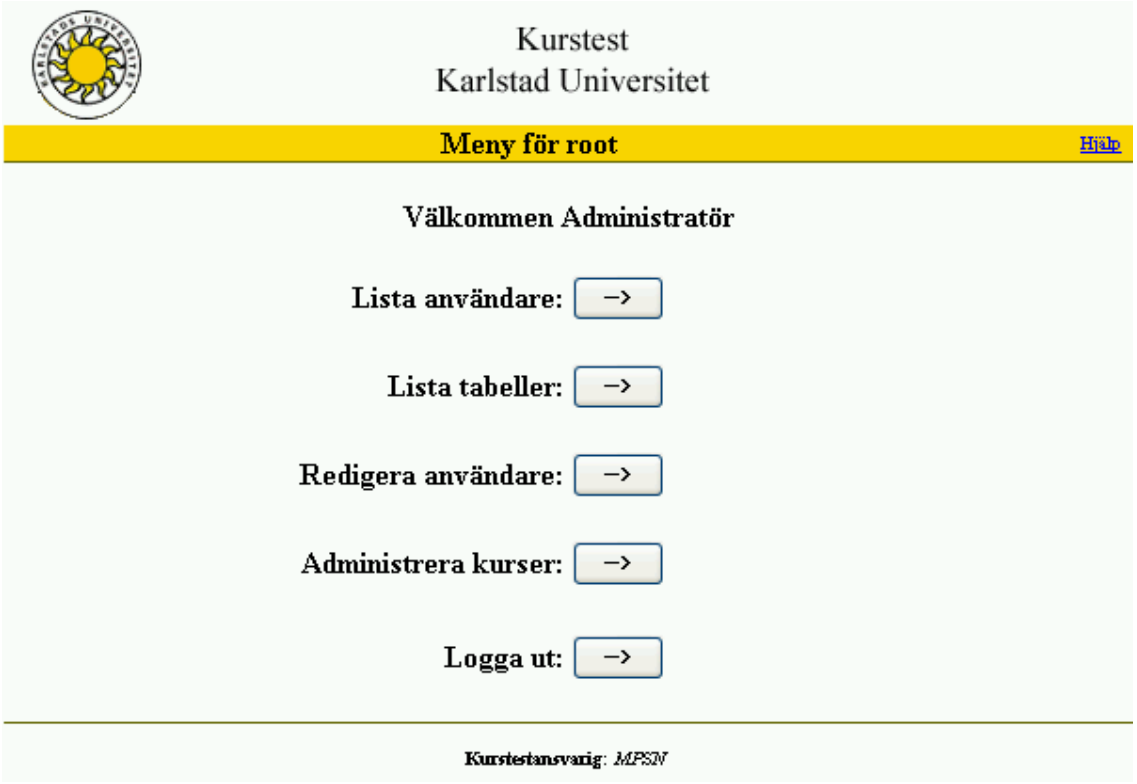
B.3 Systemadministratör

B.3.1 Inloggningssida



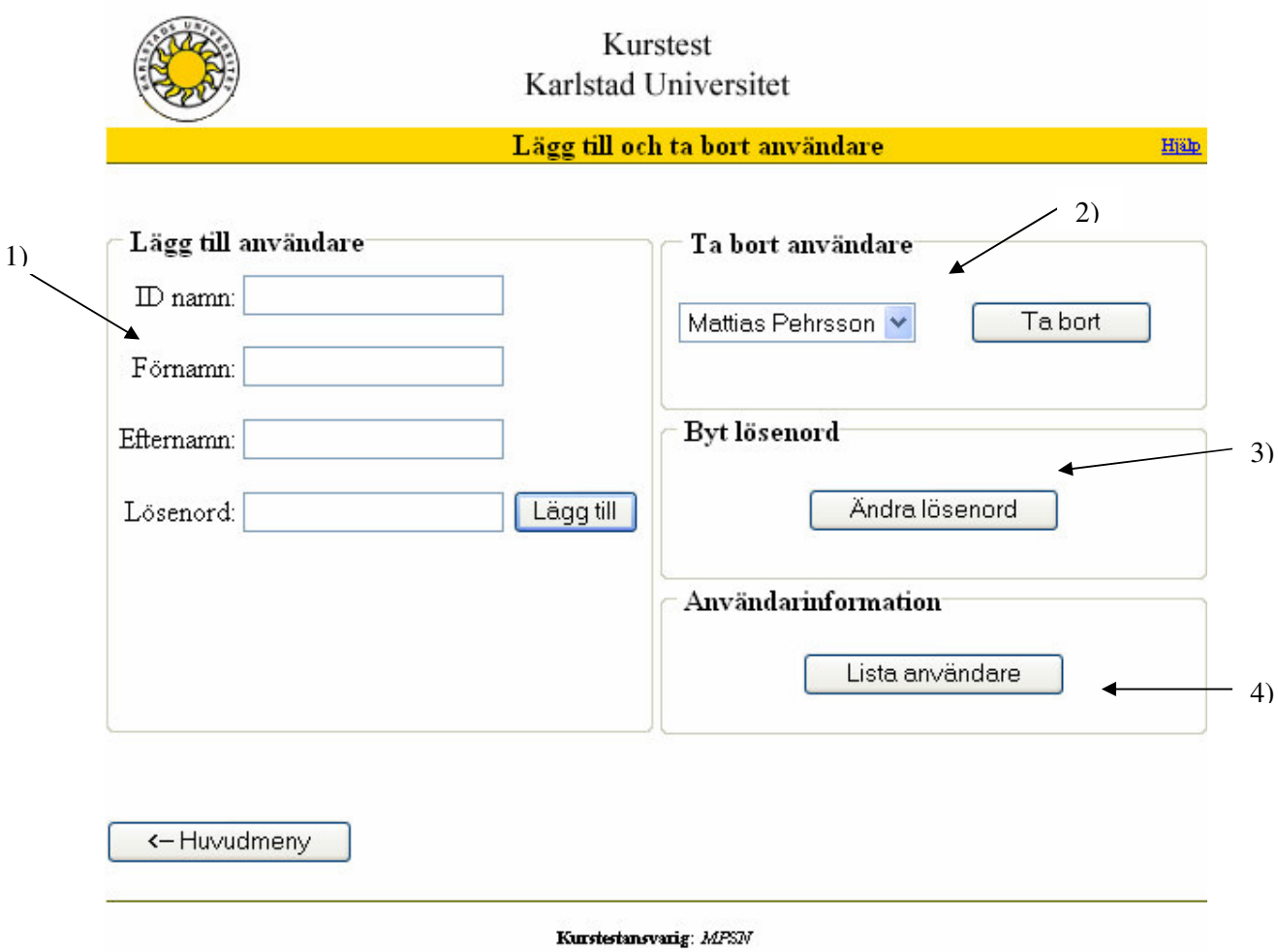
Den systemansvarige går in på inloggningssidan och loggar där in på systemet som ”root”. Det gör han/hon genom att skriva in sitt användarnamn (1) och lösenord (2) i inloggningsformuläret. När användaren gjort detta så kan han/hon logga in genom att trycka på ”Logga in”-knappen (3). Det finns även en hjälplänk (4) om problem uppstår. Från denna inloggningssida kan man även komma till studentens startsida (5).

B.3.2 Systemadministratörens startsida



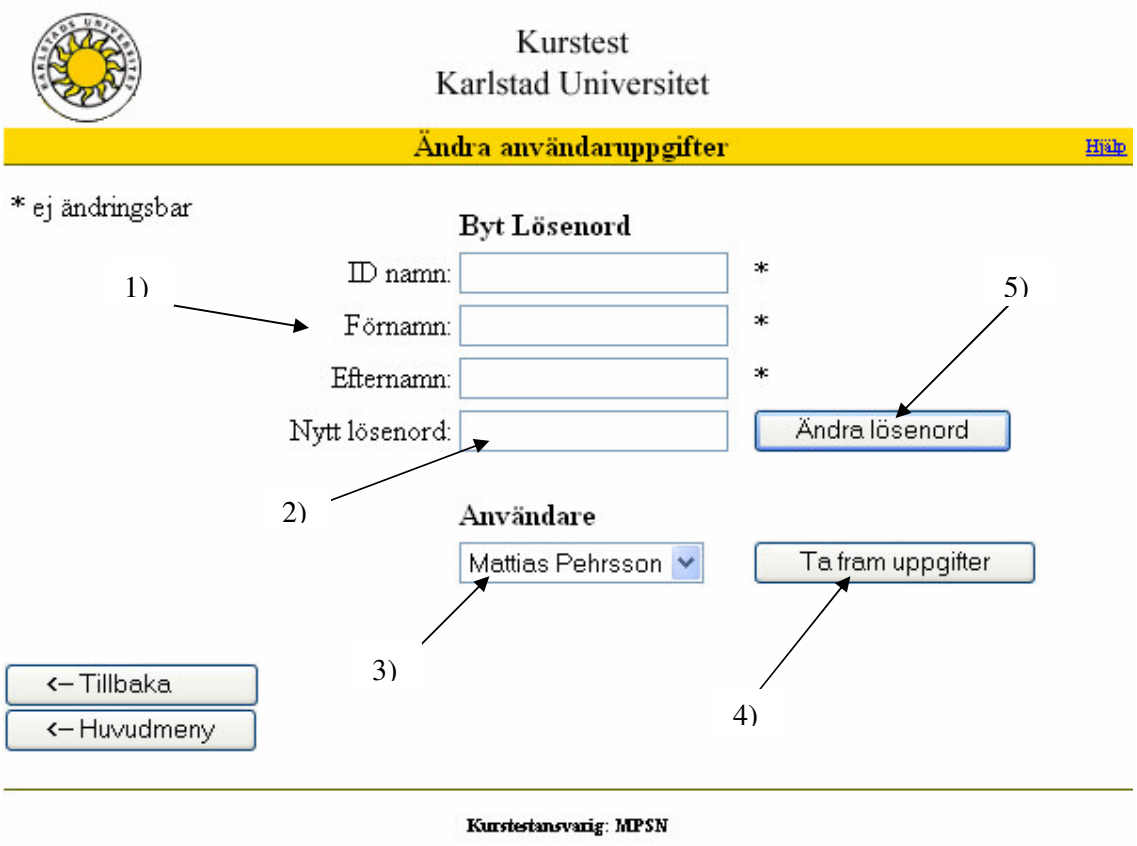
Systemadministratörens startmeny.

B.3.3 Redigera användare



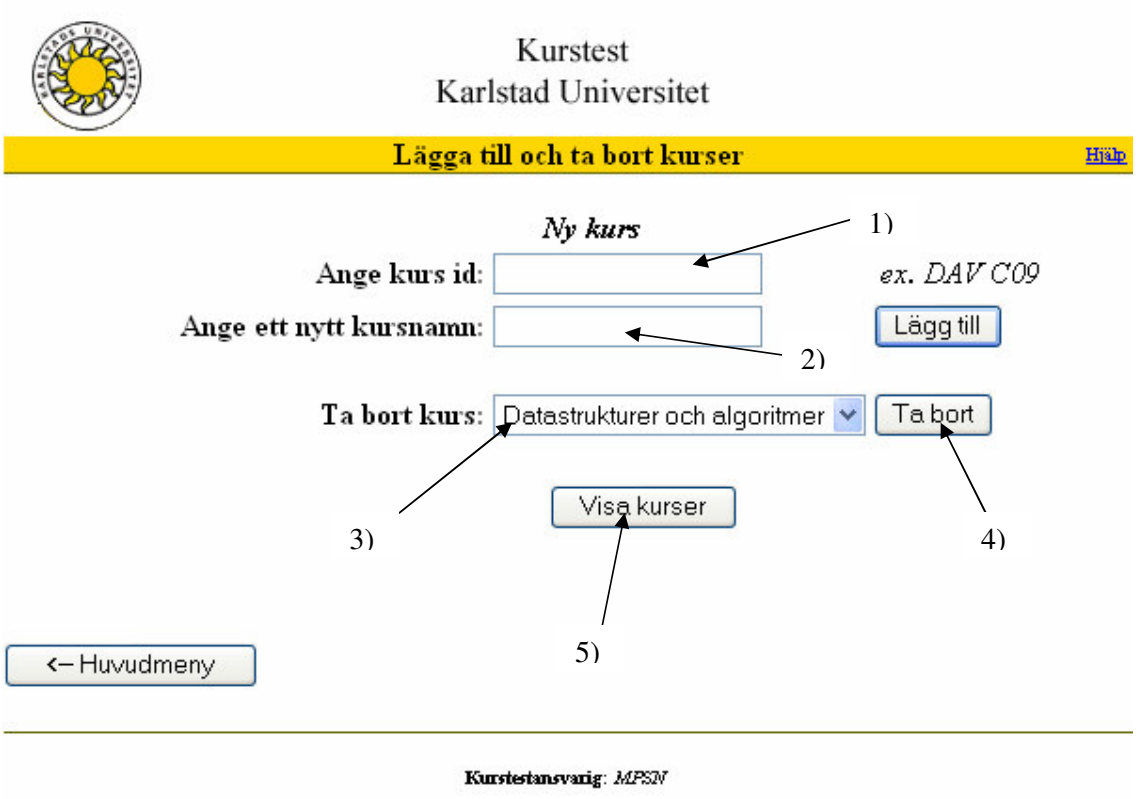
På denna sida kan systemadministratören lägga till (1) och ta bort (2) användare samt ändra lösenord (3) för en användare och lista alla användare (4).

B.3.4 Ändra lösenord



På ”ändra lösenord”-sidan ändrar systemadministratören en kurstestansvarigs lösenord. Detta görs genom att antingen skriva in användarens alla uppgifter (1), samt ett nytt lösenord (2), eller genom att välja användare (3) och trycka på knappen ”Ta fram uppgifter” (4). Då får man fram alla uppgifter på användaren (1). När man skrivit in ett nytt lösenord så trycker man på knappen ”Ändra lösenord” (5 för att ändringen av lösenord skall träda i kraft.

Administrera kurser



Kursadministratören kan lägga till och ta bort kurser. För att lägga till en kurs så anges först ett ”kurs id” (1), förslagsvis kurskoden på aktuell kurs, och namnet på kursen (2). För att ta bort en kurs så väljer man i rullningslisten kursen (3) och trycker på knappen ”Ta bort” (4). Här kan man även komma till en sida som listar alla kurser genom att klicka på knappen ”Visa kurser” (5).

C Systemmanual

Förord

Denna manual behandlar installation, handhavande och teknisk beskrivning av kunskapstestet ”Online kunskapstest”. Systemet är framställt av Mattias Pehrsson och Staffan Nilsson, som en del av en C-uppsats i Datavetenskap vid Karlstad University.

Introduktion

Kunskapstestet är ett test som är framställt för att publiceras på Internet. Produkten lämpar sig bäst som ett test för studenter och kursdeltagare för att vid kurser kontrollera att man har förstått det som sagts under föreläsningar och det man läst i kurslitteratur. Testet kan vara till hjälp för att förstå sig på innehållet i kursen. Testet skapas av den kursansvariga som skapar frågor enligt vad han/hon anser vara grundläggande i kursen.

Testet ska inte förknippas med en tentamen över Internet utan är endast ett test för varje individ att utföra efter eget behov.

Installation

Paket innehåll

Paketet inne håller en zippad version av ”Online Kunskapstest”, zip filen består av en katalog WebDocs som innehåller två mappar Bilder och PHP samt två filer index.htm och loggin.php. Mappen Bilder innehåller alla bilder som finns i testet, mappen PHP innehåller alla filer som systemet kräver. Filen index.htm är den sida som studenten kommer först till och loggin.php är den sida som den kursansvariga loggar in sig på. Det bifogas även användarmanual och systemmanual

Systemkrav

Kunskapstestet ska lagras på en server som stöder MySQL databashanterare och
PHP version 4.2.2 med ”session support enable”.

Installation på databas

Packa upp zippade versionen av ”Online Kunskapstest”. Skapa en mapp med namnet WebDocs på servern och kopiera sedan de uppackade filerna till mappen WebDocs. Detta kan göras med programmet WinSCP. När sedan filerna är kopierade till databasen kan man komma åt filerna index.htm och loggin.php genom den URL sträng som databasen finns lagrad på.

Filbeskrivning

I följande beskrivning finns en lista på alla filler som finns i mappen PHP och en beskrivning vad programmet utför i varje fil. Det finns ett förvillkor, pre och ett eftervillkor, post för varje fil. Koden i filerna innehåller kommentarer vid de avsnitt som kan vara otydliga eller som behövs belysas om vad som utförs.

Först i varje fil finns session_start() som ser till att den inloggade förblir inloggad och inte kastas ut ur systemet. Sedan följer include ”conn.php” som är till för att koppla upp sig mot rätt databas, det är denna fil som måste ändras om man byter till en annan databas eller byter användarinformation för att logga in sig på databasen. Efter det följer i de flesta filerna lite PHP-kod sedan kommer HTML-koden. I HTML-koden finns också PHP-kod inbakad mellan taggarna <? ?>.

antal_fragor.php

Beskrivning:	Filen tar fram antal frågor som studenten önskar svara på, anropas från student1.php
Pre:	En hidden variabel innehållande rubrik på testet
Post:	Returnerar antal frågor som ett test ska innehålla till test.php

change_losen.php

Beskrivning:	Filen har hand om ändring av lösenord, anropas från rootuser.php
Pre:	None
Post:	Uppdaterar tabellen User.Password

conn.php

Beskrivning:	Filen har hand om uppkopplandet mot databasen, anropas från alla filer
Pre:	None
Post:	Uppkopplad mot rätt databas på servern enterprise.cse.kau.se

listofuser.php

Beskrivning:	Filen visar någon av tabellerna, användare, kurser eller test beroende på vilken fil som anropar, anropas från någon av följande rootmeny.php ‖ rootcourse.php ‖ rubrik.php
Pre:	None
Post:	Listar hela innehållet i tabellen som stämmer in med förvillkoret.

login_new.php

Beskrivning:	Denna fil används för att kontrollera användarnamn och lösenord men även när man loggar ut från databasen. Anropas från loggin.php och rootmeny.php
Pre:	None
Post:	Vid true blir man inloggad som antingen administratör eller kursansvarig. Vid false kommer man tillbaka till inloggnings sidan och ett felmeddelande skrivs ut. Om man loggar ut från databasen kommer man till inloggnings sidan.

preview.php

Beskrivning:	Förhandsgranskar testet för att man ska få en överblick av det färdiga testet under tiden som man skapar det. Anropas från skapa.php och redigera.php.
Pre:	Två hidden variabler innehållande rubrik på testet och från vilken sida som anropet kommer ifrån.
Post:	Visar samma gränssnitt som studenten ser när han/hon gör testet

ratta.php

Beskrivning:	Rättar testet som studenten har gjort. Anropas från test.php.
Pre:	Två hidden variabler innehållande rubrik på testet samt antal frågor. Samt formuläret med test svaren från studenten.
Post:	Formuläret rättas och lämplig utskrift sker med antal rätt samt kommentarer om man svarat fel.

red_rubrik.php

Beskrivning:	Filen listar alla test som finns inom en specifik kurs. Anropas från red_test.php

Pre:	None
Post:	Returnerar rätt test till redigera.php

red_test.php

Beskrivning:	Filen listar alla kurser som finns inom ett ämne. Anropas från usermeny.php.
Pre:	None
Post:	Returnerar rätt kurs till red_rubrik.php

redigera.php

Beskrivning:	Filen används när man vill redigera eller lägga till frågor till ett befintligt test. Anropas från red_rubrik.php
Pre:	None
Post:	Testet redigerat

rootcourse.php

Beskrivning:	Filen används för att lägga till en ny kurs eller ta bort en redan existerande kurs. Man kan även lista de befintliga kurserna. Anropas från rootmeny.php.
Pre:	None.
Post:	Ny kurs registrerad eller kurs borttagen

rootmeny.php

Beskrivning:	En meny bestående av knappar som visar administratörens alla val
Pre:	Inloggad som root
Post:	TRUE

rootuser.php

Beskrivning:	Filen används för att lägga till användare till systemet eller ta bort användare från systemet. Sidan innehåller även två knappar den ena knappen används för att byta lösenord och den andra för att lista alla användare. Anropas från rootmeny.php.
Pre:	None
Post:	Ny användare registrerad eller användare borttagen

rubrik.php

Beskrivning:	Filen används för att skapa en rubrik på ett test eller ta bort ett test man kan även lista alla test. Anropas från usermeny.php.
Pre:	None
Post:	Rubrik på ett test skapat eller en rubrik på ett test borttaget

skapa.php

Beskrivning:	Filen används för att skapa test, varje fråga visas på en enskild sida och man kan förhandsgranska testet. Anropas från rubrik.php.
Pre:	En hidden variabel innehållande rubrik
Post:	Ett test skapat

student.php

Beskrivning:	Filen används för att lista alla kurser som är registrerade, om det inte finns några kurser registrerade så skrivs ett meddelande ut. Anropas från index.htm
Pre:	None
Post:	Returnerar kursnamn

student1.php

Beskrivning:	Filen används för att litsa alla test som finns inom specifik kurs. Anropas från student.php.
Pre:	En hidden variabel innehållande kursnamn
Post:	Returnerar rubrik

tabeller.php

Beskrivning:	Filen används för att lista hela innehållet i tabellerna Course, Test, Question och Answer. Anropas från rootmeny.php
Pre:	None
Post:	Listar tabeller

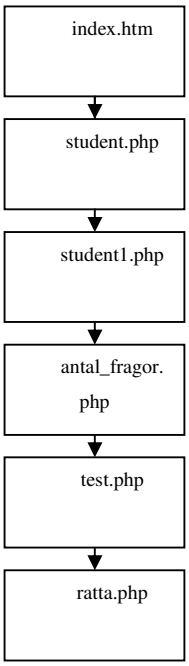
test.php

Beskrivning:	Filen används för att visa testet för studenten tar emot antal frågor som testet ska innehålla och slumpar fram frågorna. Anropas från antal_fragor.php
Pre:	Två hidden variabler innehållande rubrik och max antal frågor

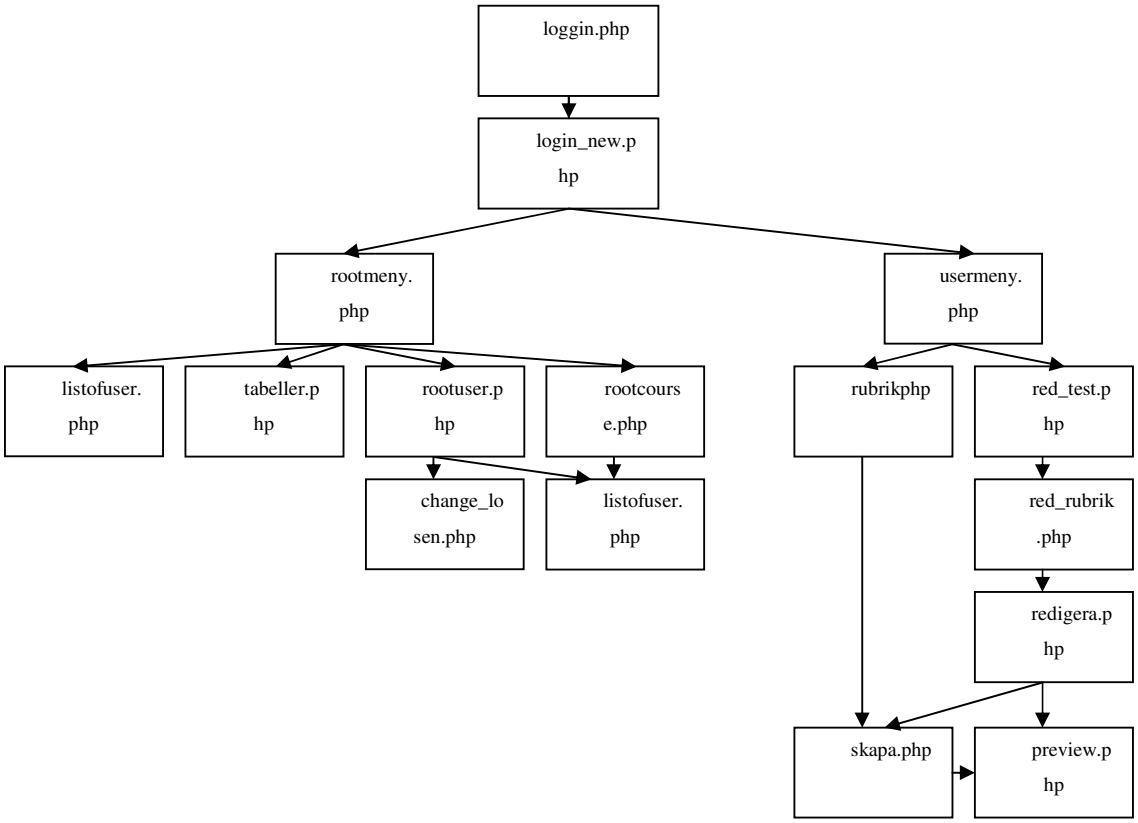
Post:	Returnerar ett formulär med frågenummer och svar

usermeny.php

Beskrivning:	En meny för den kursansvariga visas innehållande skapa test och redigera test samt logga ut
Pre:	Inloggad som kursansvarig
Post:	TRUE



Filbeskrivning student



Filbeskrivning kursansvarig/administratör

D Synpunkter

Synpunkter lämnade av Monique Pehrsson:

Gula raden: Kunskapstester
Raden under: Välkommen att testa dina kunskaper inom datavetenskap
Felstavning: ”Här kan du skapa att test” - borde väl vara ”ett test”
Åtgärdat

Borde vara en sida för Kursansvariga/en sida för studenter
Åtgärdat

Trycker på starta ett test;
Gula raden: Testa dina kunskaper
Texterna för att välja ämne och test är helt OK enligt mig.
Jag genomför ett test där jag enbart har ett alternativ. Bra!
Efter att jag valt ett test och genomfört det, så vill jag ha reda på resultatet men även se mina svar. Jag skulle då vilja se radioknapparna med mina svar samt i marginalen till höger ha en uppräkning av de rätta svaren. Då kan jag jämföra mina svar med facit. Jag vill även fortfarande se frågorna så att jag kan räkna på var jag gjorde fel.
Åtgärdat; när man rättar sitt test står det rätt med grön text eller fel med röd text beroende på om man svarat rätt på frågan, men inget facit

Tillbaka-knappen i resultatrutan tar mig tillbaka till ämnesfönstret. Hade nog förväntat mig att komma till testfönstret igen. Eller är jag enbart där för att göra ett test? Vad kommer den typiske användaren att göra, ett eller flera tester?
Åtgärdat; när man trycker på tillbaka-knappen kommer man tillbaka till testfönstret

Loggar in som root;

Kommer in i menyn som root

Här är enbart ett tycke/smakgrej. För mig behöver ni inte upprepa i knapparna vad som står i text. Står det ”Lista användare” så skriv OK på knappen. I och med att knapparna är så nära textraderna så vet man vilken knapp som hör ihop med vilken text. Det blir så mycket annars som jag måste läsa.

Här är lite synpunkter på knappar och text:

Text:	Knapp:	Kommentar:
Lista användare	Lista användare	Knappen kan heta OK.
Redigera användare	Användare	Knappen kan heta OK.
Skapa ny kurs	Ny kurs	Text: Administrera kurser, knapp OK
Kolla tabeller	tabeller	Text: Lista resultattabeller, knapp OK

Eller som ett alternativ, ta bort all text, och använd enbart knappar med texter som ingång.

Åtgärdat; en beskrivande text bredvid knappen och på knappen finns en pil

Super User – alternativ text ”root-administratör” , ”administratör”

Åtgärdat