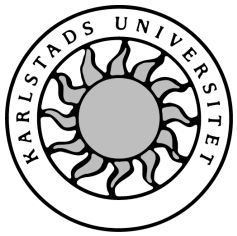


Datavetenskap

Mattias Wikström, Johan Wimnéus

Projekt Klaraborg
- Ett webbaserat personaladministreringssystem

Examensarbete

2004:09

Projekt Klaraborg
- Ett webbaserat personaladministreringssystem

Mattias Wikström, Johan Wimnéus

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.

Mattias Wikström

Johan Wimnéus

Godkänd, 2004-05-06

Handledare: Johan Eklund

Examinator: Stefan Lindskog

Sammanfattning

Detta projekt har utförts på uppdrag av Klaraborgs herrgård, vilket är kårhuset för Karlstad Universitets studentkår. Klaraborg har till uppgift att ge studenter en samlingsplats utanför studietid.

Idag finns det ett system som är baserat på papper och penna. Problemet med detta är att personalen har möjlighet att ändra sina arbetstider utan ledningens medgivande.

Syftet med projektet är att skapa ett databassystem som administrerar arbetstider för personal på Klaraborg och att ge ledningen en informationskanal där de anställda kan ta del av utgiven information.

Det nya systemet skall även kunna tillhandahålla statistik på hur ofta personal arbetar. Dessutom skall systemet ge användare möjlighet att själva se hur deras schema ser ut, och i viss mån påverka detta.

Resultatet av projektet blev ett databasbaserat system där varje användare har ett personligt användarnamn och lösenord. Gränssnittet används via webläsare och ger därmed användare möjlighet att nå systemet via Internet.

Project Klaraborg

- A web based staff administration system

abstract

This project has been done by the request of Klaraborgs Herrgård, which is the Karlstad University student union's student house. Klaraborg's task is to provide a casual meeting place for students when they aren't studying.

The intent of this project is to create a system for administrating the work time of the staff at Klaraborg. It should also act as an information channel between the management and staff of Klaraborg.

The current system for administration is based on "pencil and paper". The problem with this system is that staff can change their work hours without the management's approval.

The system should be able to provide statistics of how much the staff has been working. Additionally, the system should provide a method for the users to view their own schedule and in some ways affect it.

The result of the project was a database system where every user has its own username and password. The interface is through a web browser and thus gives the users access to the system through the Internet.

Innehåll

Figurer

1 Introduktion

1.1 Bakgrund till uppgiften

Examensarbetet skrivs på uppdrag av Klaraborg vilket är Karlstad studentkårs kårhus. Klaraborg fungerar som en krog/nattklubb för studenter med gäster. Klaraborgs personal består av studenter varav majoriteten arbetar ideellt men delar av personalen är arvoderad. Detta har till direkt följd att personalstyrkan är många till antalet. För att ha ordning på vilka som arbetar under en kväll används en bokningslista. Sista söndagen i varje månad anordnas ett möte där personalen kan skriva upp sig för de dagar de vill arbeta. Denna information sparas sedan undan i aktuell bokningslista.

Här uppstår ett problem för Klaraborgs ledning, eftersom bokningslistan är en bunt papper är det lätt att den försvinner på något kontor eller annat rum i byggnaden. Dessutom är det möjligt för de arbetande att ändra i sitt schema utan ledningens kännedom. Detta skapar onödigt merarbete för ledningen. Mängden personal genererar ytterliggare ett problem, det är svårt att få fram aktuell information till alla på ett effektivt sätt. Idag sprids informationen muntligt, vilket kan leda till att meddelandet kan bli förvanskat på vägen genom att tvetydigheter skapas när flera mellanhänder tolkar informationen.

I *Appendix B* finns förklaringar på de termer som används i denna rapport.

1.2 Definiera uppgiften

Uppgiften är att skapa ett system bestående av en databas med användargränssnitt som ger Klaraborgs ledning kontroll över den frivilliga personalens arbetstider. Dessutom skall bara ledningen kunna administrera bokningslistan. Det skall finnas möjligheter för personalen att titta på sina arbetstider. Systemet skall även fungera som informationskanal som ledningen kan använda för att ge information till personalen.
Följande krav måste uppfyllas:

- Implementationen skall vara skapad för att användas online, dvs för lätt åtkomst via Internet
- Varje användare skall ha ett eget användarnamn och lösenord
- Bokningslistorna måste formaliseras och implementeras så att nya listor lätt kan skapas
- Ett meddelandeförmedlande system och ett forum i form av en elektronisk anslagstavla skall utformas. Detta för att utgöra en informationskanal där ledningen kan ge information till alla samtidigt.
- Användarna ska kunna överblicka sina arbetstider i form av ett elektroniskt schema

För fullständig kravspecifikation se *Appendix A*.

2 Bakgrund

2.1 Klaraborgs funktion

Klaraborg som är Karlstad studentkårs kårhus har många uppgifter. De driver en nattklubb i huset för studenter. Detta är deras primära uppgift, men de tillhandahåller även lokaler för studenter, studentföreningar och företag/kommuner som vill anordna middagsbjudningar eller mindre konferenser. Studentkåren använder även Klaraborg för catering till evenemang i studentkårens regi eller på uppdrag av universitetet.

2.2 Klaraborgs organisation

2.2.1 Personal

Studentkåren har tre studenter heltidsarvoderade som är ansvariga för Klaraborg. De positioner som är arvoderade är VD, vice VD samt kock. VDn ansvarar för driften av huset och har det övergripande ansvaret för avlönad personal. Vice VDn har, förutom delar av husets drift, till huvuduppgift att tillhandahålla personal till nattklubben. Kocken är Klaraborgs inköpschef samt är den som lagar mat till bokade middagar och till cateringbeställningar.

Klaraborg har även timanställd personal, kvällsansvariga, vakter samt städpersonal.

- Under kvällstid finns kvällsansvariga som har ansvaret för nattklubben, de fungerar som arbetsledare och kallas för ”rödskjortor” eller ”ettor”. Det måste finnas en ”rödskjorta” i tjänst när Klaraborg har öppet. ”Rödskjortorna” är de som är ytterst ansvariga för vad som sker i huset under den tid det är öppet för gäster.

- Klaraborgs vakter har till uppgift att upprätthålla en bra stämning och avlägsna störande personer. Vakter är, av polisen, förordnade ordningsvakter som studerar vid karlstads universitet.

- Städpersonalen även kallad städpoolen, bestående av frivilliga studenter, städar huset.

Personalen som arbetar under kvällarna gör en grovstädning samt rengör arbetsstationerna så som bardiskar, kök och liknande efter stängning, medan städpoolen rengör öppna ytor.

- Klaraborg har två dansgolv och musiken på dessa sköts av DJs. På det övre dansgolvet spelar studenter och på det undre inhyrda DJs. Studenterna som spelar bokas av VDn och får betalt per kväll de spelar.

Den stora arbetskraften finns i poolen, dvs frivilliga studenter som arbetar ideellt, dessa personer kallas poolare och det är de som bemannar Klaraborg förutom ovan nämnd anställd personal. Poolen består av många personer så för att administrera poolens arbetstider anordnas bokningsmöten sista söndagen i varje månad. De arbetsuppgifter poolarna har är att servera mat och dryck, diska, samt bemanna garderoben. Istället för lön anordnas aktiviteter för poolarna en gång i månaden för dem som jobbat minst fyra gånger mellan de senaste aktiviteterna där de deltagit.[PC 1]

2.2.2 Aktiviteter på Klaraborg

De kvällar Klaraborg är öppet kan delas in i tre olika klassificeringar; pubkvällar, nattklubb eller abonnerat. Dessa klassificeringar används för att beskriva vilken form av aktivitet som kommer att bedrivas samt att den anger vilka öppettider huset har under den kvällen och vilken/hur mycket personal som krävs.
Pubkvällarna har öppettider mellan klockan nitton och nollett. Under en pubkväll är bara en av barerna och köket öppet. Gästerna skall under dessa kvällar kunna sitta i en lugn pubmiljö och prata eller spela sällskapsspel.
Nattklubben har öppet mellan klockan arton och nolltvå. De kvällar då Klaraborg har nattklubb är indelad i två delar, innan och efter tjugotvå. Innan tjugotvå fungerar huset som under en pubkväll medan efter klockan tjugotvå öppnar dansgolven och den övre baren. Under nattklubbskvällarna skall gästerna kunna sitta och prata samt besöka något av dansgolven.

När Klaraborg är abonnerat har huset öppet enl. önskemål (oftast mellan klockan tjugo och nolltvå). Kvällar då Klaraborg är abonnerat är det någon utomstående organisation som arrenderar huset för att nyttja det till egna aktiviteter. Under dessa kvällar ställer Klaraborg enbart upp med lokaler och personal efter abonnentens önskemål. Vilka aktiviteter som bedrivs varierar, det är upp till dem som abonnerar huset.

För den frivilliga personalen är de olika kvällarna indelade i arbetspass eller pass som de kallas. Under ett pass måste det finnas en ”rödskjorta” som har ansvaret. Vilka andra som jobbar beror på vilken typ av klassificering kvällen har. Pubpassen kräver inte så mycket personal, en ”rödskjorta” och sex till åtta poolare. Nattklubbskvällarna är indelade i två pass, tidigt och sent pass. Under det tidiga passet finns det förutom ”rödskjortan” sex till åtta poolare, medan det under sena passet finns tio till tolv poolare, tre till fyra vakter samt två DJs. Under de kvällar Klaraborg är abonnerat är det upp till abonnenten att tala om hur mycket personal de vill ha, dock minst en ”rödskjorta”. För poolarna räknas ett pass som ett arbetstillfälle.

2.2.3 Problemet

Det är under bokningsmötena poolen bokar upp sig på pass. Under bokningsmötena skriver vice VDn ner de som kan och vill jobba i bokningslistan. Denna lista sparas för senare kunna se vilka som ska jobba vilka dagar. Problemet med detta papper och penna system är att det är allt för lätt för personalen att göra ändringar utan att ledningen vet om att en ändring utförts. Dessutom har bokningslistan en tendens att försvinna och mycket tid ödslas på att leta reda på den. Men bokningslistan behöver samtidigt vara tillgänglig på olika ställen för att den skall vara till nytta. Detta utgör grunden till att Klaraborgs ledning vill skapa ett databasbaserat program.[PC 1]

2.3 Uppgift

Uppgiften är att skapa en databas med användargränssnitt, i rapporten kallat programmet. Programmet skall ersätta bokningslistan som nu finns i pappersformat. Användare ska kunna logga in och kontrollera vilka pass han/hon skall jobba samt även kunna anmäla och avanmäla intresse för att jobba. För att uppfylla kravet på att enbart ledningen skall kunna ändra i databasen måste programmet bestå av två delar, en del som alla användare kan använda samt en del som bara administratörer, dvs ledningen för Klaraborg, har tillgång till. Ur denna databas skall även statistik kunna hämtas över hur mycket personalen har arbetat.

Programmet skall även fungera som en informationskanal för dem som arbetar på Klaraborg. Denna informationskanal ska bestå av ett forum där användarna har tillgång till dels ett gemensamt och dels mer privata forum beroende på vilken personalgrupp personen tillhör. Det gemensamma Forumet skall vara en anslagstavla där all personal ska kunna delge sina tankar. Men i de grupplåsta forumen ska bara personal med samma ansvarsområden ha tillgång till att läsa och skriva. Förutom forumet skall användarna kunna skicka privata meddelanden till varandra. Detta meddelandesystem ska fungera som internpost dvs som att skicka e-post men bara till registrerade användare[PC 1].

2.4 Förutsättningar

För att skapa ett program som innesluter uppgiften måste det hantera bokningar och begränsa rättigheterna att ändra i listorna. Möjligheterna att göra ändringar i systemet begränsas till ledningen, som kommer att vara programmets administratörer. Behovet av att bokningslistan är flyttbar löses genom att göra databasen åtkomlig via Internet.

3 Genomförande

3.1 Analys och design

3.1.1 Databasen

Systemets databas implementeras med MySQL[URL 1]. MySQL är ett open-source databasprogram som är gratis. Alternativen som övervägts är dessutom Oracle[URL 2], Microsoft SQL server[URL 3] och Microsoft Access[URL 4] de har samma och i några fall även mer funktionalitet, men de faller på att de kostar pengar. Anledningen till att MySQL valdes som databas är att det fanns tillgängligt på studentkårens server vid tidpunkten för projektets start [PC 2] och sålunda behövdes inga extra program installeras för detta projekt.

3.1.2 Gränssnittet

Gränssnittet skall ge användarna av systemet relevanta och lätt förståeliga svar på de frågor som ställs till databasen. Den stora ledstjärnan vid skapandet av gränssnittet var användarvänlighet och intuitivitet eftersom användarna inte kan förväntas ta sig tiden att läsa i en manual [PC 1]. Då databasen skall vara tillgänglig via Internet jämfördes två lämpliga scriptspråk ASP(Active Server Pages)[URL 6] och PHP(Hypertext preprocessor)[URL 5]. Anledningen till att gränssnittet programmeras med PHP och HTML[URL 7]är att PHP redan finns installerat på studenkårens server[PC 2]. Ytterligare en bidragande anledning är att medlemmarna i projektgruppen har tidigare erfarenheter av PHP.

3.1.3 Meddelandesystemet

Meddelandesystemet skall ge användarna möjlighet att skicka privata meddelanden sinsemellan. Varje meddelande skall ha en avsändare, en mottagare, ett ärende och en meddelandetext. Gränssnittet skall vid inloggning visa om en användare har olästa meddelanden. Meddelandesystemet har en viktig roll i programmet, det sköter hanteringen av bokningsförfrågningar från användarna och bekräftelserna/avvisningarna från administratörerna.

Meddelandesystemet är uppdelat i två delar, ett system som tar emot meddelanden från användare till användare och ett system som tar emot bokningsförfrågningar från användarna till administratörerna. Uppdelningen är gjord för att lätt kunna dela på meddelanden och bokningsförfrågningar, dessutom skall en bokningsförfrågan vara åtkomlig för alla administratörer. Om en bokningsförfrågan skickas till en specifik administratör, som ej är tillgänglig, kan bokningsförfrågan hinna bli inaktuell innan den blir upptäckt. I den delen av meddelandesystemet där användare kan skicka internmeddelanden måste meddelandet vara adresserat från en användare till en annan användare vilket medför att meddelanden inte kan skickas till flera mottagare.

3.1.4 Forumet

Forum är en elektronisk anslagstavla där användare ska kunna läsa meddelanden av allmännintressant karaktär. Det skall finnas ett allmänt forum för alla grupper av användare. Dessutom skall ett informationsforum skapas där bara administratörer kan sätta upp meddelanden, men alla kan läsa meddelanden i forumet. Varje användare tillhör en forumsgrupp beroende på vilken personalgrupp han/hon tillhör.
De forumsgrupper som ska finnas är:

- Ettor, som skall vara en informationskanal för de kvällsansvariga
- Poolare, för de som arbetar i poolen
- Vakter, för Klaraborgs vakter
- Städ, för städpoolen på Klaraborg

Dessa grupper har sina egna forum där bara medlemmar ur den egna forumsgruppen och administratörer har åtkomst.

3.1.5 Schema

Schemat skall visa de arbetstillfällen som en användare har. Information om kommande arbetstillfällen skall presenteras och användaren ges möjlighet att anmäla intresse för att jobba under dessa tillfällen. Dessutom skall information om vilka andra som jobbar under en kväll presenteras.

Om en användare vill jobba så skall denne kunna anmäla intresse, vilket görs med hjälp av användargränssnittet, se 3.1.2 på sid 7. En administratör får då meddelande om detta och kan boka in användaren i schemat. Användaren får ett meddelande från administratören om att han/hon har blivit inbokad att arbeta.

3.1.6 Administratör

Administratören skall kunna lägga till och modifiera information om användare och schema. Dessutom skall administratören ha full tillgång till forum och schema. Administratören skall även kunna få ut statistik på hur många arbetstillfällen en viss användare har gjort under en viss tidsperiod.

3.2 Implementation

3.2.1 Inledning

Projektet startade med att en tabell lagra användarinformation(användarnamn, lösenord, namn, grupptillhörighet, kontaktinforation och kommentarer) skapades i databasen. Till detta knöts sedan en PHPsida som verifierar användarnamn och lösenord och ger tillgång till systemet. Lösenordet lagras inte som klartext i databasen. Lösenordet kodas med en MD5(Message-Digest Algoritm)[URL 8] hashning. Denna hashning gör det närmast omöjligt att återskapa lösenordet från den i databasen lagrade textsträngen. T. ex. lösenordet *'Kalle'* lagras som *'b5f51c5c18456ba2e5505e26a1d2ff70'*.

För att verifiera identiteten hos användaren, utförs ovanstående algoritm för det an-

givna lösenordet. Resultatet jämförs sedan med det lagrade lösenordet. Om de överensstämmer så godkänns användarens identitet. T. ex Användaren Kalle matar in det korrekta lösenordet *'Sunset'*. Detta kodas till *'88eb60614bb67782bd8c18afb4438329'* och jämförs med lösenordet i databasen. De överensstämmer och Kalle godkänns som användare. Senare kommer Pelle och försöker logga in som Kalle. Pelle provar lösenordet *'sunshine'*. Detta genererar *'0571749e2ac330a7455809c6b0e7af90'* som inte överensstämmer med det korrekta lösenordet. Pelle nekas tillträde.

3.2.2 Meny

En dynamisk menysida skapades, den är dynamisk på så vis att den ser olika ut beroende på om den inloggade är administratör eller inte. Här presenteras länkar vidare till andra sidor. Menyn utökas i takt med att projektet fortgår. Alternativet till att ha denna dynamiska meny hade varit att skapa två separata menysystem, en för vanliga användare och en för administratörer. Anledningen till att den mer dynamiska varianten valdes var att det är mycket gemensam kod för de två menyerna. Administratörsmenyn är egentligen en utökning av en vanlig användares meny.

3.2.3 Databas

Databasen byggdes sedan ut med tabeller för att hålla bokningsförfrågningar, meddelanden samt passinformation. Administratörens sida för att skapa nya pass implementerades se *figur C.10* på sid 28. Nya pass skapas genom att en ”skapa pass sida” genererar ett html formulär. Formuläret fylls i och skickas tillbaka till ”skapa pass sidan” som skriver in informationen i databasen och presenterar ett nytt tomt formulär för vidare inmatning.

3.2.4 Gränssnittet

Systemet byggdes sedan ut för att hantera förfrågningar. Förfrågningar finns av två typer, bokningsförfrågningar och avbokningsförfrågningar se *figur C.12* på sid. 30. Dessa är imple-

menterade så att en användare får upp en sida där dels samtliga pass han/hon är registrerad på och dels alla andra pass presenteras se *figur C.5* på sid 23. På denna sida kan användaren sedan klicka på länkar för att skicka förfrågningar till en administratör. Dessa förfrågningar hamnar i en tabell i databasen. Administratören får upp information om hur många obehandlade förfrågningar som finns när denne loggar in. För att godkänna/neka förfrågningar krävs det enbart klick på länkar. Då kommer ett meddelande med information om beslutet att skapas och skickas till berörd användare.

En enkel sida för att få ut statistik ur systemet lades till se *figur C.13 och C.14* på sid 31 och 32. Med denna kan administratörer se hur många gånger en användare har jobbat under en given tidsperiod.

Slutligen finputsades layouter och sidor för att ge bättre funktionalitet och intryck.

Skärmdumpar och källkod finns i *Appendix C* respektive *Appendix D*.

3.3 Avgränsningar

Forumet som var planerat att finnas i programmet är inte implementerat. Grundläggande databasdesign av forumet har utförts men inga gränssnitt är klara. Detta på grund av det faktum att Klaraborg hade stora problem vid start för projektarbeteskursen VT-03. Resultatet av detta blev en försenad projektstart då Klaraborgs förtroendevalda ej hade tid för möten med projektgruppen.

Bokningarna som kan göras via programmet är enbart knutna till poolare. Motsvarande tidsbokningar för övrig personal behandlas inte inom detta arbete. Men kan utan större problem läggas till senare.

4 Diskussion

4.1 Analys av resultatet

Programmet ger administratörerna, vilket är Klaraborgs ledning, ensamrätt att ändra i poolarnas arbetstider. Bokningsmötena måste fortfarande hållas och detta ger vice VD:n lite mer att göra när denne sedan måste föra över mötesresultatet till programmet. Dock får vice VD:n en tydligare överblick när det fattas personal.

Poolarna kan nu använda programmet för att anmäla sitt intresse att arbeta via Internet och behöver inte direkt ta kontakt med personal på Klaraborg för att anmäla sitt intresse. Då ledningen inte alltid kan kontaktas via telefon.

Användarvänligheten hos programmet återstår att bli testad. Men stor vikt lades vid utformandet av de grafiska användargränssnitten på att få dem så intuitiva som möjligt.

4.2 Allmän diskussion

Detta system kommer att vara ett välbehövligt tillskott till Klaraborgs organisation. Visserligen fungerade det även innan, men nu kommer det förhoppningsvis vara mer smidigt.

Målet med detta system var att förenkla och förbättra för ledningen på Klaraborg så de får struktur på personalens arbetstider. Dessutom finns nu möjlighet för poolare att själva se hur läget ser ut och när det behövs personal. Detta avlastar ledningen ytterligare.

Eftersom ledningen byts ut varje år har vi försökt göra programmet så intuitivt och lättförståeligt som möjligt. Därmed skall ingen nämnvärd utbildning av nya administratörer behövas.

I början borde både papperssystemet och databassystemet köras parallellt tills att man med säkerhet kan fastslå att det fungerar tillfredställande. Eller att programmet har en testperiod där en liten grupp testar systemet, för att efter en utvärdering använda systemet skarpt.

Dessutom kanske ny funktionalitet önskas. I så fall känner sig inte författarna främmande för att fortsätta utvecklingen utanför ramen för detta arbete.

Det största problemet som programmet kommer få är att tas i bruk. Papperssystemet har varit i bruk länge och sitter i ryggmärgen hos många. Men bara man kommer igång skall det nog inte vara något längre problem.

4.3 Ytterligare arbete

Framtida arbete med systemet:

- Skriva färdigt forumet.
- En kontrollmekanism som ger administratörerna möjlighet att hindra användare med upprepat störande/stötande språk att använda forumet och att skicka meddelanden.
- Lägga in fler grupper än poolare, så programmet även hanterar schema för ettor, vakter, DJs och städpool.
- Skriva mer avancerade funktioner för hämtande av statistik för antalet arbetade pass, genomsnittlig mängd pass per poolare m. fl.

4.3.1 Problemet

I slutet av projektets gång så togs ett beslut av FUM (Studentfullmäktige) att Klaraborgs Herrgård skall läggas ner med i princip omedelbar verkan. Detta har som följd att systemet troligen aldrig kommer att användas skarpt.

4.4 Kritik av eget arbete

Vår arbete har måhända inte varit så omfattande och genomtänkt som vi önskat oss. Koden kunde ha givits mer struktur och bättre felhantering. Vi har kod i flera filer som

har likartad funktionalitet. Denna kod borde ha sammanställts i gemensamma funktioner. Detta är dock erfarenheter som vi tar med oss in i framtiden.

Arbetet har på många sätt varit mycket lärorikt och en viktig erfarenhet för liknande arbeten i framtiden. Den viktigaste erfarenheten är nog tidsplaneringen, där vi inte hade någon tidigare erfarenhet. Nu anser vi att vi har större möjlighet att på ett korrekt sätt uppskatta tidsåtgång för olika moment i ett projekt. Vi har i vårt arbete varit mycket optimistiska vad gäller tidsåtgång. Detta har fått till följd att delar av projektet har skurits bort.

Referenser

[PC 1] Daniel Sunnefors, *VD Klaraborg Herrgård VT2003*, +46(0)54-186311

[PC 2] Calle Merkéll, *Projektansvarig Karlstad Studentkår VT2003*, +46(0)54-7001485

[URL 1] www.mysql.com, *The World's Most Popular Open Source Database*, 2003-04-26

[URL 2] http://www.oracle.com/ip/deploy/database/oracle9i/, *oracle9i database*, 2003-05-13

[URL 3] http://www.microsoft.com/sql/default.asp, *Microsoft SQL server*, 2003-05-13

[URL 4] http://www.microsoft.com/office/access/default.asp, *Microsoft Access*, 2003-05-13

[URL 5] www.php.net, *Hypertext preprocessor*, 2003-04-26

[URL 6] http://www.asp.net, *ASP.NET*, 2003-05-13

[URL 7] www.w3.org, *World Wide Web Consortium*, 2003-04-26

[URL 8] http://www.cis.ohio-state.edu/cgi-bin/rfc/rfc1321.html, *RFC 1321 Message-Digest Algorithm MD5*, 2003-05-08

A Kravspecifikation

A.1 Inledning

Efter samtal med ledningen på Klaraborg där önskemål på systemet framförts. Dessa önskemål på produkten har summerats till följande punkter.

A.2 Krav på programmet

- Programmet skall vara
  - lätt att använda
  - lätt att administrera
  - Intuitivt för användarna
  - Tillgängligt via Internet
- Programmet ska hantera
  - personliga inloggningar
  - visa pass veckovis och i detalj, vilka som redan är bokade att jobba det passet
  - skapa nya pass
  - intresseanmälan för att jobba
  - avbokningsförfrågan från poolare
  - administratören bokar poolare till ett pass
  - administratören avbokar poolare från ett pass
  - användare ska kunna skicka personliga meddelanden
  - forum beroende av anställning(Vakt, DJ, Poolare, Etta och Städpersonal)
  - möjlighet att utnämna forumsansvariga

– allmänt forum för alla anställda

– ta fram statistik på hur många arbetspass en person arbetat

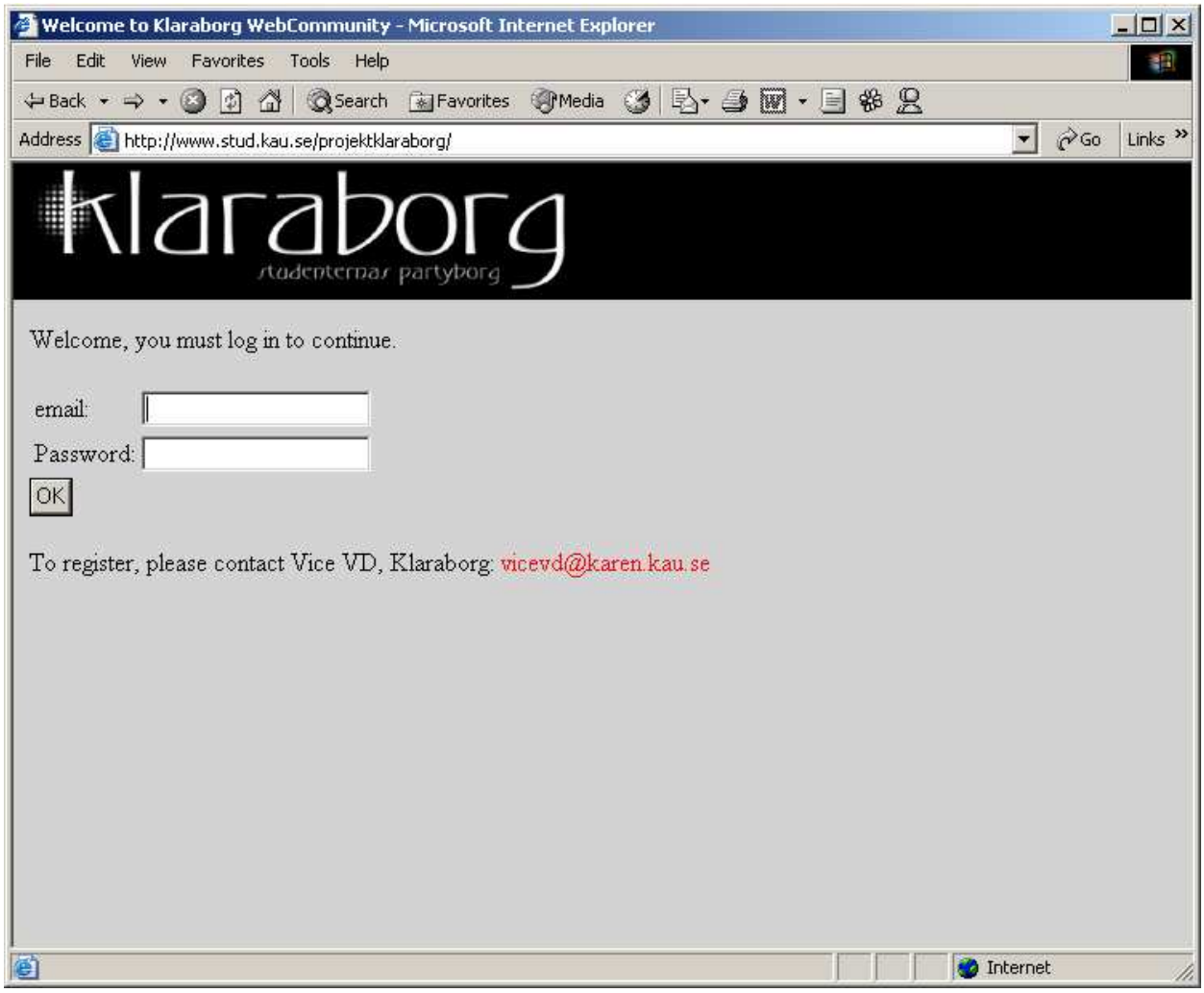
B Ordlista

Ordlista:

- Bokningslista - Den lista där information om vilka som jobbar finns lagrad.
- Bokningsmöte - Det månatliga möte där bokning av pass sker.
- Etta - Den Kvällsansvariga på Klaraborgs herrgård.
- Klaraborg - Karlstad studentkårs kårhus.
- Pass - Ett arbetstillfälle för de som jobbar på Klaraborg.
- Poolare - Studenter som arbetar ideellt under kvällarna.
- Rödskjorta - se Etta.
- Städpoolen - Städar huset efter nattklubbskväll.

C Skärmutskrifter

C.1 Alla användare

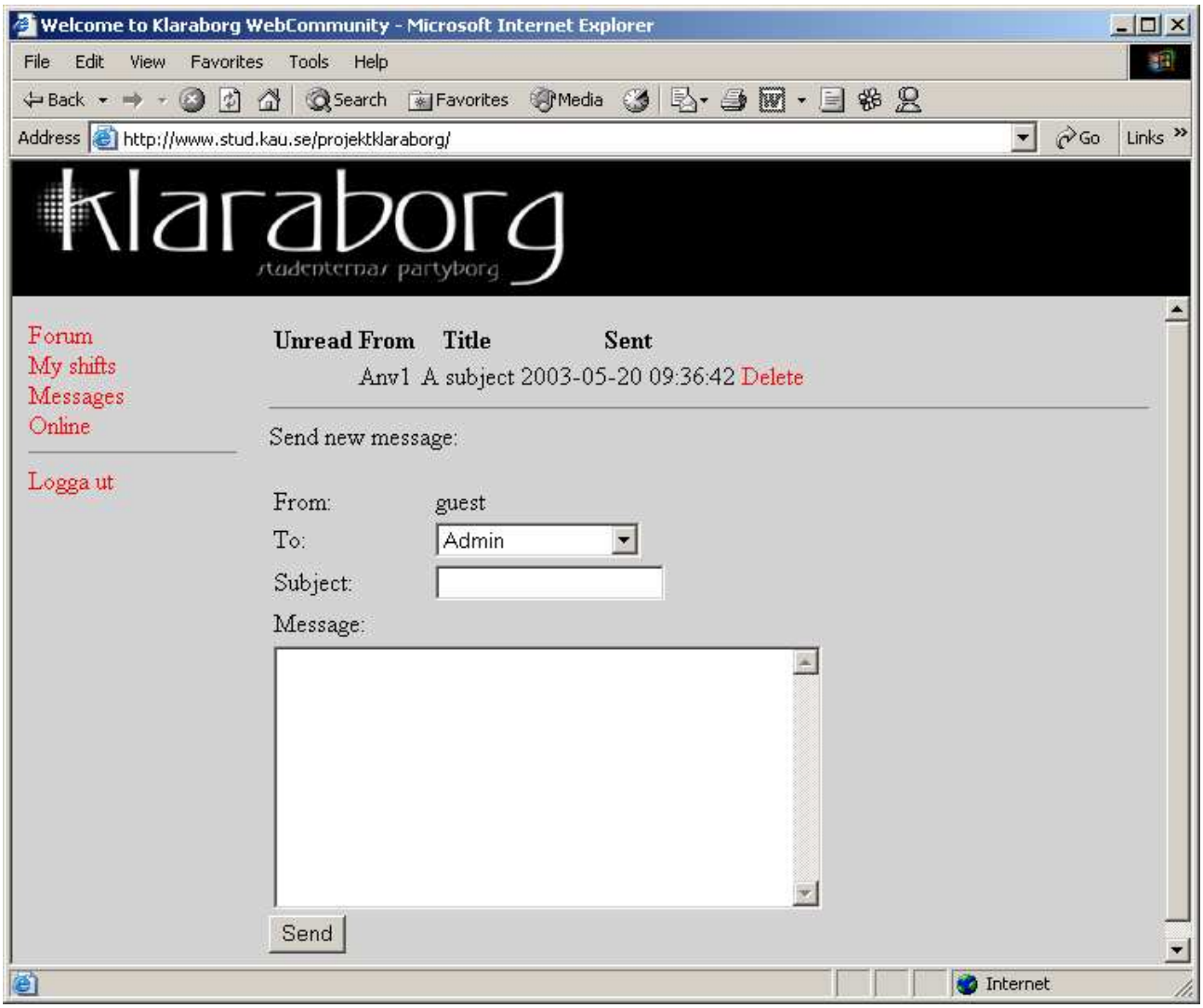


Figur C.1: Startsida

Här loggar användaren in med sin email och sitt personliga lösenord. Här finns även en email länk till vice VD.

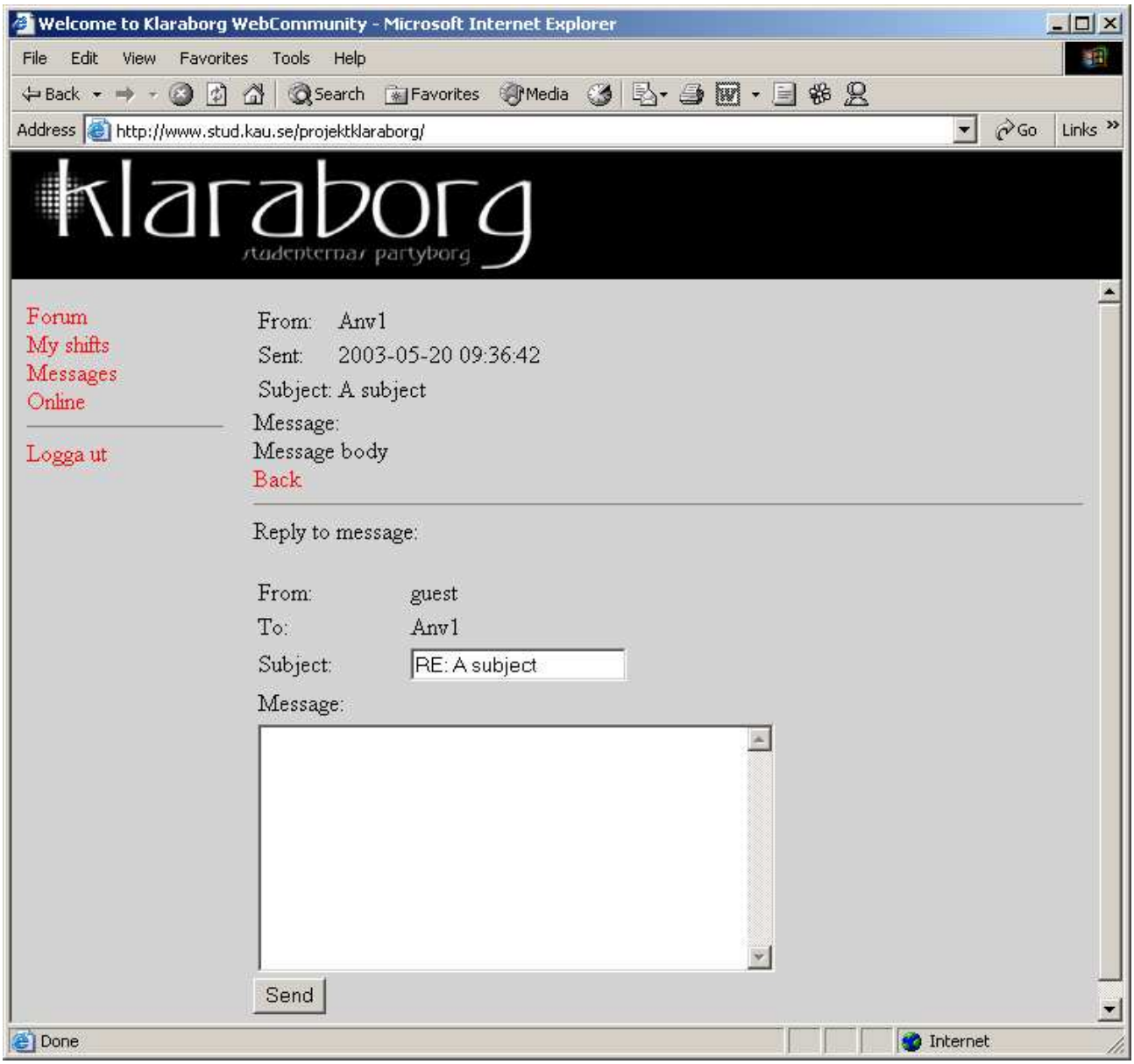
Figur C.2: Startsida för användare

Välkomstsida som visas när en användare loggat in. Presenterar antal olästa meddelanden.



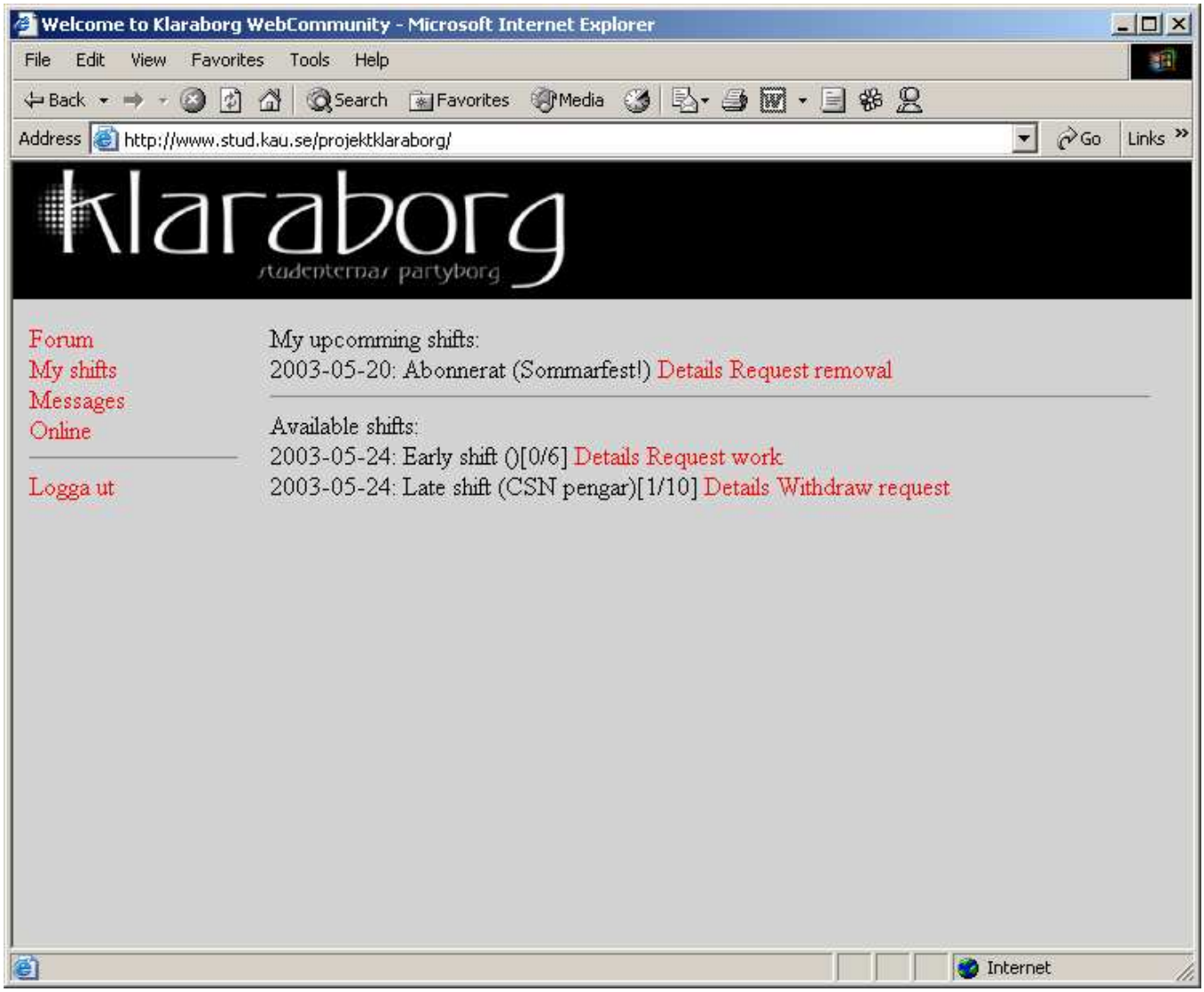
Figur C.3: Sida för att skicka meddelanden

Alla meddelanden listas och användaren har möjlighet att klicka på dessa för att läsa dem eller klicka på delete för att ta bort dem. Användaren fyller i formuläret för att skapa ett meddelande.



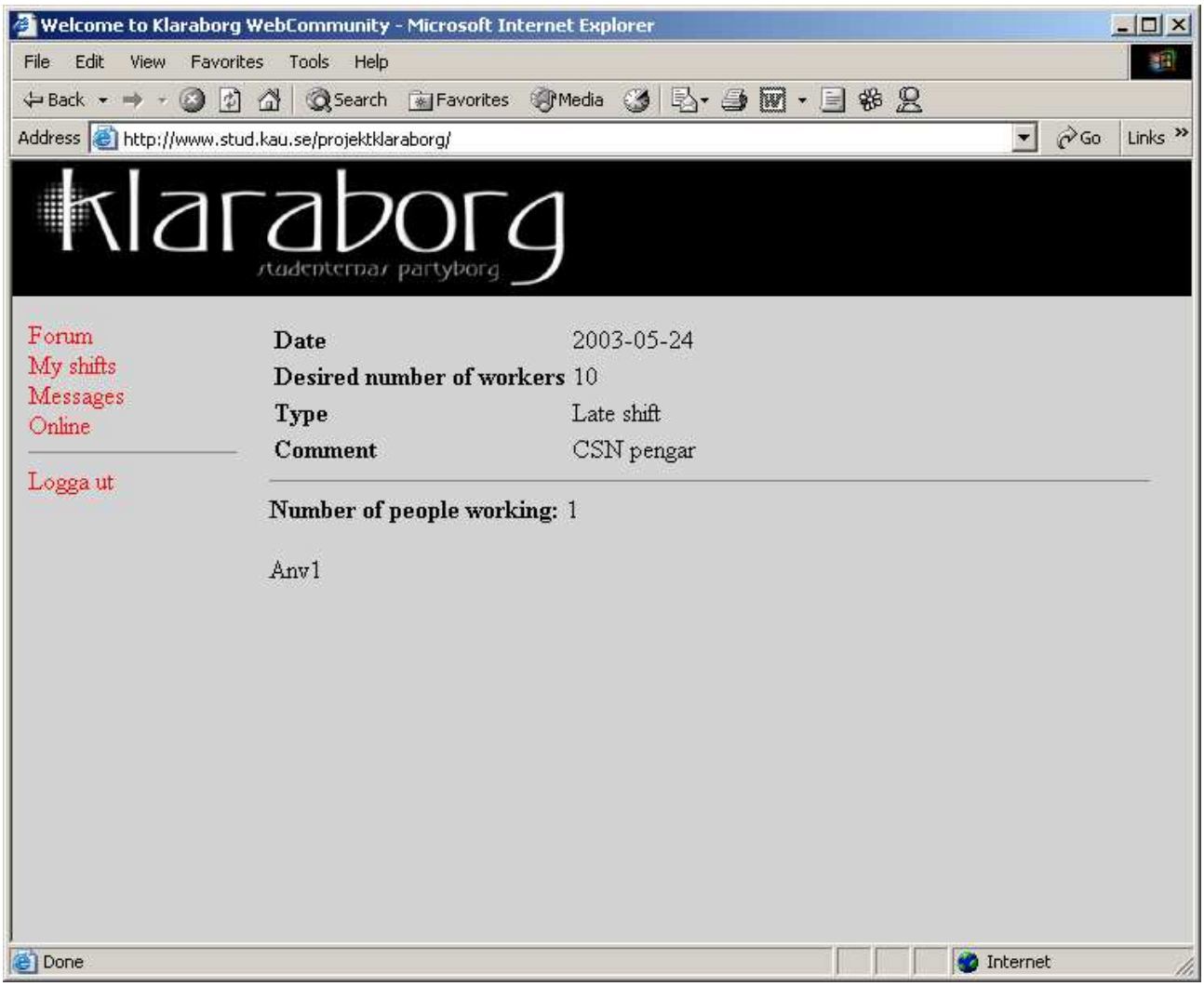
Figur C.4: Sida för att läsa meddelanden och skicka svar

För att se denna sida måste användaren klicka på ett meddelande. Det valda meddelandet visas och användaren ges möjlighet att svara till avsändaren.



Figur C.5: Visa skift

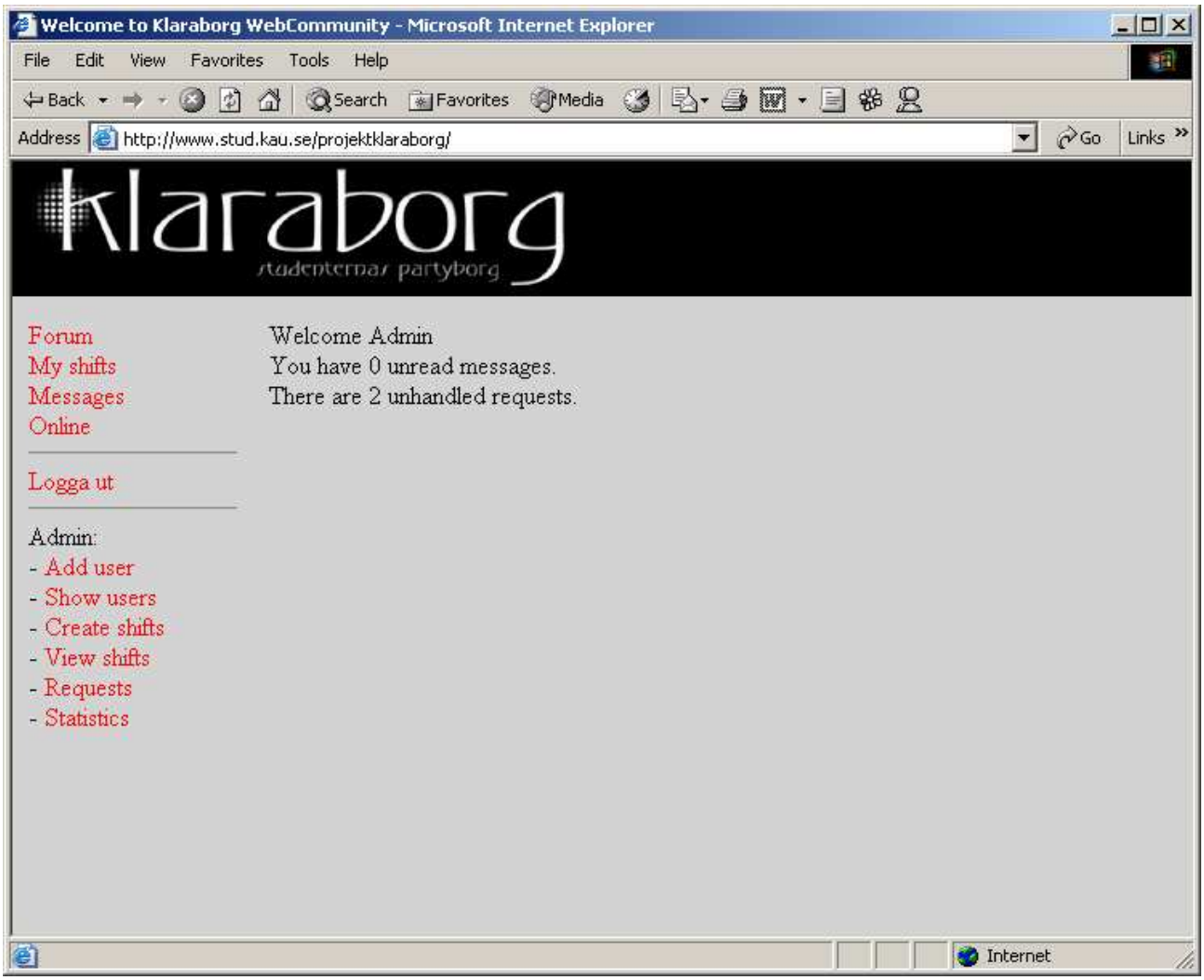
Här listas alla pass som användaren är inbokad på. Dessutom visas alla tillgängliga pass som är lediga. Användaren kan skicka förfrågningar om att bli avbokad från eget pass eller en förfrågan om att bli uppbokad på annat pass.



Figur C.6: Sida för detaljerad passinformation

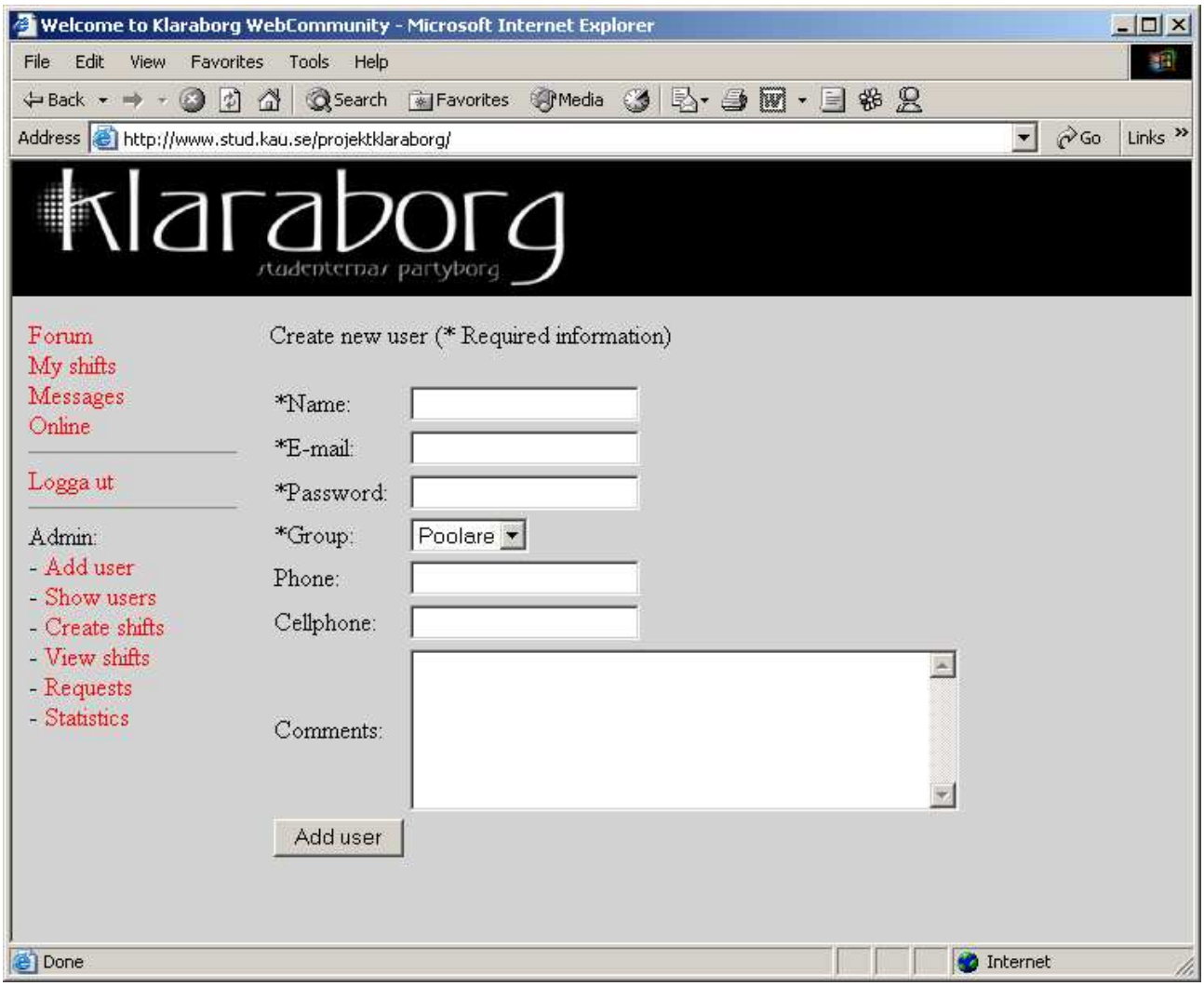
Här visas detaljerad information om ett pass, dessutom listas alla användare som är uppbokade.

C.2 Endast administratörer



Figur C.7: Administratörens startsida

Administratörens version av loginsidan visar dessutom hur många obehandlade förfrågningar som det finns i databasen. Se även *figur C.2* på sidan 20

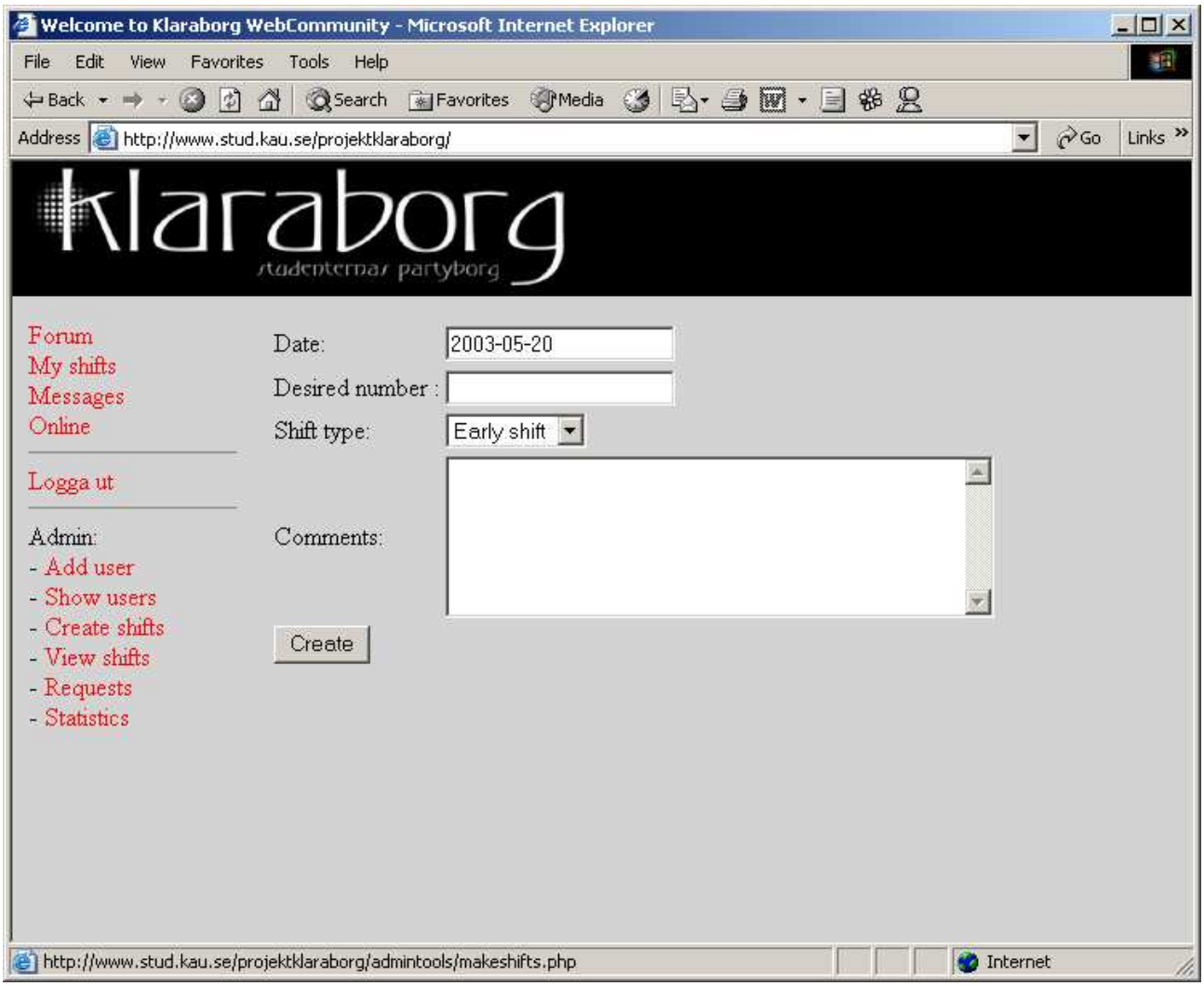


Figur C.8: Sida för att skapa användare

Administratören använder denna sida för att lägga till nya användare i systemet.

Figur C.9: Listar alla användare

Listar alla användare i systemet. Dessa är klickbara för att ge möjlighet att uppdatera användare. Se *figur C.8* på sidan 26.

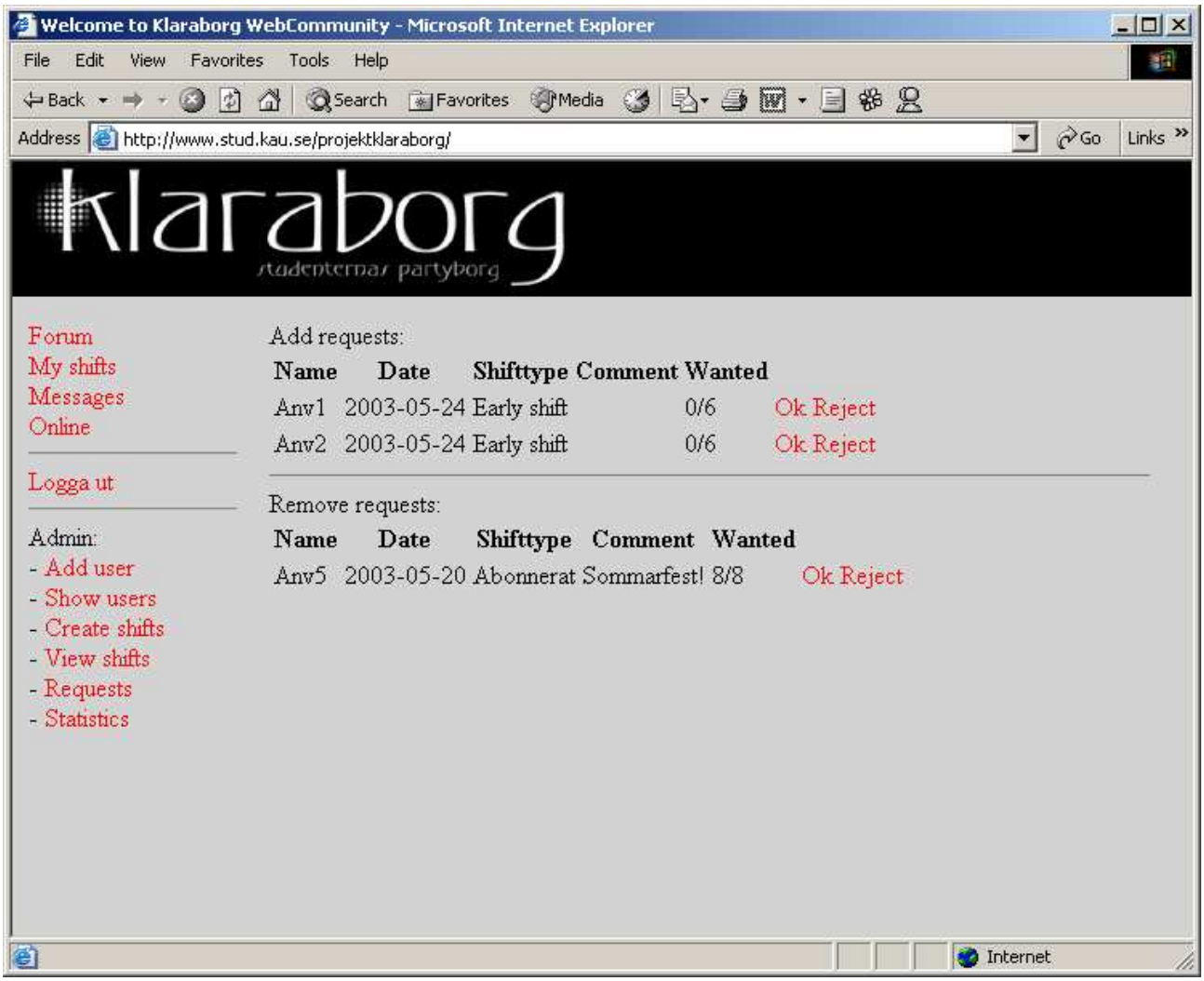


Figur C.10: Sida för att skapa pass

Här skapas de skift som användare kan boka upp sig på.

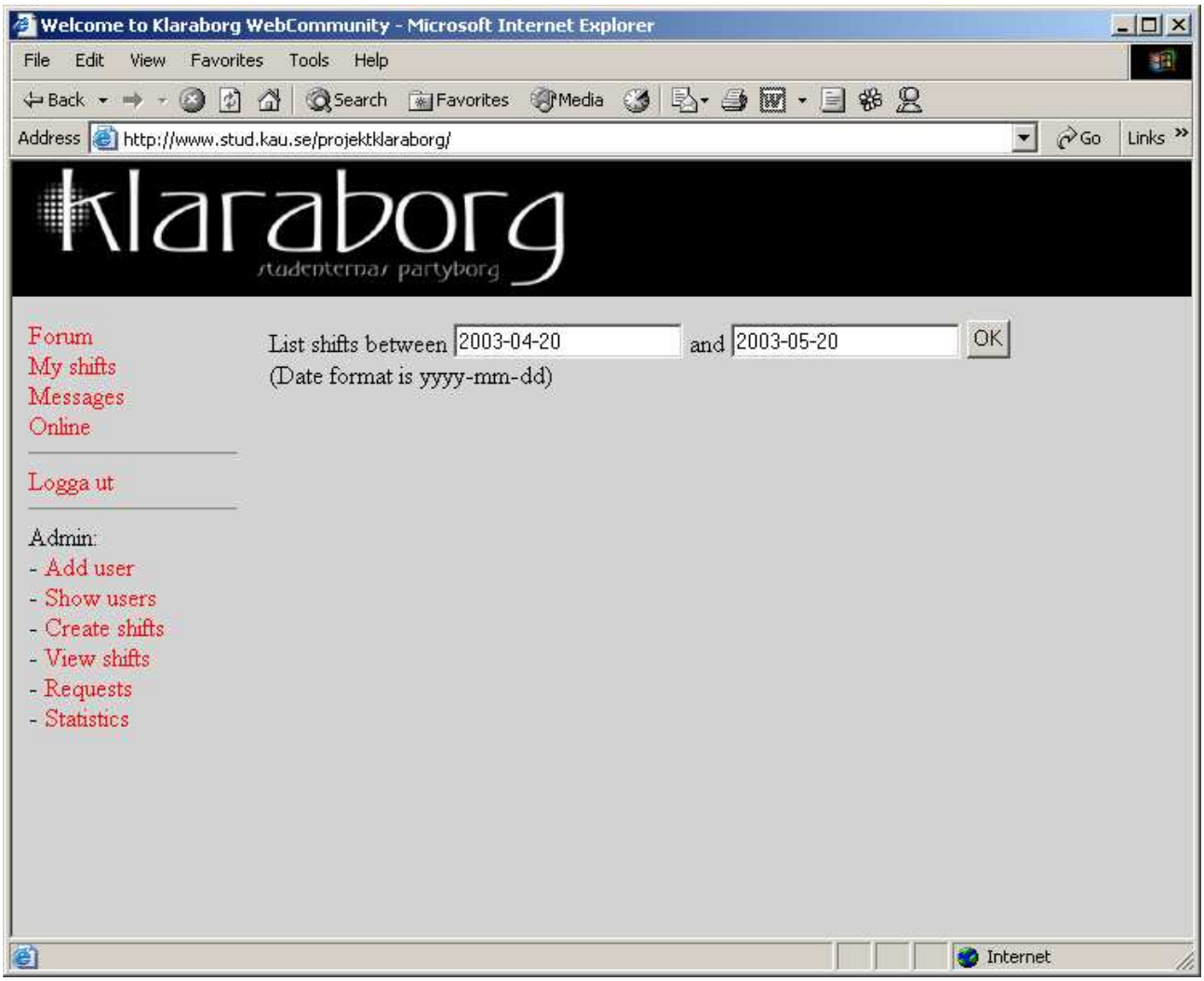
Figur C.11: Administratörens version av passinformationsidan

Förutom sidans vanliga funktionalitet (se *figur C.6* på sidan 24), kan en administratör även lägga till och ta bort användare från det aktuella passet.



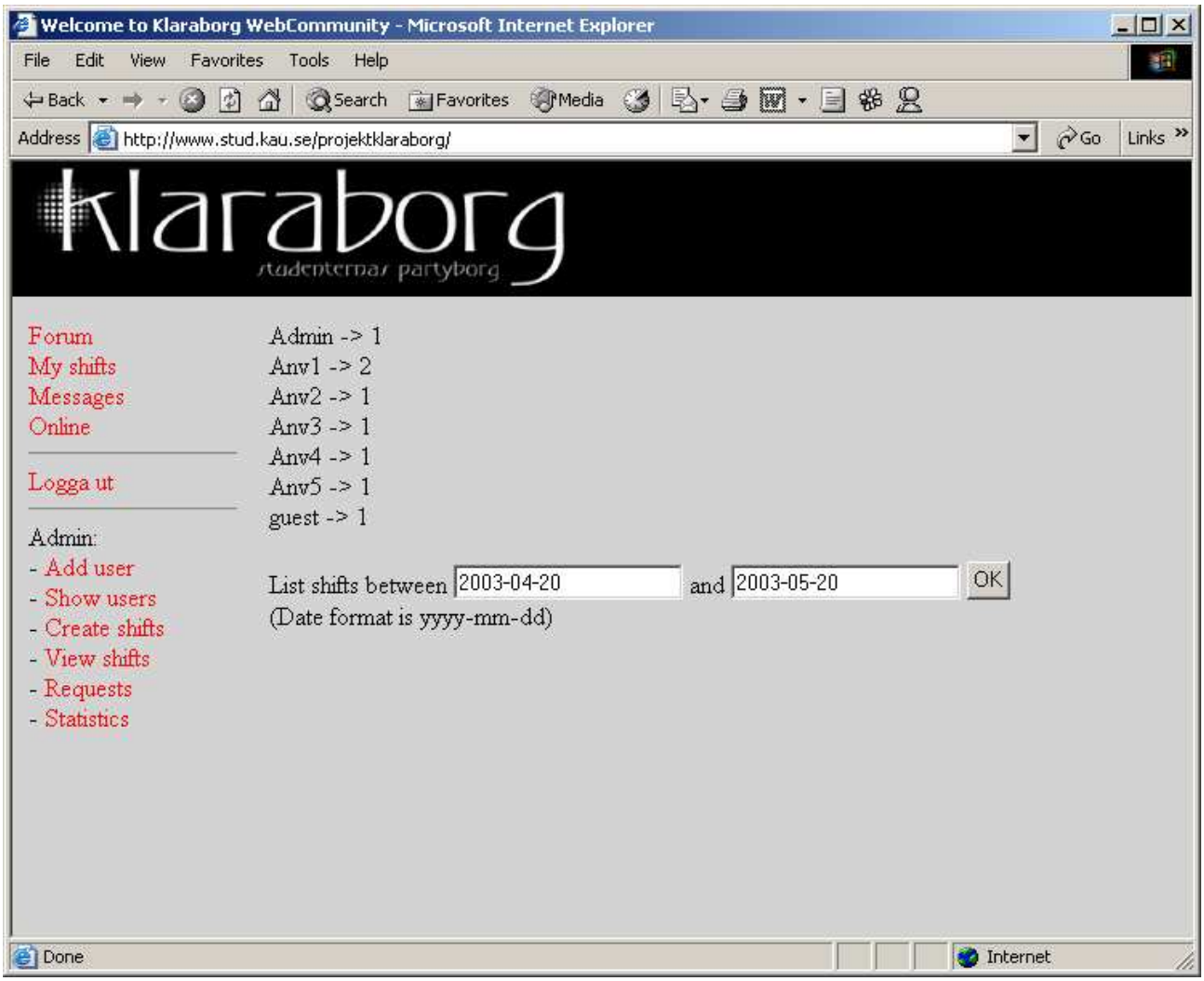
Figur C.12: Sidan där administratören hanterar förfrågningar

Här kan administratören hantera förfrågningar från användare. Förfrågningar kan accepteras eller avvisas. Ett meddelande kommer att skickas till användaren med information om beslutet.



Figur C.13: Startsidan för statistik

Sidan ger information om hur många gånger en poolare har arbetat under en given tidsperiod. Här skrivs ett startdatum och ett slutdatum in i formuläret. Se *figur C.14* på sidan 32 för resultatet av sökningen.



Figur C.14: Statistiken presenteras

Visar hur många gånger en poolare har arbetat under tidsperioden. Bara de som arbetat minst en gång presenteras.

D Källkod

D.1 main.php

```
<?php
include("security.php");
include("db.php");
include("scripts/html.php");

starthtml();
echo "Welcome $_SESSION[NAME]<BR>";

$sql = "SELECT COUNT(*) AS MESSNUM FROM MESSAGE WHERE UNREAD=1 AND TOID=".$_SESSION["USERID"];
$row = mysql_fetch_array(mysql_query($sql));
echo "You have $row[MESSNUM] unread messages.<BR>";

if ($_SESSION["ADMIN"])
{
$sql = "SELECT COUNT(*) AS REQNUM FROM REQUESTS";
$row = mysql_fetch_array(mysql_query($sql));
echo "There are $row[REQNUM] unhandled requests.<BR>";
}
endhtml();
?>
```

D.2 menu.php

```
<?php
/***   MENU.PHP   ***/
/*
Visar en menysida för användaren. Om denna tillhör ledningen
så visas även en administratörsmeny
*/

include("security.php");
include("scripts/html.php");

starthtml();
?>

<A HREF=forum.php TARGET=MAIN>Forum</A><BR>
<A HREF=shifts.php TARGET=MAIN>My shifts</A><BR>
<A HREF=message.php TARGET=MAIN>Messages</A><BR>
<A HREF=online.php TARGET=MAIN>Online</A><BR>
<HR>
<A HREF=logout.php TARGET="_top">Logga ut</A><BR>
<?php
if ($_SESSION["ADMIN"])
{
?>
<HR>
Admin:<BR>
 - <A HREF=admintools/add_user.php TARGET=MAIN>Add user</A><BR>
 - <A HREF=admintools/users.php TARGET=MAIN>Show users</A><BR>
 - <A HREF=admintools/makeshifts.php TARGET=MAIN>Create shifts</A><BR>
 - <A HREF=admintools/view_shift.php TARGET=MAIN>View shifts</A><BR>
 - <A HREF=admintools/requests.php TARGET=MAIN>Requests</A><BR>
 - <A HREF=admintools/statistics.php TARGET=MAIN>Statistics</A><BR>
<?
}
endhtml();
?>
```

D.3 do login.php

```
<?php
$link = mysql_connect("localhost", "projektklaraborg", "fEHKw87S")
        or die("Could not connect");
mysql_select_db("projektklaraborg");

$sql = "select USERID, GROUPID, PASS, NAME from USERS where EMAIL='$FORM_EMAIL'";
$result = mysql_query($sql);
$row = mysql_fetch_array($result);

if (md5($FORM_PASSWORD) == $row["PASS"])
{
   session_start();
   $_SESSION["USERID"] = $row["USERID"];
   $_SESSION["NAME"] = $row["NAME"];
```

```
   $_SESSION["ADMIN"] = ($row["GROUPID"] == 1);
   SetCookie ("LastEmail",$FORM_EMAIL, time()+60*60*24*30);
   header("Location: http://".$_SERVER['HTTP_HOST'].dirname($_SERVER['PHP_SELF']))."/".$relative_url."main_frames.html");
}
else
   header("Location: http://".$_SERVER['HTTP_HOST'].dirname($_SERVER['PHP_SELF']))."/".$relative_url."login.html");

?>
```

D.4 db.php

```
<?php
$link = mysql_connect("localhost", "projektklaraborg", "********")
        or die("Could not connect to database");
mysql_select_db("projektklaraborg");

?>
```

D.5 index.html

```
<HTML>
<HEAD>
   <TITLE>Welcome to Klaraborg WebCommunity</TITLE>
</HEAD>
<FRAMESET ROWS="85,*" COLS="*" FRAMEBORDER=0 FRAMESPACING=0>
   <FRAME NAME=LOGO SRC="logo.html" SCROLLING=no>
   <FRAME NAME=MAIN_FRAMES SRC="login.php">
</FRAMESET>
</HTML>
```

D.6 klaraborg.css

```
BODY
{
    BACKGROUND-COLOR: lightgrey
}
A:link
{
    COLOR: Red;
    text-decoration: none
}
A:visited
{
    COLOR: Red;
    text-decoration: none
}
A:hover
{
    COLOR: black;
    text-decoration: underline
}
.ROW1
{
    BACKGROUND-COLOR: #D3DCE3
}
```

D.7 logout.php

```
<?php
include("security.php");
include("db.php");
$sql = "UPDATE USERS SET ONLINE=0 WHERE USERID=".$_SESSION["USERID"];
mysql_query($sql);
session_unset();
session_destroy();
header("Location: http://".$_SERVER['HTTP_HOST']."/projektklaraborg/");
?>
```

D.8 main frames.html

```
<HTML>
<HEAD>
</HEAD>
<FRAMESET  ROWS="*" COLS="150,*" FRAMEBORDER=0 FRAMESPACING=0>
   <FRAME NAME = MENU SRC = "menu.php" scrolling=no>
   <FRAME NAME = MAIN SRC = "main.php">
</FRAMESET>
</HTML>
```

D.9 message.php

```
<?
/* Functions in this script:
** message_view($MESSAGEID)
** message_send($MESSAGEID)
** message_remove($MESSAGEID)
** message_save()
** message_list()
*/

include("scripts/html.php");
include("scripts/select.php");
include("security.php");
include("db.php");

starthtml();
?>
<?
if ($_POST["FORM_SUBMIT"])
{
message_save();
}

if ($REMOVEID)
{
message_remove($REMOVEID);
}

if ($MESSAGEID)
{
message_view($MESSAGEID);
message_send($MESSAGEID);
}
else
{
message_list();
message_send("0");
}
endhtml();
?>

<? function message_remove($REMOVEID)
{
$sql= "DELETE FROM MESSAGE WHERE MESSAGEID=$REMOVEID";
mysql_query($sql);
}
?>

<? function message_view($MESSAGEID)
{
mysql_query("UPDATE MESSAGE SET UNREAD=0 WHERE MESSAGEID=$MESSAGEID");
$sql = "SELECT NAME, SENTDATE, SUBJECT, MESSAGE FROM MESSAGE INNER JOIN USERS ON FROMID=USERID WHERE MESSAGEID=$MESSAGEID";
$row = mysql_fetch_array(mysql_query($sql));
?>
<TABLE>
<TR>
<TD>From:</TD>
<TD><?=$row["NAME"]?></TD>
</TR>
<TR>
<TD>Sent:</TD>
<TD><?=$row["SENTDATE"]?></TD>
</TR>
<TR>
<TD>Subject:</TD>
<TD><?=$row["SUBJECT"]?></TD>
</TR>
</TABLE>
Message:<BR>
<?=$row["MESSAGE"];?>
<?
echo "<BR>";
echo "<A HREF='$_SERVER[PHP_SELF]'>Back</A>";
}
?>

<? function message_save()
{
$sql = "INSERT INTO MESSAGE (FROMID, TOID, SUBJECT, MESSAGE, SENTDATE) VALUES(";
$sql = $sql.$_SESSION["USERID"].", ";
$sql = $sql.$_POST["FORM_TO"].", ";
$sql = $sql."'".$_POST["FORM_SUBJECT"]."', ";
$sql = $sql."'".$_POST["FORM_MESSAGE"]."', ";
```

```
$sql = $sql."NOW()".")";

if (mysql_query($sql))
echo "Message sent\n";
else
echo "Message NOT sent\n";
}
?>

<? function message_list()
{
$sql = "SELECT MESSAGEID, USERS.NAME AS FROMNAME, SUBJECT, UNREAD, SENTDATE FROM MESSAGE INNER JOIN USERS ON USERS.USERID=MESSAGE.FROMID"
"WHERE TOID=".$_SESSION["USERID"]." ORDER BY SENTDATE DESC";
$result = mysql_query($sql);
echo "<TABLE>";
echo "<TR>";
echo "<TH>Unread</TH>";
echo "<TH>From</TH>";
echo "<TH>Title</TH>";
echo "<TH>Sent</TH>";
echo "</TR>";
while($row = mysql_fetch_array($result))
{
echo "<A HREF='".$_SERVER[PHP_SELF]."?MESSAGEID=".$row["MESSAGEID"]."'>";
echo "<TR>";
echo "<TD>".($row["UNREAD"]==1?"x":"")."</TD>";
echo "<TD>".$row["FROMNAME"]."</TD>";
echo "<TD>".$row["SUBJECT"]."</TD>";
echo "<TD>".$row["SENTDATE"]."</TD>";
echo "<TD><A HREF='$_SERVER[PHP_SELF]?REMOVEID=".$row["MESSAGEID"]."'>Delete</A></TD>";
echo "</TR>";
echo "</A>";
}
echo "</TABLE>";
}
?>

<? function message_send($MESSAGEID)
{
echo "<HR>";
echo ($MESSAGEID==0?"Send new message:":"Reply to message:");
?>
   <FORM ACTION="<?=$_SERVER[PHP_SELF]?>" METHOD=POST>
   <TABLE>
   <TR>
      <TD>From:</TD>
      <TD><?=$_SESSION["NAME"];?></TD>
   </TR>
   <TR>
      <TD>To:</TD>
      <TD>
      <?
      if ($MESSAGEID==0)
      selectbox("SELECT NAME, USERID FROM USERS ORDER BY NAME", "FORM_TO", "NAME", "USERID", "");
   else
   {
    $sql = "SELECT NAME, FROMID FROM MESSAGE INNER JOIN USERS ON FROMID=USERID WHERE MESSAGEID=$MESSAGEID";
    $result = mysql_query($sql);
    $row = mysql_fetch_array($result);
    echo $row["NAME"];
    echo "<INPUT TYPE=HIDDEN NAME=FORM_TO VALUE=".$row["FROMID"].">";
   }?>
   </TD>
</TR>
<TR>
    <TD>Subject:</TD>
    <TD><INPUT TYPE=TEXT NAME="FORM_SUBJECT" <?
    if ($MESSAGEID!=0)
    {
    $sql = "SELECT SUBJECT FROM MESSAGE WHERE MESSAGEID=$MESSAGEID";
    $row = mysql_fetch_array(mysql_query($sql));
    echo "VALUE='RE: $row[SUBJECT]' ";
    }
    ?>WIDTH=50></TD>
</TR>
<TR><TD>Message:</TD></TR>
<TR><TD COLSPAN=2 ><TEXTAREA NAME="FORM_MESSAGE" ROWS=10 COLS=40></TEXTAREA></TD></TR>
<TABLE>
<INPUT TYPE=SUBMIT NAME="FORM_SUBMIT" VALUE="Send"
</FORM>
<?
}
?>
```

D.10 security.php

```
<?php
session_start();
if (!$_SESSION["NAME"])
{
header("Location: http://".$_SERVER['HTTP_HOST']."/projektklaraborg/");
}
?>
```

D.11 shift.php

```
<?
/** SHIFT.PHP */
/*
**ARGUMENTS:
**shiftid  (Q/F)
**FORM_USER  (F)
**FORM_REMOVE_USER  (F)
*/
include("scripts/html.php");
include("security.php");
include("scripts/select.php");
include("scripts/javascripts.php");
include("db.php");

starthtml();
if (!$shiftid)
{
usermessage("ERROR: No shift specified");
die("NO SHIFTID");
}
if ($FORM_USER && $_SESSION["ADMIN"])
{
$sql = "INSERT INTO SHIFTLIST (USERID, SHIFTID) VALUES($FORM_USER, $shiftid)";
if (!mysql_query($sql))
usermessage("ERROR: Could not add user, maybe user is already working");
}
if ($FORM_REMOVE_USER && $_SESSION["ADMIN"])
{
$sql = "DELETE FROM SHIFTLIST WHERE USERID=$FORM_REMOVE_USER AND SHIFTID=$shiftid";
mysql_query("$sql");
}

$sql = "SELECT * FROM SHIFTS INNER JOIN SHIFTTYPES ON SHIFTTYPES.SHIFTTYPEID=SHIFTS.SHIFTTYPEID WHERE SHIFTS.SHIFTID=$shiftid";
$result = mysql_query($sql);
$row = mysql_fetch_array($result);
echo "<TABLE>";
echo "<TR>";
echo    "<TD><B>Date</B></TD>";
echo    "<TD>".$row["DATE"]."</TD>";
echo "</TR>";
echo "<TR>";
echo    "<TD><B>Desired number of workers</B></TD>";
echo    "<TD>".$row["WANTED"]."</TD>";
echo "</TR>";
echo "<TR>";
echo    "<TD><B>Type</B></TD>";
echo    "<TD>".$row["NAME"]."</TD>";
echo "</TR>";
echo "<TR>";
echo    "<TD><B>Comment</B></TD>";
echo    "<TD>".$row["COMMENT"]."</TD>";
echo "</TR>";
echo "</TABLE>";
echo "<HR>";

//Find out how many are working
$sql = "SELECT COUNT(*) AS ANTAL FROM SHIFTLIST WHERE SHIFTID=$shiftid";
$result = mysql_query($sql);
$row = mysql_fetch_array($result);
echo "<B>Number of people working:</B> ".$row["ANTAL"]."<BR><BR>";

//List all users assigned to this shift
$sql = "SELECT USERS.NAME FROM USERS INNER JOIN SHIFTLIST ON USERS.USERID=SHIFTLIST.USERID WHERE SHIFTID=$shiftid ORDER BY USERS.NAME";
$result = mysql_query($sql);
while ($row = mysql_fetch_array($result))
{
echo $row["USERID"]." ".$row["NAME"]."<BR>";
}
?>
```

```
<? if ($_SESSION["ADMIN"])
{
?>
   <FORM ACTION="<?=$_SERVER[PHP_SELF]?>" METHOD=POST>
   <BR><B>Remove user:</B>
   <INPUT TYPE=HIDDEN NAME="shiftid" VALUE=<?=$shiftid?>>
   <?selectbox("SELECT USERS.USERID, USERS.NAME FROM USERS INNER JOIN SHIFTLIST ON USERS.USERID=SHIFTLIST.USERID WHERE SHIFTID=$shiftid ORDER BY USERS.NAME",
               "FORM_REMOVE_USER",
               "NAME",
               "USERID",
               "");?>
   <INPUT TYPE=SUBMIT VALUE=Remove>
   </FORM>
<HR>
<FORM ACTION="<?=$_SERVER[PHP_SELF]?>" METHOD=POST>
<B>Add user: </B>
<INPUT TYPE=HIDDEN NAME="shiftid" VALUE=<?=$shiftid?>>
<?selectbox("SELECT NAME, USERID FROM USERS ORDER BY NAME", "FORM_USER", "NAME", "USERID", "");?>
<INPUT TYPE=SUBMIT VALUE=Add>
</FORM>
<?
}
endhtml();
?>
```

D.12 shifts.php

```
<?
include("scripts/html.php");
include("security.php");
include("db.php");

starthtml();

if ($APPLY)
{
$sql = "INSERT INTO REQUESTS (USERID, SHIFTID, TYPE) VALUES (".$_SESSION["USERID"].", $APPLY, 1)";
mysql_query($sql);
}

if ($WITHDRAW)
{
$sql = "DELETE FROM REQUESTS WHERE TYPE=1 AND USERID=".$_SESSION["USERID"]." AND SHIFTID=$WITHDRAW";
mysql_query($sql);
}

if ($FREE)
{
$sql = "INSERT INTO REQUESTS (USERID, SHIFTID, TYPE) VALUES (".$_SESSION["USERID"].", $FREE, 2)";
//echo $sql."\n<BR>";
mysql_query($sql);

}

if ($REMOVE)
{
$sql = "DELETE FROM REQUESTS WHERE TYPE=2 AND USERID=".$_SESSION["USERID"]." AND SHIFTID=$REMOVE";
//echo $sql."\n<BR>";
mysql_query($sql);
}


$sql = "SELECT SHIFTS.SHIFTID AS SID, NAME, DATE, COMMENT FROM SHIFTS INNER JOIN SHIFTLIST ON SHIFTS.SHIFTID=SHIFTLIST.SHIFTID "
       "INNER JOIN SHIFTTYPES ON SHIFTS.SHIFTTYPEID=SHIFTTYPES.SHIFTTYPEID WHERE DATE>=NOW() AND USERID=".$_SESSION["USERID"];
$result = mysql_query($sql);
echo "My upcomming shifts:<BR>";
while($row = mysql_fetch_array($result))
{
echo $row["DATE"].": ".$row["NAME"]." (".$row["COMMENT"].")  <A HREF=shift.php?shiftid=".$row["SID"].">Details ";
$row2 = mysql_fetch_array(mysql_query("SELECT USERID FROM REQUESTS WHERE TYPE=2 AND USERID=".$_SESSION["USERID"]." AND SHIFTID=".$row["SID"]));
if ($row2["USERID"]=="")
linkself("FREE", $row["SID"], "Request removal");
else
linkself("REMOVE", $row["SID"], "Withdraw request");
echo "<BR>";

}
echo "<HR>";
echo "Available shifts:<BR>";

$sql = "SELECT SHIFTS.SHIFTID AS SHIFTID, NAME, DATE, WANTED, COMMENT FROM SHIFTS "
       "INNER JOIN SHIFTTYPES ON SHIFTS.SHIFTTYPEID=SHIFTTYPES.SHIFTTYPEID WHERE DATE>=NOW() ORDER BY DATE, SHIFTS.SHIFTTYPEID";
```

```
$result = mysql_query($sql);
while($row = mysql_fetch_array($result))
{
$row2 =  mysql_fetch_array(mysql_query("SELECT COUNT(*) AS POOL FROM SHIFTLIST WHERE SHIFTID=".$row["SHIFTID"]));
$row3 =  mysql_fetch_array(mysql_query("SELECT USERID FROM SHIFTLIST WHERE USERID=".$_SESSION["USERID"]." AND SHIFTID=".$row["SHIFTID"]));
if (/*$row["WANTED"] > $row2["POOL"] && */$row3["USERID"] == "") //Folk behövs
{
echo $row["DATE"].": ".$row["NAME"]." (".$row["COMMENT"].")[".$row2["POOL"]."/".$row["WANTED"]."] <A HREF=shift.php?shiftid=$row[SHIFTID]>Details </A>";
$row4 = mysql_fetch_array(mysql_query("SELECT USERID FROM REQUESTS WHERE TYPE=1 AND USERID=".$_SESSION["USERID"]." AND SHIFTID=".$row["SHIFTID"]));
if ($row4["USERID"]=="")
linkself("APPLY", $row["SHIFTID"], "Request work");
else
linkself("WITHDRAW", $row["SHIFTID"], "Withdraw request");
echo "<BR>";
}
}
endhtml();
?>
```

D.13 add user.php

```
<?php
/***   ADD_USER.PHP   ***/
/* Sidan används för att skapa nya användare */

include("../db.php");
include("../scripts/select.php");
session_start();
?>
<HTML>
<HEAD>
<LINK REL=stylesheet HREF="/projektklaraborg/klaraborg.css" TYPE="text/css">
</HEAD>
<BODY>
<?php
if ($FORM_SUBMIT)
{
if ($FORM_NAME == "" || $FORM_EMAIL == "" || $FORM_PASS == "")
echo ("Du måste skriva in i alla fält markerade med *<BR>");
else
{
   $sql = "SELECT USERID FROM USERS WHERE EMAIL='$FORM_EMAIL'";
   $result = mysql_query($sql);
   $row = mysql_fetch_array($result);
   if ($row["USERID"] == "")
   {
    $passord = md5($FORM_PASS);
    $sql = "INSERT INTO USERS (EMAIL, PASS, GROUPID, NAME, COMMENT, PHONE, MOBILE) "
       "VALUES ('$FORM_EMAIL', '$passord', '$FORM_GROUP', '$FORM_NAME', '$FORM_COMMENT', '$FORM_PHONE', '$FORM_MOBILE')";
    if (mysql_query($sql))
      echo "User created. <BR>";
    else
       echo "Error occured during creation!<BR>";
   }
   else
    echo "E-mail already in database!<BR>";

}
}
?>
Create new user (* Required information)

<FORM NAME=ADDUSER METHOD=POST ACTION="<?=$PHP_SELF?>">
<TABLE>
<TR>
   <TD>*Name: </TD>
   <TD><INPUT TYPE=TEXT NAME=FORM_NAME></TD>
</TR>

<TR>
   <TD>*E-mail: </TD>
   <TD><INPUT TYPE=TEXT NAME=FORM_EMAIL></TD>
</TR>
<TR>
   <TD>*Password: </TD>
   <TD><INPUT TYPE=TEXT NAME=FORM_PASS></TD>
</TR>
<TR>
   <TD>*Group: </TD>
   <TD>
   <?selectbox("SELECT * FROM GROUPS", "FORM_GROUP", "NAME", "GROUPID", 4);?>
   </TD>
```

```
</TR>
<TR>
   <TD>Phone: </TD>
   <TD><INPUT TYPE=TEXT NAME=FORM_PHONE></TD>
</TR>
<TR>
   <TD>Cellphone: </TD>
   <TD><INPUT TYPE=TEXT NAME=FORM_MOBILE></TD>
</TR>

<TR>
   <TD>Comments: </TD>
   <TD><TEXTAREA NAME=FORM_COMMENT ROWS=6 COLS=40></TEXTAREA></TD>
</TR>

<TR>
   <TD><INPUT TYPE=SUBMIT NAME=FORM_SUBMIT VALUE="Add user"></TD>
</TR>

</FORM>
</BODY>
</HTML>
```

D.14 makeshifts.php

```
<?php
include("../db.php");
?>


<?php
if ($FORM_SUBMIT)
{
   $sql = "INSERT INTO SHIFTS (DATE, WANTED, SHIFTTYPEID, COMMENT) VALUES('$FORM_DATE', $FORM_WANTED, $FORM_SHIFTTYPE, '$FORM_COMMENT')";
   if (!mysql_query($sql))
    echo ("An error has occured, check input and try again<BR>");
   else
    echo ("Shift created <BR>");
}
?>
<HTML>
<HEAD>
<LINK REL=stylesheet HREF="../klaraborg.css" TYPE="text/css">
</HEAD>
<BODY>
<FORM METHOD=POST ACTION="<?=$PHP_SELF?>">
<TABLE>
<TR>
   <TD>Date:</TD>
   <TD><INPUT TYPE=TEXTBOX NAME=FORM_DATE VALUE="<?=date("Y-m-d");?>"></TD>
</TR>
<TR>
  <TD>Desired number :</TD>
  <TD><INPUT TYPE=TEXTBOX NAME=FORM_WANTED></TD>
</TR>
<TR>
<TD>Shift type:</TD>
<TD>
   <SELECT NAME=FORM_SHIFTTYPE>
   <?php
   $sql = "SELECT * from SHIFTTYPES";
   $result = mysql_query($sql);
   while($row = mysql_fetch_array($result))
   {
      echo ("<OPTION VALUE=".$row["SHIFTTYPEID"].">".$row["NAME"]."</OPTION>\n");
   }
   ?>
   </SELECT>
</TD>
</TR>
<TR>
   <TD>Comments: </TD>
   <TD><TEXTAREA NAME=FORM_COMMENT ROWS=6 COLS=40></TEXTAREA></TD>
</TR>
<TR>
<TD><INPUT TYPE=SUBMIT VALUE="Create" NAME=FORM_SUBMIT></TD>
</FORM>
</BODY>
</HTML>
```

D.15 requests.php

```
<?
include("../db.php");
include("../security.php");
include("../scripts/html.php");

starthtml();

if ($REMOVE && $USER)
{
$sql = "DELETE FROM SHIFTLIST WHERE USERID=$USER AND SHIFTID=$REMOVE";
if (mysql_query($sql))
{
echo "-> Shift removed<BR>";
$sql = "DELETE FROM REQUESTS WHERE USERID=$USER AND SHIFTID=$REMOVE";
//echo $sql."\n<BR>";
mysql_query($sql);
$sql = "SELECT * FROM SHIFTS INNER JOIN SHIFTTYPES ON SHIFTS.SHIFTTYPEID=SHIFTTYPES.SHIFTTYPEID WHERE SHIFTID=$REMOVE";
//echo $sql."\n<BR>";
$row = mysql_fetch_array(mysql_query($sql));
$sql = "INSERT INTO MESSAGE (FROMID, TOID, SUBJECT, MESSAGE, SENTDATE) VALUES(".$_SESSION["USERID"].
       ", $USER, 'Removal approved', 'You have been removed from the pool for $row[NAME], $row[DATE]', NOW())";
//echo $sql."\n<BR>";
mysql_query($sql);
}
else
echo "-> Shift NOT removed<BR>";
}
if ($REJECT_REMOVE && $USER)
{
$sql = "DELETE FROM REQUESTS WHERE USERID=$USER AND SHIFTID=$REJECT_REMOVE";
mysql_query($sql);
$sql = "SELECT * FROM SHIFTS INNER JOIN SHIFTTYPES ON SHIFTS.SHIFTTYPEID=SHIFTTYPES.SHIFTTYPEID WHERE SHIFTID=$REJECT_REMOVE";
$row = mysql_fetch_array(mysql_query($sql));
$sql = "INSERT INTO MESSAGE (FROMID, TOID, SUBJECT, MESSAGE, SENTDATE) VALUES(".$_SESSION["USERID"].
               ", $USER, 'Removal rejected!', 'Your request for removal on $row[NAME], $row[DATE] has been rejected', NOW())";
mysql_query($sql);
}

if ($ASSIGN && $USER)
{
$sql = "INSERT INTO SHIFTLIST (USERID, SHIFTID) VALUES ($USER, $ASSIGN)";
//echo $sql."\n<BR>";
if (mysql_query($sql))
{
echo "-> Shift added<BR>";
$sql = "DELETE FROM REQUESTS WHERE USERID=$USER AND SHIFTID=$ASSIGN";
//echo $sql."\n<BR>";
mysql_query($sql);
$sql = "SELECT * FROM SHIFTS INNER JOIN SHIFTTYPES ON SHIFTS.SHIFTTYPEID=SHIFTTYPES.SHIFTTYPEID WHERE SHIFTID=$ASSIGN";
//echo $sql."\n<BR>";
$row = mysql_fetch_array(mysql_query($sql));
$sql = "INSERT INTO MESSAGE (FROMID, TOID, SUBJECT, MESSAGE, SENTDATE) VALUES(".$_SESSION["USERID"].
                      ", $USER, 'Work approved', 'You have been added to the pool for $row[NAME], $row[DATE]', NOW())";
//echo $sql."\n<BR>";
mysql_query($sql);
}
else
echo "-> Shift NOT added<BR>";
}

if ($REJECT_ASSIGN && $USER)
{
$sql = "DELETE FROM REQUESTS WHERE USERID=$USER AND SHIFTID=$REJECT_ASSIGN";
mysql_query($sql);
$sql = "SELECT * FROM SHIFTS INNER JOIN SHIFTTYPES ON SHIFTS.SHIFTTYPEID=SHIFTTYPES.SHIFTTYPEID WHERE SHIFTID=$REJECT_ASSIGN";
$row = mysql_fetch_array(mysql_query($sql));
$sql = "INSERT INTO MESSAGE (FROMID, TOID, SUBJECT, MESSAGE, SENTDATE) VALUES(".$_SESSION["USERID"].
               ", $USER, 'Work rejected!', 'Your request for work on $row[NAME], $row[DATE] has been rejected', NOW())";
mysql_query($sql);
}
request_list();
endhtml();
?>

<?
function request_list()
{
echo "Add requests:";
$sql = "SELECT REQUESTS.USERID AS UID, REQUESTS.SHIFTID AS SID, USERS.NAME AS UNAME, WANTED, DATE, SHIFTTYPES.NAME AS SNAME, SHIFTS.COMMENT AS SCOMMENT "
               "FROM REQUESTS INNER JOIN SHIFTS ON REQUESTS.SHIFTID=SHIFTS.SHIFTID INNER JOIN SHIFTTYPES ON SHIFTS.SHIFTTYPEID=SHIFTTYPES.SHIFTTYPEID "
               "INNER JOIN USERS ON REQUESTS.USERID=USERS.USERID WHERE TYPE=1";
$result = mysql_query($sql);
```

```
echo "<TABLE>";
echo "<TR>";
echo "<TH>Name</TH>";
echo "<TH>Date</TH>";
echo "<TH>Shifttype</TH>";
echo "<TH>Comment</TH>";
echo "<TH>Wanted</TH>";
echo "</TR>";
while($row = mysql_fetch_array($result))
{
echo "<TR>";
echo "<TD>".$row["UNAME"]."</TD>";
echo "<TD>".$row["DATE"]."</TD>";
echo "<TD>".$row["SNAME"]."</TD>";
echo "<TD>".$row["SCOMMENT"]."</TD>";
$sql = "SELECT COUNT(*) AS C FROM SHIFTLIST WHERE SHIFTID=$row[SID]";
$row2 = mysql_fetch_array(mysql_query($sql));
echo "<TD>".$row2["C"]."/".$row["WANTED"]."</TD>";
echo "<TD><A HREF='$_SERVER[PHP_SELF]?ASSIGN=".$row["SID"]."&USER=".$row["UID"]."'>Ok</A></TD>";
echo "<TD><A HREF='$_SERVER[PHP_SELF]?REJECT_ASSIGN=".$row["SID"]."&USER=".$row["UID"]."'>Reject</A></TD>";
echo "</TR>";
}
echo "</TABLE>";

echo "<HR>";

echo "Remove requests:";
$sql = "SELECT REQUESTS.USERID AS UID, REQUESTS.SHIFTID AS SID, USERS.NAME AS UNAME, WANTED, DATE, SHIFTTYPES.NAME AS SNAME, SHIFTS.COMMENT AS SCOMMENT "
               "FROM REQUESTS INNER JOIN SHIFTS ON REQUESTS.SHIFTID=SHIFTS.SHIFTID INNER JOIN SHIFTTYPES ON SHIFTS.SHIFTTYPEID=SHIFTTYPES.SHIFTTYPEID "
               "INNER JOIN USERS ON REQUESTS.USERID=USERS.USERID WHERE TYPE=2";
$result = mysql_query($sql);
echo "<TABLE>";
echo "<TR>";
echo "<TH>Name</TH>";
echo "<TH>Date</TH>";
echo "<TH>Shifttype</TH>";
echo "<TH>Comment</TH>";
echo "<TH>Wanted</TH>";
echo "</TR>";
while($row = mysql_fetch_array($result))
{
echo "<TR>";
echo "<TD>".$row["UNAME"]."</TD>";
echo "<TD>".$row["DATE"]."</TD>";
echo "<TD>".$row["SNAME"]."</TD>";
echo "<TD>".$row["SCOMMENT"]."</TD>";
$sql = "SELECT COUNT(*) AS C FROM SHIFTLIST WHERE SHIFTID=$row[SID]";
$row2 = mysql_fetch_array(mysql_query($sql));
echo "<TD>".$row2["C"]."/".$row["WANTED"]."</TD>";
echo "<TD><A HREF='$_SERVER[PHP_SELF]?REMOVE=".$row["SID"]."&USER=".$row["UID"]."'>Ok</A></TD>";
echo "<TD><A HREF='$_SERVER[PHP_SELF]?REJECT_REMOVE=".$row["SID"]."&USER=".$row["UID"]."'>Reject</A></TD>";
echo "</TR>";
}
echo "</TABLE>";
}
?>
```

D.16 users.php

```
<?php
include("../db.php");
include("../scripts/html.php"); //for htmlstuff
include("../scripts/select.php");  //for selectbox

function listusers()
{
$sql = "SELECT USERID,EMAIL,GROUPS.NAME,USERS.NAME,COMMENT,PHONE,MOBILE FROM USERS INNER JOIN GROUPS ON GROUPS.GROUPID = USERS.GROUPID ORDER BY USERS.NAME";
$result = mysql_query($sql);
echo "<TABLE>\n";
while($row = mysql_fetch_array($result))
{
echo ("<TR>\n");
for ($i = 1; $i < count($row); $i++)
{
   if ($i == 1)
echo ("<A HREF='$_SERVER[PHP_SELF]?USERID=$row[0]'>");
   echo ("<TD>".$row[$i]."</TD>\n");
       if ($i == 1)
echo ("</A>");
}
echo ("</TR>\n");
};
```

```
echo "</TABLE>\n";
}

function showuser($id)
{

$sql = "SELECT * FROM USERS WHERE USERID=$id";
$result = mysql_query($sql);
$row = mysql_fetch_array($result);
?>
<FORM NAME=UPDATEUSER METHOD=POST ACTION='<?=$_SERVER["PHP_SELF"]?>'>
<TABLE>
<INPUT TYPE=HIDDEN NAME=USERID VALUE='<?=$id?>'>
<TR>
    <TD>Namn: </TD>
    <TD><INPUT TYPE=TEXT NAME=FORM_NAME VALUE='<?=$row["NAME"]?>'></TD>
</TR>
<TR>
    <TD>Email: </TD>
    <TD><INPUT TYPE=TEXT NAME=FORM_EMAIL VALUE='<?=$row["EMAIL"]?>'></TD>
</TR>
<TR>
   <TD>Password: *</TD>
   <TD><INPUT TYPE=TEXT NAME=FORM_PASS></TD>
</TR>
<TR>
   <TD>Grupp: </TD>
   <TD>
   <?selectbox("SELECT * from GROUPS", "FORM_GROUP", "NAME", "GROUPID", $row["GROUPID"]);?>
   </TD>
</TR>
<TR>
   <TD>Telnummer: </TD>
   <TD><INPUT TYPE=TEXT NAME=FORM_PHONE VALUE='<?=$row["PHONE"]?>'></TD>
</TR>
<TR>
   <TD>Mobilnummer: </TD>
   <TD><INPUT TYPE=TEXT NAME=FORM_MOBILE VALUE='<?=$row["MOBILE"]?>'></TD>
</TR>
<TR>
   <TD>Kommentarer: </TD>
   <TD><TEXTAREA NAME=FORM_COMMENT ROWS=6 COLS=40><?=$row["COMMENT"]?></TEXTAREA></TD>
</TR>
<TR>
    <TD><INPUT TYPE=SUBMIT NAME=FORM_SUBMIT VALUE="Uppdatera"></TD>
</TR>
</FORM>
 * Skriv nytt för att förändra, annars lämna tomt
<?
}
function updateuser($id)
{
$sql = "UPDATE USERS SET ";
$sql = $sql."NAME='$_POST[FORM_NAME]', ";
$sql = $sql."EMAIL='$_POST[FORM_EMAIL]', ";
if ($_POST[FORM_PASS] != "")
{
$pass = md5($_POST[FORM_PASS]);
$sql = $sql."PASS='$pass', ";
}
$sql = $sql."PHONE='$_POST[FORM_PHONE]', ";
$sql = $sql."MOBILE='$_POST[FORM_MOBILE]', ";
$sql = $sql."COMMENT='$_POST[FORM_COMMENT]', ";
$sql = $sql."GROUPID=$_POST[FORM_GROUP]";
$sql = $sql." WHERE USERID=$id";
if (mysql_query($sql))
echo "Uppdateringen lyckades<BR>";
else
echo "Uppdateringen misslyckades<BR>";
}

starthtml();
if ($USERID)
{
if($FORM_SUBMIT)
{
updateuser($USERID);
listusers();
}

else
showuser($USERID);
}
```

```
else
{
listusers();
}
endhtml();
?>
```

D.17 view shift.php

```
<?php
include("../db.php");
include("../security.php");
?>
<HTML>
<HEAD>
<LINK REL=stylesheet HREF="../klaraborg.css" TYPE="text/css">
</HEAD>
<BODY>

<?
$sql = "SELECT DATE, NAME, WANTED, SHIFTID FROM SHIFTS INNER JOIN SHIFTTYPES ON SHIFTS.SHIFTTYPEID=SHIFTTYPES.SHIFTTYPEID";
$result = mysql_query($sql);
?>
<TABLE>
<?php
echo ("<TR>\n");
echo ("<TD> </TD>");
echo ("<TD>DATE</TD>\n");
echo ("<TD>NAME</TD>\n");
echo ("<TD>WANTED</TD>\n");
echo ("<TD>WORKING</TD>\n");
//echo ("<TD>SHIFTID</TD>\n");
echo ("</TR>");


while($row = mysql_fetch_array($result))
{
echo "<TD><A HREF=../shift.php?shiftid=".$row["SHIFTID"].">?</A>";
echo ("<TD>".$row[DATE]."</TD>\n");
echo ("<TD>".$row[NAME]."</TD>\n");
echo ("<TD>".$row[WANTED]."</TD>\n");
$sql2 = "SELECT COUNT(*) AS C FROM SHIFTLIST WHERE SHIFTID='$row[SHIFTID]'";
$row2 = mysql_fetch_array(mysql_query($sql2));
echo ("<TD>".$row2[C]."</TD>\n");

/* for ($i = 0; $i < count($row); $i++)
   echo );*/
echo ("</TR>\n");
};
?>
</TABLE>
</BODY>
</HTML>
```

D.18 html.php

```
<?php
function starthtml()
{
?>
<HTML>
<HEAD>
<LINK REL=stylesheet HREF="http://<?=$_SERVER['HTTP_HOST']?>/projektklaraborg/klaraborg.css" TYPE="text/css">
</HEAD>
<BODY>
<?
}
function endhtml()
{
echo "</BODY>\n";
echo "</HTML>\n";
}
function linkself($VARNAME, $VARVALUE, $MESS)
{
echo "<A HREF='$_SERVER[PHP_SELF]?$VARNAME=$VARVALUE'>$MESS</A>";
}
?>
```

D.19 javascripts.php

```
<?
function usermessage($MESSAGE)
{
?>
<SCRIPT LANGUAGE="JAVASCRIPT">
alert("<?=$MESSAGE?>");
</SCRIPT>
<?
}
?>
```

D.20 select.php

```
<?php
function selectbox($sql, $formname, $namecolumn, $datacolumn, $default)
{
$result = mysql_query($sql);
   echo ("<SELECT NAME=$formname>\n");
   while($row = mysql_fetch_array($result))
   {
      echo ("<OPTION ");
      if ($row[$datacolumn] == $default)
         echo ("SELECTED ");
      echo ("VALUE=".$row[$datacolumn].">".$row[$namecolumn]."</OPTION>\n");
   }
   echo ("</SELECT>\n");
}
?>
```