Webbaseret logokonkurrence

Claus Jensen
Datalogisk Institut, Københavns Universitet,
Universitetsparken 1, 2100 København Ø, Danmark
surf@diku.dk

Indholdsfortegnelse

1 Problemspecifikation

1.1 Projektmål

Projektets primære mål er at skabe et system bestående af en række websider der gør følgende muligt:

- Se og stemme på en række logoforslag.
- Mindske muligheden for at samme person kan afgive flere stemmer.
- Tilføje nye logoforslag.
- Fremvise en statistik over afgivne stemmer på de forskellige logoer.

Det sekundær mål for projektet er at opnå et større kendskab til programmeringssproget Perl, specielt i forbindelse med webprogrammering. Det er samtidigt et mål, at afprøve hvilke muligheder der ligger i at bruge Perl moduler til at understøtte programmering og designopgaver i forbindelse med udviklingen af websider.

1.1.1 Hvad skal systemet bruges til?

Systemet skal bruges til at afgøre hvilket logo der skal være det fremtidige logo for CPH STL. Logoet skal blandt andet optræde på CPH STLs websider.

1.1.2 Hvad skal systemet kunne?

Systemets indgangsside skal indeholde følgende hovedkomponenter:

- Et billede af den T-shirt som kan vindes i forbindelse med afstemningen.
- En samling af de logoer som det er mulig at stemme på. De forskellige logoer skal fremstå indbydes på en så fair måde som mulig.
- Et link til siden der tildeler brugeren en adgangskode.
- Et link der gennem en loginside fører brugeren til afstemningssiden.
- Et link der gennem loginsiden fører brugeren til en side hvor brugerens egne logo kan tilføjes logokonkurrencen.
- Et link til statistik-siden hvor det er mulig at se hvor mange stemmer de deltagende logo har modtaget.

Siden som tildeler brugeren en adgangskode skal indeholde et felt hvor brugeren kan indskrive sin e-mail adresse. Når systemet har modtaget brugerens e-mail adresse skal det fremstille en adgangskode og afsende den til brugeren.

Loginsiden skal verificere at brugeren er kendt af systemet og har adgang til at stemme på et logo eller tilføje et logo.

Afstemningssiden skal give brugen mulighed for at stemme på et logo. Udover at stemme skal brugeren også have mulighed for at angive hvilken størrelse T-shirt hun ønsker, hvis hun vinder en T-shirt. Det skal derudover også være muligt at sende en hilsen til CPH STL udviklerne. Brugeren skal have mulighed for at ændre sine valg efter at have afgivet sin stemme.

Siden hvor brugerne kan tilføje deres egne logo skal indeholde muligheden for at sende (uploade) et forslag til systemet.

Statistik-siden skal give en oversigt over hvor mange stemmer de forskellige logoer har modtaget i løbet af afstemningen. Samtidig skal den vise hvor mange procent af det samlede antal stemmer de enkelte logo har modtaget.

1.1.3 Brugeren

Brugeren af systemet forudsættes at være en almindelig Internet bruger. Systemet kan derfor ikke indeholde funktionalitet som kræver at brugeren har et kendskab der overstiger denne brugergruppes kvalifikationer.

2 Analyse

2.1 Inddata

Slutbrugeren kan levere flere forskellige typer inddata til systemet. Det kan være oplysninger om brugeren selv, brugerens stemme eller et af brugerens logoer.

Før brugeren kan afgive sin stemme i logokonkurrencen skal vedkommende afgive et sæt oplysninger, der så godt det er muligt, entydigt identificer brugeren. Dette sker for at undgå at den samme person afgiver flere stemmer i konkurrencen. Da der på nuværende tidspunkt ikke i større udstrækning eksistere digitale signaturer, vil det være mest nærliggende at bruge brugerens e-mail adresse til at identificere brugeren. Brugen af e-mail adresser sikrer ikke en optimal løsning, fordi en bruger kan godt have flere e-mail adresser. Brugen af e-mail sikrer derimod at hver bruger kun har et begrænset antal stemmer.

I forbindelse med afstemningen kan brugerens levere følgende inddata til systemet:

- Hvilket logo brugeren har stemt på.
- Hvilken størrelse T-shirt brugeren ønsker hvis denne vinder T-shirt konkurrencen.
- Hilsner eller meddelelser til CPH STL udviklingsteam.

Hvis brugeren vil tilføje et logo til logokonkurrencen skal brugeren levere logoet i form af et billede som kan vises af en webbrowser. Hvis det er teknisk mulig og ikke er for omkostningskrævende vil det blive forsøgt at aflæse billedes pixel højde og bredde. Hvis det ikke er muligt at foretage denne aflæsning vil det være nødvendig at brugeren også leverer billedes højde og bredde i pixel som inddata.

Alle de ovennævnte inddata vil så længe konkurrencen er i gang blive opbevaret af systemet, og det vil, med undtagelse af e-mail adressen, være muligt for brugeren at ændre disse hvis hun skulle ombestemme sig.

2.2 Systemdata

For at brugerens adgang til systemet kan sikres skal brugeren også tildeles en adgangskode. Det er systemets ansvar at fremstille og opbevare denne adgangskode.

Det er både i forbindelse med afstemningen og i forbindelse med visningen af logoerne vigtigt at de forskellige logoer får et unikt navn. Derfor er fremstillingen og opbevaringen af navne, til de af brugeren leveret logoer, overladt til systemet.

Da brugeren, når hun har fået sin adgangskode, frit kan lægge billeder ind i systemet, kan det anbefales at have styr på hvilke billeder man gerne vil vise. Derfor introduceres en variabel der styrer om et billede skal vises. Variablen kan default sættes til sand eller falsk alt efter webadministratorens ønske.

2.3 Funktion

Systemet skal kunne gemme de data som det har modtaget fra brugeren. Systemet skal også kunne vise data modtaget fra brugeren og andre evt. i behandlet form.

Visningen af konkurrencens forskellige logoer skal foregå på en så fair måde som mulig. En måde at løse dette problem er at forlange af konkurrencedeltagernes logoer skal indpasses efter en bestemt størrelse skabelon. Denne skabelon, eller skabeloner, fastsætter på forhånd højden og bredden af logoet. En anden måde er at fastlægge et areal som logoet skal fylde. Det giver så igen to muligheder: den første er at lade brugerne selv tilpasse deres logo så de overholder areal-kravet og den anden er at lade systemet tilpasse den størrelse logoet vises med på websiden. Den sidste løsning er klart at foretrække og er derfor den valgte. Beregningen af logoets nye areal foregår vha. to ligninger med to ubekendte på følgende måde:

Areal: Det pixel areal logoet optager.

h_0**:** Logoets oprindelige pixel højde.

w_0**:** Logoets oprindelige pixel bredde.

h**:** Logoets nye pixel højde.

w**:** Logoets nye pixel bredde.

$$\frac{h_0}{w_0} = \frac{h}{w}$$

$$\frac{hw_0}{h_0} = w$$

$$areal = hw$$
$$areal = \frac{h(hw_0)}{h_0}$$
$$\frac{areal\, h_0}{w_0} = h^2$$
$$h = \sqrt{\frac{areal\, h_0}{w_0}}$$
$$w = \frac{areal}{h}$$

Som beskrevet ovenfor skal brugeren for at kunne stemme eller tilføje et logo have tildelt en adgangskode, og systemet skal kende brugers e-mail. Verificeringen af brugerens e-mail adresse kan forgå på flere niveauer. Ved brugerens indtastning af e-mail adressen kan der opsættes en standard for hvilke tegn der må være i adressen, hvilke der skal være og i hvilken rækkefølge de skal forekomme. Brugerens e-mail adresse kan yderligere verificeres ved at brugerens adgangskode sendes til brugeren vha. den e-mail som systemet har modtaget fra brugeren.

Systemet skal kunne fremstille en adgangskode til brugeren. Sikkerhedsniveauet for adgangskoden kan generelt spænde vidt. I forbindelse med dette system er der valg den samme adgangskode-politik som bruges af Bugzilla-systemet. I Bugzilla-systemet fremstilles adgangskoden vha. en tilfældighedsgenerator og adgangskoden gemmes efterfølgende i krypteret form.

Brugerens indlogning på systemet skal verificeres af systemet ved at sammenligne den af brugeren indtastede e-mail og adgangskode med den af systemet opbevaret e-mail og adgangskode.

I forbindelse med statistik-side skal det for hvert logo vises antallet af stemmer logoet har modtaget og hvor mange stemmer i procent det har modtaget. Der er to måder at vise resultaterne på, sorteret efter antallet af stemmer eller at vise resultaterne i samme rækkefølge som de forskellige logoer vises på indgangssiden, den sidst nævnte metode er valgt her. Beregningen af antallet af stemmer er en simpel sammentælling, og procenttallet beregnes ved simpel procentregning.

(antal stemmer på logo/samlede antal stemmer) * 100

2.4 Grænseflade

Systemet består af seks hovedsider som beskrevet i problemspecifikationen. Det funktionsmæssige design er allerede fastlagt ovenfor og vil derfor ikke blive behandlet nærmere. Der er ikke fremsat krav om brugen af et bestemt type grafisk design, og der er derfor ikke lagt vægt på dette i denne analyse. Når det gælder visningen af de forskellige logoer på indgangssiden, er der lagt vægt på fleksibilitet. Fleksibiliteten tænkes opnået ved at indsætte de forskellige logoer i en tabel hvor det er muligt at ændre antallet af kolonner.

2.5 Hastighed

Der er ikke stillet krav til systemets load tider. Det må dog forventes at systemets indgangsside der indeholder de forskellige konkurrence-logoer vil være den langsomste side. Det kan forventes at de andre sider vil have load tider klart under denne side.

2.6 Sikkerhed

For at have så stor sikkerhed omkring systemet som muligt vil følgende blive prioriteret i forbindelse med udviklingen: Der fastsættes en grænse for størrelsen af logo filer der kan indsættes i systemet, dette gøres for at undgå at systemet bruges til DOS angreb. Kryptering af de adgangskoder der opbevares i databasen; dette gøres for at undgå at brugerens adgangskode kompromitteres i forbindelse med et angreb på databasen.

2.7 Udbygning af system

Der er forskellige muligheder for at udbygge eller generalisere systemet yderligere. Disse muligheder vil blive nævnt her men de vil ikke blive forsøgt implementeret.

Mulighed for at tilføje nye logoer til konkurrencen betyder at brugere som allerede har stemt nu bliver givet nye muligheder. Systemet giver samtidig brugeren mulighed for ændre sin stemme, men medmindre brugeren følger konkurrencen tæt, vil hun ikke have kendskab til de nye valgmuligheder. En løsning på dette problem kan være at sende en e-mail til alle brugere der har afgivet deres stemme. E-mailen skal informere om at et nyt logo er med i konkurrencen. En sådan løsning på problemet kan risikere at irritere brugerne og det foreslås at vælge en udvidelse hvor brugeren, i forbindelse med afgivelsen af sin stemme, eksplicit skal vælge at få tilsendt disse e-mails.

En yderligere mulighed for at udvide systemet er at generalisere systemets funktion til at omfatte afstemninger omkring spørgsmål som CPH STL gruppen ønsker deres brugergruppe hjælp til at afklare. Dette kunne f.eks. være et spørgsmål som, er websiderne struktureret på den rigtige måde? Endelig kunne systemet udvides til at brugeren også fik muligheden for selv at formulere spørgsmål og afholde afstemninger.

3 Design

Systemet skal så vidt det er hensigtsmæssigt og muligt opdeles i et 3-tiere komponentlag. Lagene vil bestå af et datalag, et funktionslag og et grænsefladelag.

3.1 Data

Systemets persistente data gemmes enten som filer eller i en database.

De forskellige logoer gemmes som binære filer et sted som brugerne af sikkerhedsmæssige årsager ikke må kende. Der sættes en øvre grænse for hvor store filer der må gemmes. Alle filer over denne størrelse afvises.

Der findes forskellige systemkonstante, som f.eks. antallet af HTML-tabelkolonner og den maximale størrelse af en logofil. Disse konstanter vil med fordel kunne opbevares i en fil, for at give nem adgang til at konfigurere systemet. Konstanterne kan alternativt opbevares i databasen eller indgå i programteksten.

Alt anden persistent data gemmes i databasen. Data opdeles i tre tabeller indeholdende brugerens information, logoinformation og mulige T-shirt størrelser.

Brugerens informations tabellen (Stamdata) indeholder følgende felter:
StamdataID int(11)
Email varchar(60)
Passwd varchar(20)
LogoID int
TshirtID int
Greeting text
PRIMARY KEY (StamdataID)
UNIQUE Email (Email)
Hvis systemet havde været større og der kunne forudsiges en stor belastning på brugertabellen, kunne Greeting med fordel være pladseret i en separat tabel. Dette ville gavne både adgangstiden og det lagerforbrug som Greeting bruger.

Logoinfomations tabellen (Logo) indeholder følgende felter:
LogoID int(11)
StamdataID int(11)
PixH int
PixW int
ShowHide tinyint
PRIMARY KEY (LogoID)

T-shirt informationstabellen (Tshirt) indeholder følgende felter:
TshirtID int(11)
Size varchar (4)
PRIMARY KEY (TshirtID)
UNIQUE Size (Size)

Der er følgende fremmednøgle relationer mellem tabellerne:
Stamdata.LogoID - Logo.LogoID
Stamdata.TshirtID - Tshirt.TshirtID
Logo.StamdataID - Stamdata.stamdataID

Der skal overføres ikke-persistent data mellem visse af websiderne for at kontrollere om brugeren er logget ind på systemet. Dette kan gøres på to måder enten vha. cookies eller vha. HTML-parameteroverførsel. Parameteroverførselen vha. HTML-forms er valgt for ikke at irritere de brugere der ikke ønsker at bruge

cookies. Samtidig minskes fordelene ved at bruge cookies ved at der kun er et begrænset antal websider i systemet. Der skal yderligere overføres information til loginside om hvor brugen er på vej hen. Denne information overføres også vha. HTML-parametre.

3.2 Funktioner

Følgende funktioner udgør datalaget:
Email_to_StamdataID henter StamdataID i databasen udfra e-mail adresse.
Email_to_Passwd henter Password i databasen udfra e-mail adresse.
Email_to_LogoID_TshirtID_Greeting henter LogoID, TshirtID og Greeting i databasen udfra e-mail adresse.
InsertNewUser indsætter brugerens e-mail og adgangskode i databasen.
InsertNewLogo indsætter logoinformationer i databasen. Samtidig genererer den et navn for logoet.
InsertVote indsætter en brugers afstemningsinformation i databasen.
GetLogoID henter LogoID fra Logo-tabellen.
GetTshirtID_Size henter TshirtID og Size fra Tshirt tabellen.
Email_to_LogoID_TshirtID_Greeting henter LogoID, TshirtID og Greeting i databasen udfra e-mail adresse.
GetLogoID_antalGroupLogoID henter LogoID og det samlede antal LogoIDer grupperet på LogoID fra Stamdata tabellen.
GetAllAntalLogoID henter antallet af LogoIDer fra Stamdata tabellen.
GetLogoIDPixhPixw henter LogoID, PixH og PixW fra Logo-tabellen .

Følgende funktioner udgør funktionlaget:
Procent beregner hvor mange procent af stemmerne et logo har modtaget af det samlede antal stemmer.
NewHeightWidth beregner den højde og bredde som et logo skal vises med.
FileUpload gemmer en uploadet logofil til et sikkert sted.
Følgende funktioner vil i så vid udstrækning som muligt blive lånt fra Bugzilla projektet.
MailPassword skal sende adgangskoden til brugeren.
CheckEmailSyntax checker formatet af brugerens indtastede e-mail.
GenerateRandomPassword generere en tilfældig adgangskode.
Crypt krypterer adgangskoden.
confirm_login checker om brugeren eksisterer i systemet og om hun brugerer den korrekte adgangskode.

4 Programbeskrivelse

Der er brugt følgende værktøj.
Perl
Perl modulerne: DBI 1.20 (database modul) og CGI 2.42 (webgrænseflade modul)
MySQL

Alt programtekst til systemet er kommenteret og det anses derfor ikke nødvendigt at beskrive programmet yderligere.

Det har desværre ikke vist sig muligt eller hensigtmæssigt, at gennemføre en 3-tiere opdeling grundet mangler ved CGI modulet og enkelte fejl i DBI modulet. Af samme grund er flere af funktionerne beskrevet i design-afsnittet ikke implementeret som selvstændige funktioner.

De to 3-tiere lag datalaget og funktionslaget er sammen med konfigurationskonstanterne samlet i en fil. Dette er gjort af rent praktiske grunde og som konsekvens af projektets størrelse.

5 Afprøvning

Der er foretaget en brugerens afprøvning og denne er dokumenteret i bilag vha. en række skærmbilder. Denne afprøvning viste ingen fejl ved systemet.

I forbindelse med brugerens afprøvning er det ikke muligt at afprøve følgende vedrørende systemdata og systemfunktionalitet:

- Svarer de viste logoer til de logoer som i databasen er sat til at skulle vises.
- Registreres brugerens afstemningsdata i databasen.
- Registreres brugerens logodata i databasen.
- Beregnes størrelsen af logoet korrekt.
- Foretages en korrekt generering af adgangskode.
- Fortages et korrekt check af e-mail adressens format.

Alle disse ting er afprøvet særskilt og fundet i orden.

6 Brugervejledning

Der er ikke skrevet nogen slutbrugervejledning da det forventes at brugerne vil kunne bruge systemet uden en vejledning.

7 Efterskrift og konklusion

Den primære målsætning er opfyldt: der eksisterer et fungerende system der efter en endelig godgenkendelse kan tages i brug. Den sekundære målsætning er også opfyldt, der er blevet oparbejdet en erfaring med Perl og Perl moduler i forbindelse med udvikling af websider. På baggrund af den oparbejdede erfaring kan det anbefales at bruge Perl og Perl database-modulet DBI til udvikling af websider som kræver databaseadgang. Derimod viste CGI modulet sig kun brugbart i begrænset omfang og det må anbefales at kikke på andre

webmoduler til fremstillingen af grænsefladekomponenten. Hvis der kan findes en bedre grænsefladekomponent rummer Perl gode muligheder for udviklingen af interaktive websider. Samtidig rummer Perl muligheden for at opbygge et n-tiere komponentlag, noget som ikke er normalt for andre scriptsprog der bruges i forbindelse med udvikling af interaktive websider.

8 Bilag

8.0.1 Program og database

8.0.2 Det færdige Webdesign og brugerens afprøvning

Det færdige Webdesign og brugerens afprøvning dokumenteres vha. de samme skærmbilder.

9 Bilag — Programudskrifter

9.1 logo.cgi

```
#!/usr/bin/perl -w

#logo.cgi

use diagnostics;
use strict;

use CGI;
use CGI::Carp qw(fatalsToBrowser);

use DBI;

require 'Logo_globals.pl';

# Skaber forbindelse med databasen, OBS login og password skal aendres
my $dbh = DBI->connect(  'DBI:mysql:logobase',
                         'root',
                         '',
                      ) || die "Database connection not made: $DBI::errstr";

# Henter LogoID, PixH og PixW fra Logo tabellen
my $table_data = $dbh->prepare("select LogoID, PixH, PixW from Logo where ShowHide=1");
$table_data->execute();
my( $LogoID,  $PixH, $PixW);
$table_data->bind_columns( undef, \$LogoID,  \$PixH, \$PixW);


my $queryCGI = new CGI;

# Webside genereres
print $queryCGI->header, $queryCGI->start_html('Logo Voting');

print $queryCGI->h2('Logo Voting');

# Opsaetning af tabel indeholdende billed af T-shirt
print $queryCGI->table({-border=>1,-cellspacing=>1,-cellpadding=>1,-WIDTH=>"100%",-HEIGHT=>"1%"},
        $queryCGI->Tr({-valign=>"TOP"},
        [
         $queryCGI->td([($queryCGI->h3('Vote for a logo and you can vind a T-shirt with this logo.')),
                       ($queryCGI->img({src=>'../perl/logotshirt',  align=>'LEFT'}))]),
        ]
       )
);

print $queryCGI->br;

# Generere tabel med billeder
print "<table BORDER=0 CELLSPACING=10 CELLPADDING=0 COLS=4 WIDTH=\"100%\" HEIGHT=\"30%\"> \n";
print "<tr> \n";
my $i=0;
my $numberofcols = Param('numberofcols');
my $NewPixH;
my $NewPixW;
my $LogoPatch = Param('logofilepatch');
my $Logoname = Param('lname');
while(  $table_data->fetch() ) {
  print "<td>";
  ($NewPixH, $NewPixW)  = NewHeightWidth($PixH, $PixW);
  print "<img SRC=\"$LogoPatch$Logoname$LogoID\" height=$NewPixH width=$NewPixW> logo $LogoID";
  print "</td> \n";
  $i++;
  if (($i % $numberofcols) == 0) {print"</tr><tr> \n";}
}
print "</tr> \n";
print "</table> \n";

print $queryCGI->br;
```

9.2 logo.cgi

```
print $queryCGI->br;
print $queryCGI->br;
print $queryCGI->br;

# Tabel indeholdende links til de andre hovedsider
print $queryCGI->table({-border=>0,-cellspacing=>1,-cellpadding=>1,-WIDTH=>"100%",-HEIGHT=>"1%"},
        $queryCGI->Tr({-valign=>"TOP"},
        [
         $queryCGI->td([($queryCGI->a({href=>"createlogoaccount.cgi"},"Get password to vote")),
                       ($queryCGI->a({href=>"login.cgi?page=uploadlogo.cgi"},"Add a new proposal"))]),
         $queryCGI->Tr(),
         $queryCGI->td([($queryCGI->a({href=>"login.cgi?page=formvote.cgi"},"Give your Vote")),
                       ($queryCGI->a({href=>"stat.cgi"},"View status"))])
        ]
      )
);

print $queryCGI->end_html;

$table_data->finish();
# Forbindelse til database lukkes
$dbh->disconnect();
```

9.3 createlogoaccount.cgi

```
#!/usr/bin/perl -w

#cretelogoaccount.cgi

use diagnostics;
use strict;

use CGI;
use CGI::Carp qw(fatalsToBrowser);
use DBI;

require 'Logo_globals.pl';

# Skaber forbindelse med databasen, OBS login og password skal aendres
my $dbh = DBI->connect( 'DBI:mysql:logobase',
                        'root',
                        '',
                      ) || die "Database connection not made: $DBI::errstr";

my $queryCGI = new CGI;

# Overfoerte parameter fra kaldside login
my $emailadr = $queryCGI->param('emailadr');

if (defined $emailadr) {
    # Email adressens format checkes
    CheckEmailSyntax($emailadr);
    # Det checkes om brugeren har konto
    if (DBname_to_id($emailadr) != 0) {
        # Webside med fejlmeedelse genereres
        print $queryCGI->header, $queryCGI->start_html('Account Exists');
        print "A Logo account for ";
        print $queryCGI->tt($emailadr);
        print qq|  already exists.
                |;
        print $queryCGI->end_html;
        exit;
    }
    # Den nye bruger indsaettes i databasen og password returneres
    my $password = InsertNewUser($emailadr);
    # Password sendes til brugeren
    MailPassword($emailadr, $password);
    # Webside genereres
    print $queryCGI->header, $queryCGI->start_html('Account created');
    print "Your password was sent to the folling e-mail adress ".$emailadr;
    print $queryCGI->end_html;
    exit;
}

# Her genereres den del af side som brugeren foerst moeder
print $queryCGI->header, $queryCGI->start_html('Create a new logo account');
print q{
To create a logo account, all that you need to do is to enter a
legitimate e-mail address.  The account will be created, and its
password will be mailed to you.
};

# Html form saettes op
my $method = "post";
my $action;
my $encoding;
print $queryCGI->startform($method,$action,$encoding);
print $queryCGI->table({-border=>0,-cellspacing=>1,-cellpadding=>1},
                       $queryCGI->Tr({-align=>"RIGHT",-valign=>"TOP"},
                                     $queryCGI->td([$queryCGI->strong('E-mail address:')]
                                                   ),
                                     $queryCGI->td([$queryCGI->textfield(-name=>'emailadr',
```

9.4 createlogoaccount.cgi

```
                                                                    -default=>'',
                                                                    -size=>35,
                                                                    -maxlength=>50)]
                                                    )
                                        )
                                );
print  $queryCGI->td([($queryCGI->submit(-name=>'btSubmit', -value=>'Create Account'))]);
print $queryCGI->endform;
print $queryCGI->end_html;
# Forbindelse til database lukkes
$dbh->disconnect();
```

9.5 login.cgi

```
#!/usr/bin/perl -w

#login.cgi

use diagnostics;
use strict;

use CGI;
use CGI::Carp qw(fatalsToBrowser);

my $queryCGI = new CGI;

# Webside genereres den indeholder et felt til indskrivning af e-mail adreesen
# og et felt til indskrivning af password
print $queryCGI->header, $queryCGI->start_html('Logo Voting');

print $queryCGI->h2('Login');

print $queryCGI->br;
print $queryCGI->br;

# Overfoerte parameter fra kaldside indeholdende tagetside
my $page = $queryCGI->param('page');

# Html form saettes op
my $method = "post";
my $action = $page;
my $encoding;

print $queryCGI->startform($method,$action,$encoding);

print $queryCGI->h5('Write your email adresse here.');
print $queryCGI->textfield(-name=>'emailadr',
                           -default=>'',
                           -size=>30,
                           -maxlength=>50);

print $queryCGI->br;
print $queryCGI->br;

print $queryCGI->h5('Write your password here.');
print $queryCGI->password_field(-name=>'passwd',
                          -value=>'',
                          -size=>30,
                          -maxlength=>50);

print $queryCGI->br;
print $queryCGI->br;

print $queryCGI->table({-border=>0,-cellspacing=>10,-cellpadding=>10},
                       $queryCGI->Tr({-align=>"CENTER",-valign=>"TOP"},
                                     [
                                      $queryCGI->td([($queryCGI->submit(-name=>'btSubmit',
                                                                        -value=>'Submit')),
                                                     ($queryCGI->reset(-value=>'Reset'))])
                                     ]
                                     )
                       );


print $queryCGI->endform;

print $queryCGI->end_html;
```

9.6 formvote.cgi

```
#!/usr/bin/perl -w

#formvote.cgi

use diagnostics;
use strict;

use CGI;
use CGI::Carp qw(fatalsToBrowser);

use DBI;

require 'Logo_globals.pl';

# Skaber forbindelse med databasen, OBS login og password skal aendres
my $dbh = DBI->connect( 'DBI:mysql:logobase',
                        'root',
                        '',
                      ) || die "Database connection not made: $DBI::errstr";

# Henter LogoID fra Logo tabellen
my $table_datalogo = $dbh->prepare("select LogoID from Logo where ShowHide=1");
$table_datalogo->execute();
my( $LogoID);
$table_datalogo->bind_columns( undef, \$LogoID);

# Generer et array af hash tabeller indeholdende LogoID og et navn bestaaende af teksten Logo og LogoID
my $txtLogo = "Logo ";
my %proposels;                          #hash table
my @LogoIDs = ();                       #toemmer array da dette ikke altid goeres aotomatisk
while( $table_datalogo->fetch() ) {
    $proposels{$LogoID} = $txtLogo.$LogoID;
    push(@LogoIDs,$LogoID);
}
$table_datalogo->finish();

# Henter TshirtID og Size fra Tshirt tabellen.
my $table_datatshirt = $dbh->prepare("select TshirtID, Size from Tshirt");
$table_datatshirt->execute();
my( $TshirtID, $Size);
$table_datatshirt->bind_columns( undef, \$TshirtID, \$Size);

# Generer et array af hash tabeller indeholdende TshirtID og Size
my %Tshirts;                            #hash table
my @TshirtIDs = ();                     #toemmer array da dette ikke altid goeres aotomatisk
while( $table_datatshirt->fetch() ) {
    $Tshirts{$TshirtID} = $Size;
    push(@TshirtIDs,$TshirtID);
}
$table_datatshirt->finish();

my $queryCGI = new CGI;

# Overfoerte parameter fra kaldside login og password
my $emailadr = $queryCGI->param('emailadr');
my $passwd = $queryCGI->param('passwd');

# Henter tidligere afstemnings infomation hvis den eksistere
my $old_LogoID;
my $old_TshirtID;
my $old_greeting;
($old_LogoID, $old_TshirtID, $old_greeting) = Email_to_LogoID_TshirtID_Greeting($emailadr);

if (defined $emailadr) {
    # Login og password checkes
    confirm_login($emailadr, $passwd);
    # Webside genereres
```

9.7 formvote.cgi

```
    print $queryCGI->header, $queryCGI->start_html('Logo Voting');
    print $queryCGI->h2('Logo Voting');
    print $queryCGI->br;
    print $queryCGI->br;
    # Html form saettes op
    my $method = "post";
    my $action = "addvote.cgi";
    my $encoding;
    print $queryCGI->startform($method,$action,$encoding);
    print $queryCGI->hidden(-name=>'emailadr',
                            -default=>[$emailadr]);
    print $queryCGI->hidden(-name=>'passwd',
                            -default=>[$passwd]);
    # Html tabel genereres og popup menuer fyldes med array LogoIDs og array TshirtIDs
    print $queryCGI->table({-border=>0,-cellspacing=>10,-cellpadding=>10},
                           $queryCGI->Tr({-align=>"CENTER",-valign=>"TOP"},
                                         [
                                          $queryCGI->td(['My vote goes to:',
                                                         ($queryCGI->popup_menu(-name=>'proposel',
                                                                                -values=>[@LogoIDs],
                                                                                -labels=>{ %proposels },
                                                                                -default=>$old_LogoID
                                                                                ))])

                                          ]
                                         )
                           );
    print $queryCGI->br;
    print $queryCGI->table({-border=>0,-cellspacing=>10,-cellpadding=>10},
                           $queryCGI->Tr({-align=>"CENTER",-valign=>"TOP"},
                                         [
                                          $queryCGI->td(['I use t-shirt size:',
                                                         ($queryCGI->popup_menu(-name=>'tshirt',
                                                                                -values=>[@TshirtIDs],
                                                                                -labels=>{ %Tshirts },
                                                                                -default=>$old_TshirtID
                                                                                ))])
                                          ]
                                         )
                           );

    print $queryCGI->br;
    print $queryCGI->br;
    print $queryCGI->h5('Additional greetings to the development team.');
    print $queryCGI->textarea(-name=>'greeting',
                              -rows=>10,
                              -default=>$old_greeting,
                              -columns=>50);
    print $queryCGI->table({-border=>0,-cellspacing=>10,-cellpadding=>10},
                           $queryCGI->Tr({-align=>"CENTER",-valign=>"TOP"},
                                         [
                                          $queryCGI->td([($queryCGI->submit(-name=>'btSubmit',
                                                                            -value=>'Submit')),
                                                         ($queryCGI->reset(-value=>'Reset'))])
                                          ]
                                         )
                           );

    print $queryCGI->endform;
    print $queryCGI->end_html;
}

if (!defined $emailadr) {
    print $queryCGI->header, $queryCGI->start_html('Logo Voting');

    print $queryCGI->h5('You have to login!');

    print $queryCGI->end_html;
```

9.8 addvote.cgi

```
#!/usr/bin/perl -w

#addvote.cgi

use diagnostics;
use strict;

use CGI;
use CGI::Carp qw(fatalsToBrowser);
use DBI;

require 'Logo_globals.pl';


# Skaber forbindelse med databasen, OBS login og password skal aendres
my $dbh = DBI->connect( 'DBI:mysql:logobase',
                        'root',
                        '',
                      ) || die "Database connection not made: $DBI::errstr";

my $queryCGI = new CGI;

# Overfoerte parameter fra kaldside login og password
my $passwd = $queryCGI->param('passwd');
my $emailadr = $queryCGI->param('emailadr');

if (defined $emailadr) {
    # Login og password checkes
    confirm_login($emailadr, $passwd);
    # Overfoerte parameter logovalg, T-shirt stoerrelse og hilsen
    my $proposel = $queryCGI->param('proposel');
    my $tshirt = $queryCGI->param('tshirt');
    my $greeting = $queryCGI->param('greeting');
    # Brugerens logovalg, T-shirt stoerrelse og hilsen indsaettes i databasen
    InsertVote($emailadr, $proposel, $tshirt, $greeting);

    # Webside med tak genereres
    print $queryCGI->header, $queryCGI->start_html('Logo Voting');
    print $queryCGI->h2('Thank you for voting!');
    print $queryCGI->br;
    print $queryCGI->h3('Remember you can alwas change your vote.');
    print $queryCGI->end_html;
}
else {
    # Webside med fejlmedelse genereres
    print $queryCGI->header, $queryCGI->start_html('Logo Voting');
    print $queryCGI->h5('You have to login!');
    print $queryCGI->end_html;
}
# Forbindelse til database lukkes
    $dbh->disconnect();
```

9.9 uploadlogo.cgi

```
#!/usr/bin/perl -w

#uploadlogo.cgi

use diagnostics;
use strict;

use CGI;
use CGI::Carp qw(fatalsToBrowser);

my $queryCGI = new CGI;

# Overfoerte parameter fra kaldside login og password
my $emailadr = $queryCGI->param('emailadr');
my $passwd = $queryCGI->param('passwd');

# Webside genereres den bestaa af et upload felt og to tekst felter
print $queryCGI->header, $queryCGI->start_html('Logo Voting');

print $queryCGI->h2('Logo Voting');

print $queryCGI->br;
print $queryCGI->br;

# Html form saettes op
my $method = "post";
my $action = "addlogo.cgi";
my $encoding;

print $queryCGI->start_multipart_form($method,$action,$encoding);

print $queryCGI->hidden(-name=>'emailadr',
                        -default=>[$emailadr]);

print $queryCGI->hidden(-name=>'passwd',
                        -default=>[$passwd]);

print $queryCGI->h4('Filename');
print $queryCGI->filefield(-name=>'uploaded_file',
                           -default=>'starting value',
                           -size=>30,
                           -maxlength=>80);

print $queryCGI->br;
print $queryCGI->br;

print $queryCGI->h5('Write your logos pixel height here.');
print $queryCGI->textfield(-name=>'pixh',
                           -default=>'',
                           -size=>10,
                           -maxlength=>50);

print $queryCGI->br;
print $queryCGI->br;

print $queryCGI->h5('Write your logos pixel width here.');
print $queryCGI->textfield(-name=>'pixw',
                           -default=>'',
                           -size=>10,
                           -maxlength=>50);

print $queryCGI->br;
print $queryCGI->br;

print $queryCGI->table({-border=>0,-cellspacing=>10,-cellpadding=>10},
                       $queryCGI->Tr({-align=>"CENTER",-valign=>"TOP"},
                                     [
```

9.10 uploadlogo.cgi

```
                                        $queryCGI->td([($queryCGI->submit(-name=>'btSubmit',
                                                                          -value=>'Submit')),
                                                       ($queryCGI->reset(-value=>'Reset'))])
                                    ]
                                   )
                        );

print $queryCGI->endform;
print $queryCGI->end_html;
```

9.11 addlogo.cgi

```
#!/usr/bin/perl -w

#addlogo.cgi

use diagnostics;
use strict;

use CGI;
use CGI::Carp qw(fatalsToBrowser);
use DBI;

require 'Logo_globals.pl';

# Skaber forbindelse med databasen, OBS login og password skal aendres
my $dbh = DBI->connect( 'DBI:mysql:logobase',
                        'root',
                        '',
                      ) || die "Database connection not made: $DBI::errstr";

# Det maximale antal byte saettes for at undgaa DOS angreb gennem upload funktionen
my $max_upload_byte = Param('max_upload_byte');
$CGI::POST_MAX=1024 * $max_upload_byte;

my $queryCGI = new CGI;

# Overfoerte parameter fra kaldside login og password
my $emailadr = $queryCGI->param('emailadr');
my $passwd = $queryCGI->param('passwd');

if (defined $emailadr) {
    # Login og password checkes
    confirm_login($emailadr, $passwd);
    # Overfoerte parameter upload filen, dens hoejde og bredde
    my $filename = $queryCGI->param('uploaded_file');
    my $pixh = $queryCGI->param('pixh');
    my $pixw = $queryCGI->param('pixw');
    # Logo parameter indsaettes i database og logonavn returneres
    my $LogoID = InsertNewLogo ($emailadr, $pixh, $pixw);

    # Den binare logo fil kopieres til et sikkert sted, OBS sted skal aendres.
    FileUpload($filename, $LogoID);

    # Webside med tak genereres
    print $queryCGI->header, $queryCGI->start_html('Logo Voting');
    print $queryCGI->h2('Thank you for your logo!');
    print $queryCGI->end_html;

}
else {
    # Webside med fejlmedelse genereres
    print $queryCGI->header, $queryCGI->start_html('Logo Voting');
    print $queryCGI->h5('You have to login!');
    print $queryCGI->end_html;
}
# Forbindelse til database lukkes
$dbh->disconnect();
```

9.12 stat.cgi

```
#!/usr/bin/perl -w

#stat.cgi

use diagnostics;
use strict;

use CGI;
use CGI::Carp qw(fatalsToBrowser);

use DBI;

require 'Logo_globals.pl';

# Skaber forbindelse med databasen, OBS login og password skal aendres
my $dbh = DBI->connect( 'DBI:mysql:logobase',
                        'root',
                        ,
                      ) || die "Database connection not made: $DBI::errstr";

# Henter LogoID og det sammlede antal LogoIDer grupperet paa LogoID fra Stamdata tabellen
my $table_data = $dbh->prepare("select LogoID, count(LogoID) as Votes from Stamdata where LogoID is not null group by LogoID;");
$table_data->execute();
my( $LogoID, $Votes);
$table_data->bind_columns( undef, \$LogoID, \$Votes);

# Henter sammlede antal LogoIDer fra Stamdata tabellen
my $sum = $dbh->selectrow_array("select count(LogoID) from Stamdata");

my $queryCGI = new CGI;

print $queryCGI->header, $queryCGI->start_html('Logo Voting');

# Saetter tabel header og generere raekker
my @headings = ('Logo','Number of votes','Number of votes in %');
my @rows = $queryCGI->th(\@headings);
while( $table_data->fetch() ) {
   push(@rows, $queryCGI->td([$queryCGI->strong($LogoID), $Votes, Procent($Votes, $sum)]));
}

# Opbygger tabel
print $queryCGI->table({-border=>undef},
        $queryCGI->caption($queryCGI->strong('Logo Stats')),
        $queryCGI->Tr({-align=>"CENTER",-valign=>"TOP"}, \@rows)
        );

print Param('version');
print $queryCGI->end_html;

$table_data->finish();
# Forbindelse til database lukkes
$dbh->disconnect();
```

9.13 Logo_globals.pl

```
use diagnostics;
use strict;

use CGI;
use CGI::Carp qw(fatalsToBrowser);

use DBI;

# Alle funktioner laant fra Bugzilla systemet er copyright beskyttet under the Mozilla Public
# License Version 1.1
# Tak for laan til foelgende:
# Terry Weissman <terry@mozilla.org>
# Dan Mosedale <dmose@mozilla.org>
# Joe Robins <jmrobins@tgix.com>
# Dave Miller <justdave@syndicomm.com>
# Christopher Aillon <christopher@aillon.com>
# Jake <jake@acutex.net>

# System konstanter.
# Det maximale antal byte saettes for at undgaa DOS angreb gennem upload funktionen.
$::param{'$max_upload_byte'} = 1000;
# Navnet paa stigen hvor logo filerne opbevaeres, OBS sted skal aendres.
$::param{'logofilepatch'} =  "../perl/";
# Prefix til logonavn
$::param{'lname'} = "logo";
# Det areal som alle logoer tildeles.
$::param{'MaxPicSize'} = 5000;
# Antalet af koloner i html-tabellen.
$::param{'numberofcols'} = 3;


my $mailtemplate=q{From: Logo-daemon
To: %mailaddress%
Subject: Your Logo password.

To vote for a logo, you most use the following:

 E-mail address: %login%
       Password: %password%

};


# Skaber forbindelse med databasen, OBS login og password skal aendres
my $dbh = DBI->connect( 'DBI:mysql:logobase',
                        'root',
                        '',
                      ) || die "Database connection not made: $DBI::errstr";



my $queryCGI = new CGI;

# Funktion laant fra Bugzilla systemet, og derefter let modificeret
# Haantere system konstanter.
sub Param ($) {
    my ($value) = (@_);
    if (defined $::param{$value}) {
        return $::param{$value};
    }
}

# Funktion laant fra Bugzilla systemet, og derefter let modificeret
# Checker formatet paa e-mial adresser
sub CheckEmailSyntax {
    my ($addr) = (@_);
    my $match = q:^[^@]*@[^@]*\\.[^@]*$:;
```

9.14 Logo_globals.pl

```
    if ($addr !~ /$match/ || $addr =~ /[\\\(\)<>&,;:"\[\] \t\r\n]/) {

        # For security, escape HTML special characters.
        $addr = $queryCGI->escapeHTML($addr);
        print $queryCGI->header, $queryCGI->start_html('Check e-mail syntax');
        print "The e-mail address you entered\n";
        print $queryCGI->b($addr);
        print " didn't match our minimal\n";
        print "syntax checking for a legal email address.\n";
        print "q:^[^@]*@[^@]*\\.[^@]*$:"  .  "\n";
        print "It must also not contain any of these special characters: ";
        print $queryCGI->tt('\\ ( ) &amp; &lt; &gt; , ; : \" [ ]');
        print "or any whitespace.\n";
        print "<p>Please click <b>Back</b> and try again.\n";
        print $queryCGI->end_html;
        exit;
    }
}

# Funktion laant fra Bugzilla systemet
# Kryptere password
sub Crypt {
    # Crypts a password, generating a random salt to do it.
    # Random salts are generated because the alternative is usually
    # to use the first two characters of the password itself, and since
    # the salt appears in plaintext at the beginning of the crypted
    # password string this has the effect of revealing the first two
    # characters of the password to anyone who views the crypted version.

    my ($password) = @_;

    # The list of characters that can appear in a salt.  Salts and hashes
    # are both encoded as a sequence of characters from a set containing
    # 64 characters, each one of which represents 6 bits of the salt/hash.
    # The encoding is similar to BASE64, the difference being that the
    # BASE64 plus sign (+) is replaced with a forward slash (/).
    my @saltchars = (0..9, 'A'..'Z', 'a'..'z', '.', '/');

    # Generate the salt.  We use an 8 character (48 bit) salt for maximum
    # security on systems whose crypt uses MD5.  Systems with older
    # versions of crypt will just use the first two characters of the salt.
    my $salt = '';
    for ( my $i=0 ; $i < 8 ; ++$i ) {
        $salt .= $saltchars[rand(64)];
    }

    # Crypt the password.
    my $cryptedpassword = crypt($password, $salt);

    # Return the crypted password.
    return $cryptedpassword;
}

# Generere et tilfaeldigt password
sub GenerateRandomPassword {
    my ($size) = @_;

    # Generated passwords are eight characters long by default.
    $size ||= 8;

    # The list of characters that can appear in a randomly generated password.
    # Note that users can put any character into a password they choose themselves.
    my @pwchars = (0..9, 'A'..'Z', 'a'..'z', '-', '_', '!', '@', '#', '$', '%', '^', '&', '*');

    # The number of characters in the list.
    my $pwcharslen = scalar(@pwchars);
```

9.15 Logo_globals.pl

```
    # Generate the password.
    my $password = "";
    for ( my $i=0 ; $i<$size ; $i++ ) {
        $password .= $pwchars[rand($pwcharslen)];
    }

    # Return the password.
    return $password;
}

# Funktion laant fra Bugzilla systemet
sub PerformSubsts {
    my ($str, $substs) = (@_);
    $str =~ s/%([a-z]*)%/(defined $substs->{$1} ? $substs->{$1} : Param($1))/eg;
    return $str;
}

# Funktion laant fra Bugzilla systemet, og derefter let modificeret
# Sender password til brugeren
sub MailPassword {
    my ($login, $password) = (@_);
    my $msg = PerformSubsts($mailtemplate,
                            {"mailaddress" => $login,
                             "login" => $login,
                             "password" => $password});

    open SENDMAIL, "|/usr/lib/sendmail -t";
    print SENDMAIL $msg;
    close SENDMAIL;
}


# Funktion laant fra Bugzilla systemet
# Use detaint_string() when you know that there is no way that the data
# in a scalar can be tainted, but taint mode still bails on it.
# WARNING!! Using this routine on data that really could be tainted
#           defeats the purpose of taint mode.  It should only be
#           used on variables that cannot be touched by users.
sub detaint_string {
    my ($str) = @_;
    $str =~ m/^(.*)$/s;
    $str = $1;
}

# Funktion laant fra Bugzilla systemet
# This routine is largely copied from Mysql.pm.
sub SqlQuote {
    my ($str) = (@_);
#     if (!defined $str) {
#         confess("Undefined passed to SqlQuote");
#     }
    $str =~ s/([\\\'])/\\$1/g;
    $str =~ s/\0/\\0/g;
    # If it's been SqlQuote()ed, then it's safe, so we tell -T that.
    $str = detaint_string($str);
    return "'$str'";
}

# Funktion laant fra Bugzilla systemet, og derefter let modificeret
# Henter StamdataID i databasen udfra e-mail adresse
sub DBname_to_id {
    my ($name) = (@_);
    my $sql = "select StamdataID from Stamdata where Email = @{[SqlQuote($name)]}";
    my $r = $dbh->selectrow_array($sql);
    if (defined $r && $r =~ m/^([1-9][0-9]*)$/) {
        return $1;
    } else {
```

9.16 Logo_globals.pl

```
        return 0;
    }
}

# Henter Password i databasen udfra e-mail adresse
sub Email_to_Passwd {
    my ($name) = (@_);
    my $sql = "select Email, Passwd from Stamdata where Email = @{[SqlQuote($name)]}";
    return my @row = $dbh->selectrow_array($sql);
}

# Henter LogoID, TshirtID og Greeting i databasen udfra e-mail adresse
sub Email_to_LogoID_TshirtID_Greeting {
    my ($name) = (@_);
    my $sql = "select LogoID, TshirtID,Greeting from Stamdata where Email = @{[SqlQuote($name)]}";
    return my @row = $dbh->selectrow_array($sql);
}

# Indsaette en ny bruger i databasen
sub InsertNewUser {
    my ($username) = (@_);

    # Generere nyt password til brugeren
    my $password = GenerateRandomPassword();
    my $cryptpassword = Crypt($password);

    # Indsaetter brugeren i databasen
    $username = SqlQuote($username);
    $cryptpassword = SqlQuote($cryptpassword);
    my $table_data = $dbh->prepare("insert into Stamdata (Email, passwd)
      VALUES ($username, $cryptpassword)");
    $table_data->execute();

    # Returner passwordet til den kaldende funktion
    return $password;
}

# Indsaetter et nyt logo i databasen
sub InsertNewLogo {
    my ($username, $pixh, $pixw) = (@_);

    my $StamdataID = DBname_to_id($username);

    # Laaser tabel Logo for read and write det er ikke muligt kun at laase write
    my $lock_data = $dbh->prepare("lock tables Logo write;");
    $lock_data->execute();

    # Indsaetter data i Logo tabellen
    my $table_data = $dbh->prepare("insert into Logo (StamdataID, PixH, PixW)
    VALUES ($StamdataID, $pixh, $pixw)");
    $table_data->execute();
    my $sql =  "select max(LogoID) from Logo";
    my $r = $dbh->selectrow_array($sql);

    # Aabner tabel Logo
    my $unlock_data = $dbh->prepare("unlock tables");
    $unlock_data->execute();

    # Hvis der ikke er en ID at returnere skal der skabes et unikt navn
    # her er valg brugerens e-mail adresse kombineret med system tiden
    if (defined $r && $r =~ m/^([1-9][0-9]*)$/) {
        return $1;
    } else {
        my $time = time();
        return $username.$time;
    }
}
```

9.17 Logo_globals.pl

```
# Indsaetter en brugers afstemnings information i databasen
sub InsertVote {
    my ($username, $proposel, $tshirt, $greeting) = (@_);

    # Henter StamdataID
    my $StamdataID = DBname_to_id($username);

   # Indsaetter brugerens afstemnings information i databasen
    $greeting = SqlQuote($greeting);
    my $table_data = $dbh->prepare("update  Stamdata set LogoID=$proposel where StamdataID=$StamdataID");
    $table_data->execute();
    my $table_data = $dbh->prepare("update  Stamdata set TshirtID=$tshirt where StamdataID=$StamdataID");
    $table_data->execute();
    my $table_data = $dbh->prepare("update  Stamdata set Greeting=$greeting where StamdataID=$StamdataID");
    $table_data->execute();
}

# Beregner den hoejde og bredde som et logo skal vises med.
sub  NewHeightWidth {
     my ($PixH, $PixW) = (@_);
     my $MaxPicSize = Param('MaxPicSize');
     my $NewPixH;
     my $NewPixW;
     my @NewPix;
     $NewPixH =  int(sqrt(($MaxPicSize*$PixH)/$PixW));
     push (@NewPix, $NewPixH);
     $NewPixW = int($MaxPicSize/$NewPixH);
     push (@NewPix, $NewPixW);
     return @NewPix;
}

# Beregner hvor mange procent af stemmerne et logo har modtaget af det samlede antal stemmer.
sub Procent {
     my ($NrOneLogo, $NrAllLogo) = (@_);
     my $LogoProcent;
     $LogoProcent = int(($NrOneLogo/$NrAllLogo*100)+0.5);
     return $LogoProcent;
}

# Den binare logo fil kopieres til et sikkert sted, OBS sted skal aendres
sub FileUpload {
    my ($filename, $LogoID) = (@_);
    my $filepatch = Param('logofilepatch');
    my $lname =  Param('lname');
    my $name = $lname.$LogoID;
    open (OUTFILE,">>".$filepatch.$name);
    while (my $bytesread=read($filename,my $buffer,1024)) {
        print OUTFILE $buffer;
    }
    close $filename;


}

# Funktion laant fra Bugzilla systemet, og derefter let modificeret
# Viser fejlmedelse
sub DisplayError {
  my ($message, $title) = (@_);
  $title ||= "Error";

  print $queryCGI->header, $queryCGI->start_html($title);
  print  $message;
  print $queryCGI->end_html;

  return 1;
}
```

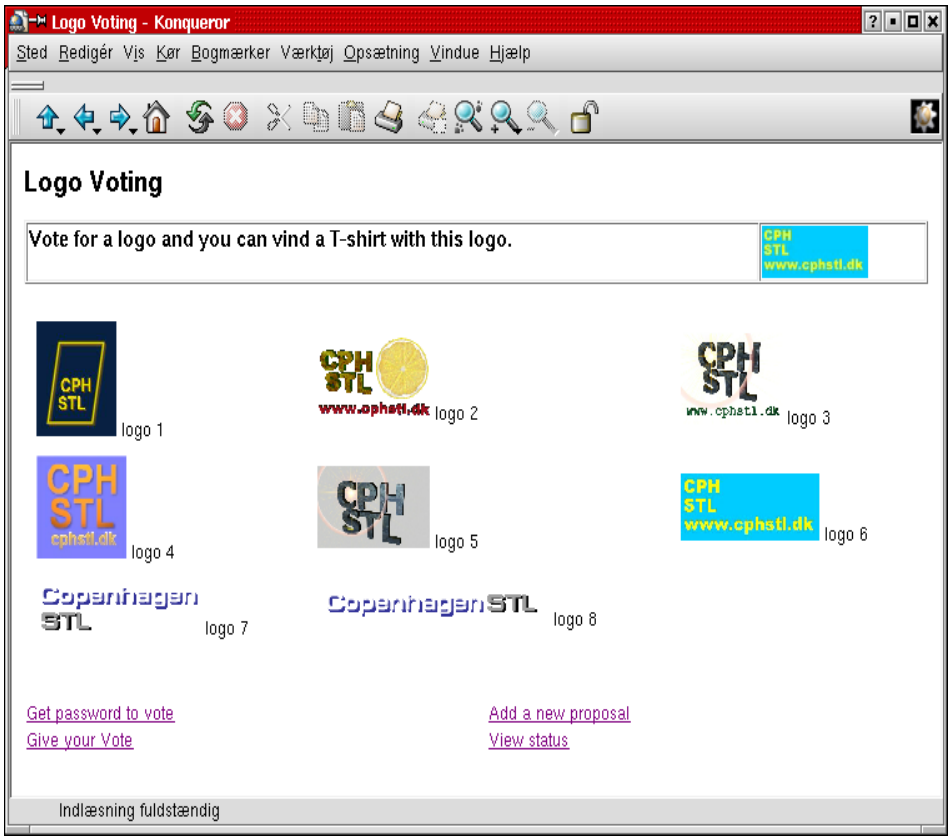
9.18 Logo_globals.pl

```
# Funktion laant fra Bugzilla systemet, og derefter let modificeret
# Sammenligner de af brugeren indtastet login informationer med bruger informationerne i databasen
sub confirm_login {
  my ($emailadr, $passwd) = (@_);

    if (defined $emailadr && defined $passwd) {
        # Tester af brugerens e-mail er en valid email addresse
        CheckEmailSyntax($emailadr);

        # Henter brugerens StamdataID og krypteret password fra database.
        my $userid;
        my $realcryptpwd;
        ($userid, $realcryptpwd) = Email_to_Passwd($emailadr);

        # Check at brugeren eksistere hvis ikke send fejlmeddelse
        $userid
          || DisplayError("The username or password you entered is not valid.")
          && exit;

        # Valider brugeren
        # Hent salt fra brugerens kryptede password.
        my $salt = $realcryptpwd;

        # Brug saltet til at kryptere det password som brugeren indtastede
        my $enteredCryptedPassword = crypt( $passwd , $salt );

        # Sammenlig de to password og send fejlmeddelse hvis de er forskellige
        ($enteredCryptedPassword eq $realcryptpwd)
            || DisplayError("The username or password you entered is not valid.")
            && exit;
    }
}
```

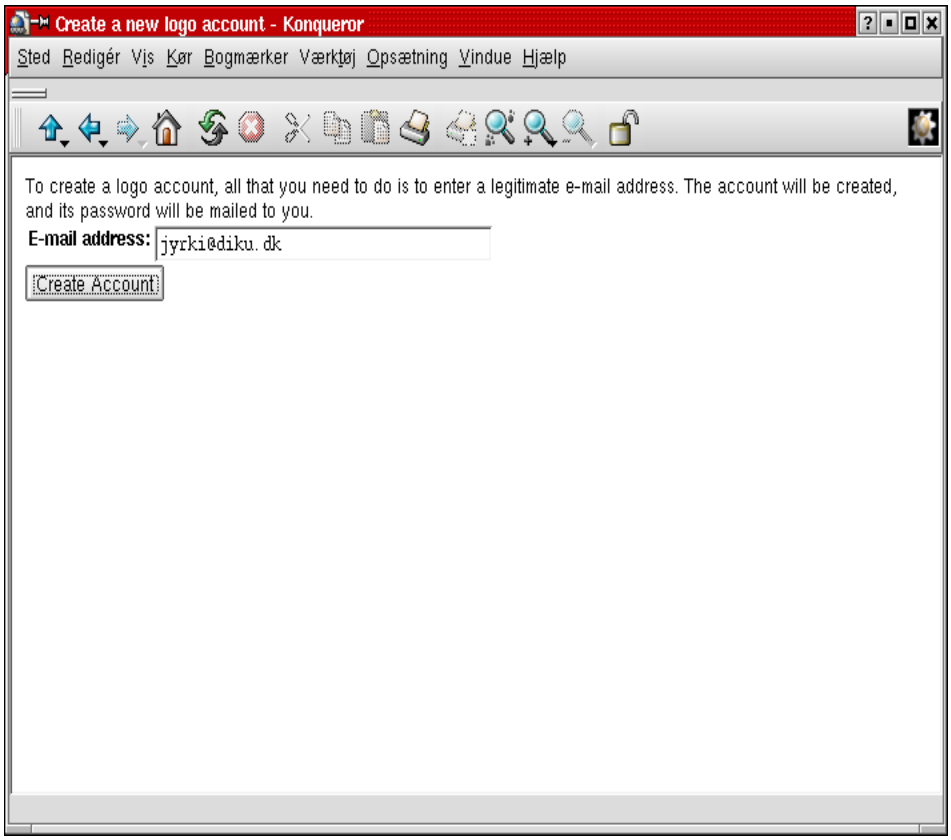
9.19 logodatabase

```
create database logobase;
use logobase;

drop table if exists Stamdata;
create table Stamdata (
  StamdataID int(11) unsigned DEFAULT '0' NOT NULL auto_increment,
  Email varchar(60) DEFAULT '0' NOT NULL,
  Passwd varchar(20),
  LogoID int,
  TshirtID int,
  Greeting text,
  PRIMARY KEY (StamdataID),
  UNIQUE Email (Email)
  );

drop table if exists Tshirt;
create table Tshirt (
  TshirtID int(11) unsigned DEFAULT '0' NOT NULL auto_increment,
  Size varchar (4),
  PRIMARY KEY (TshirtID),
  UNIQUE Size (Size)
  );

insert into Tshirt values (',','xxxs');
insert into Tshirt values (',','xxs');
insert into Tshirt values (',','xs');
insert into Tshirt values (',','s');
insert into Tshirt values (',','m');
insert into Tshirt values (',','l');
insert into Tshirt values (',','xl');
insert into Tshirt values (',','xxl');
insert into Tshirt values (',','xxxl');

drop table if exists Logo;
create table Logo (
  LogoID int(11) unsigned DEFAULT '0' NOT NULL auto_increment,
  StamdataID int(11) unsigned DEFAULT '0' NOT NULL,
  PixH int,
  PixW int,
  Showhide tinyint unsigned DEFAULT '0' NOT NULL,
  PRIMARY KEY (LogoID)
  );
```

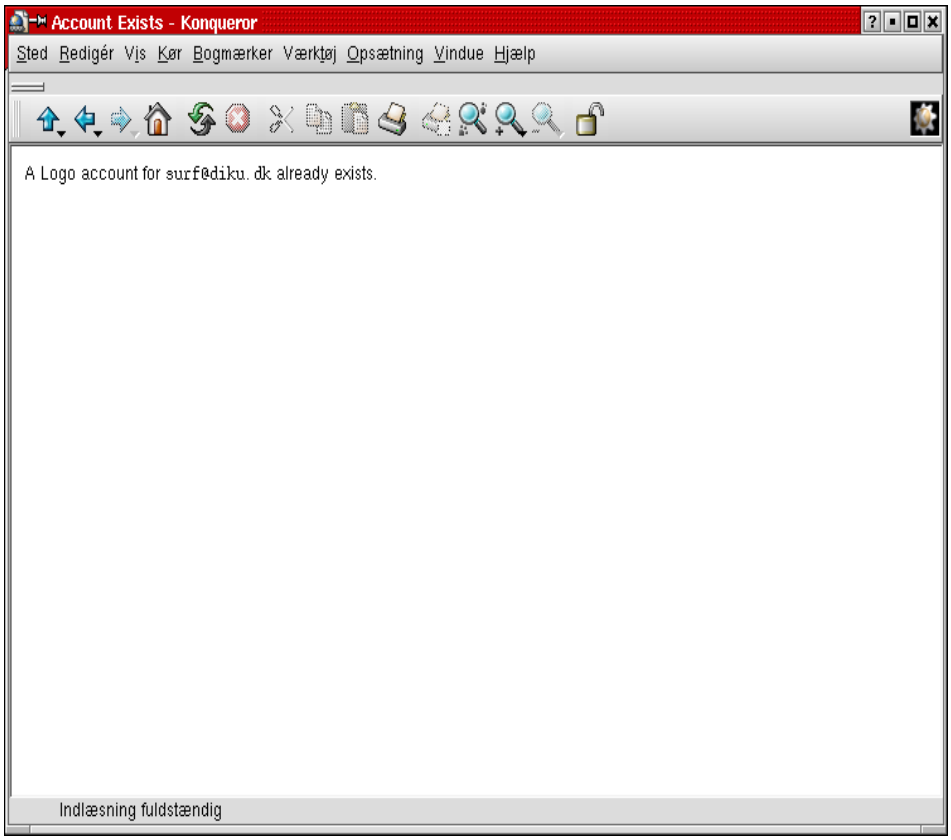
10 Bilag — Det færdige webdesign



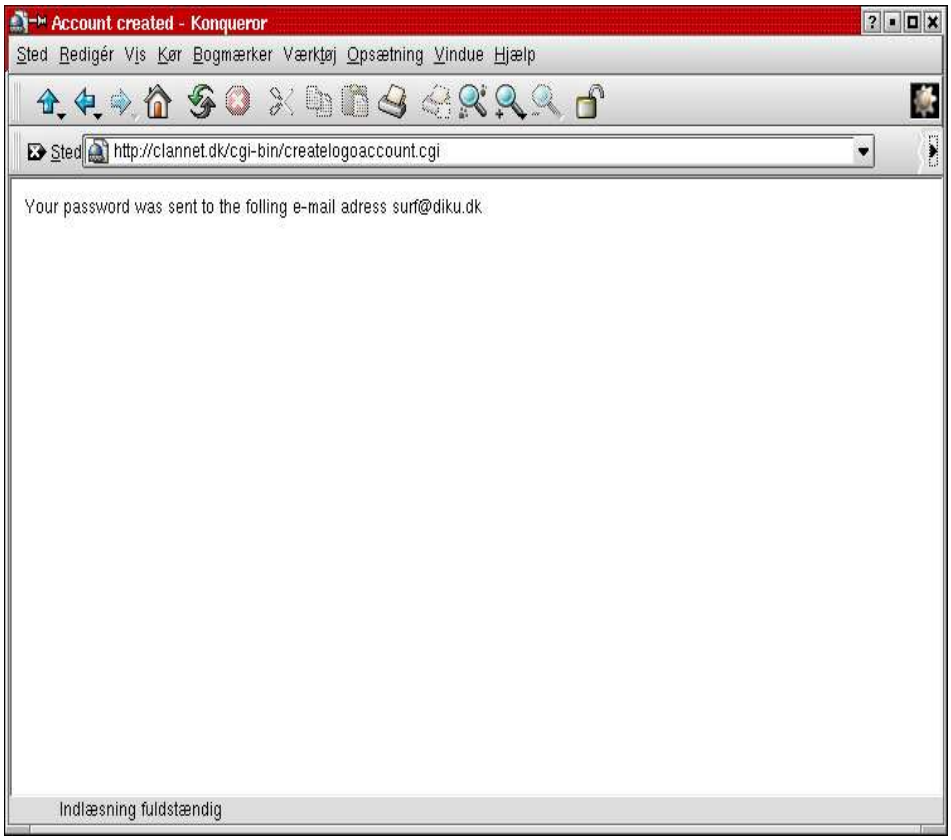
Figur 1: Indgangssiden.



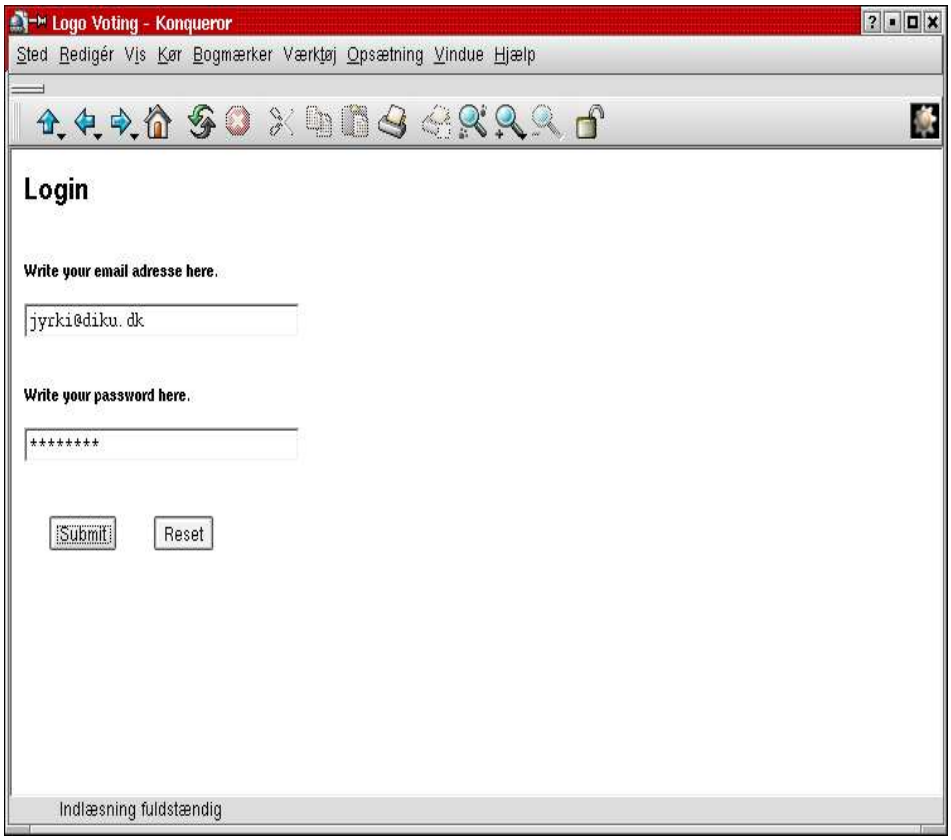
Figur 2: Adgangskodegenereringssiden.



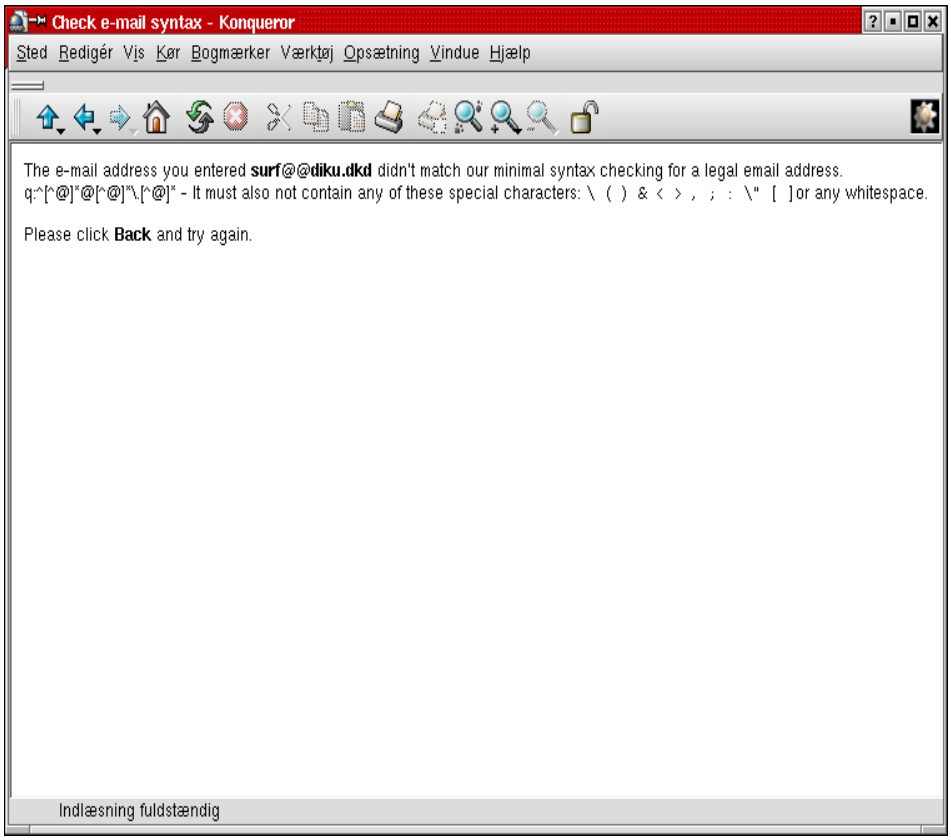
Figur 3: Adgangskodegenereringssiden når brugeren allerede eksisterer.



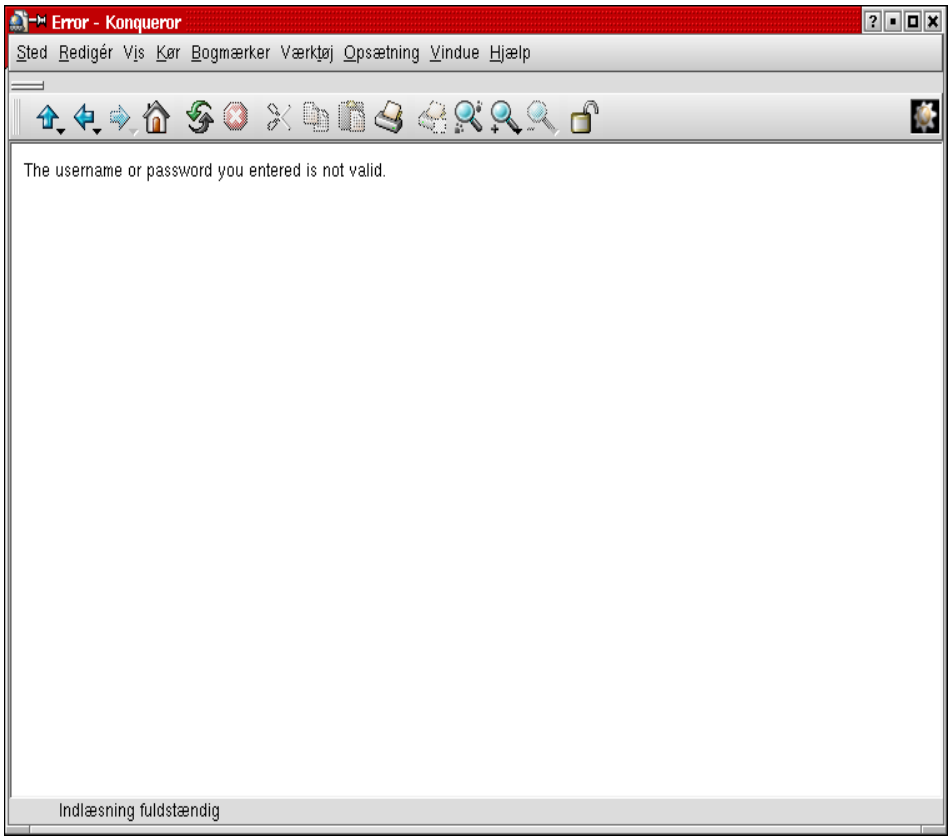
Figur 4: Adgangskodegenereringssiden efter adgangskoden er sent til brugeren.



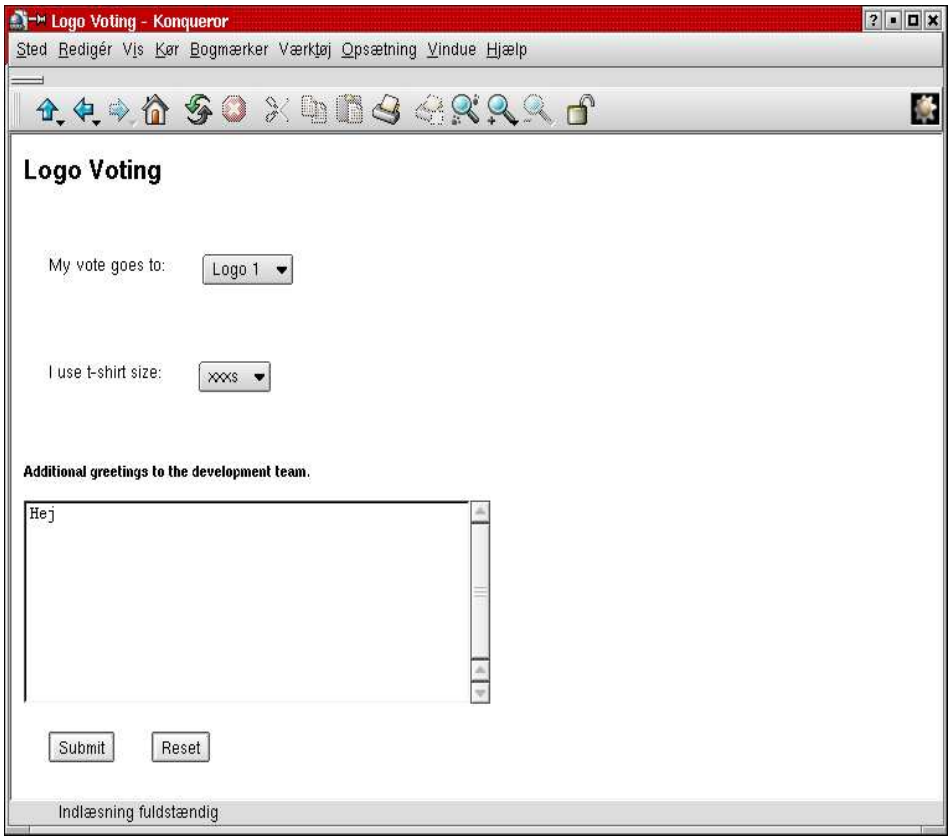
Figur 5: Loginsiden.



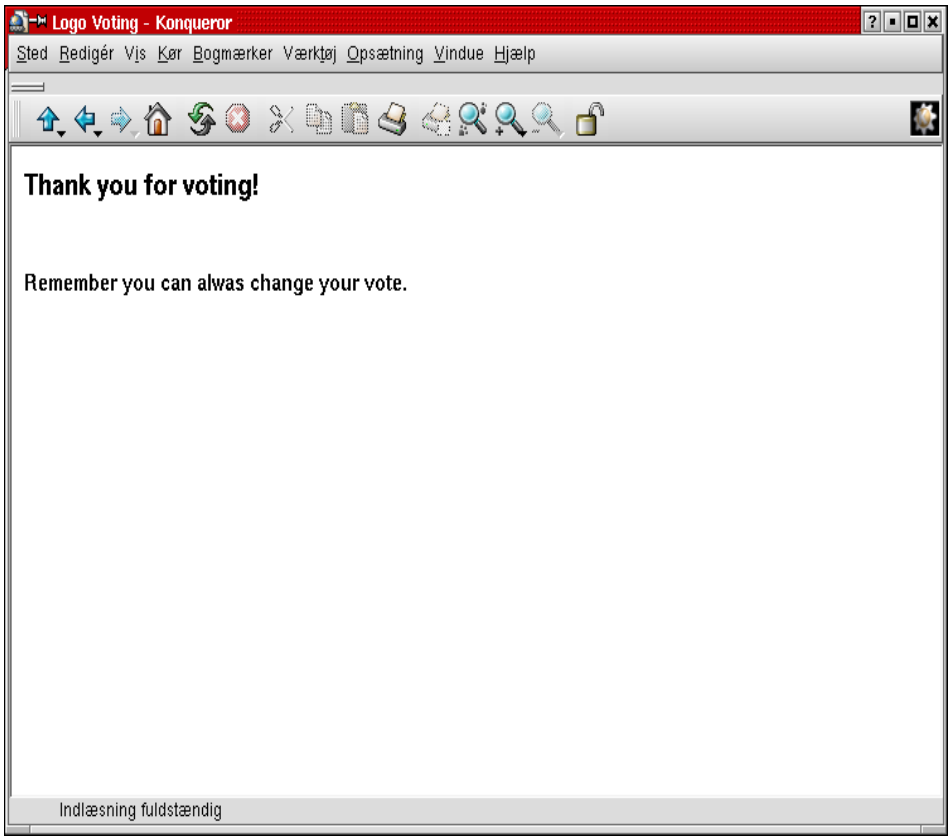
Figur 6: Fejlsiden som vises hvis brugeren indtaster et forkert e-mail format.



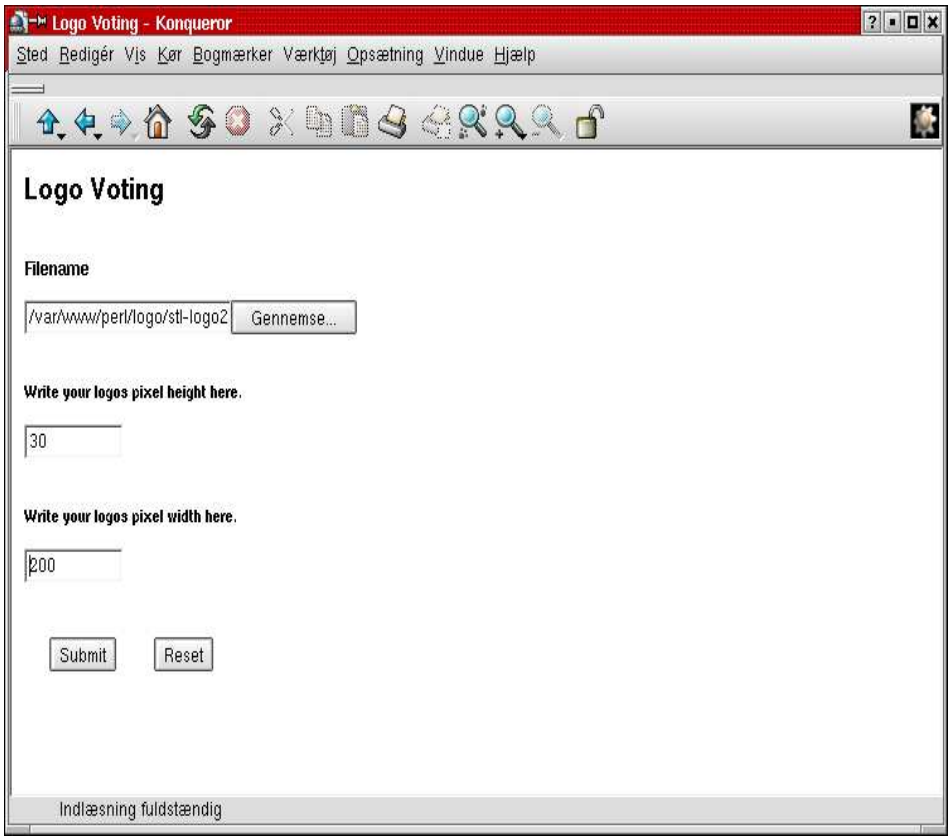
Figur 7: Fejlsiden som vises hvis brugeren indtaster en forkert e-mail eller adgangskode.



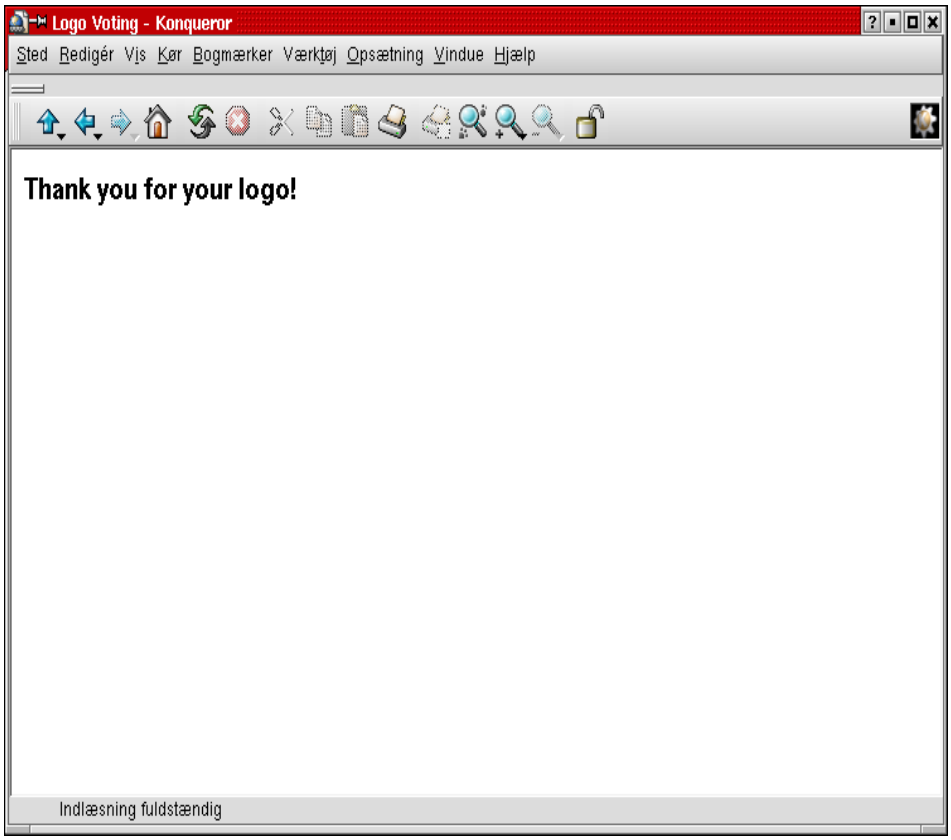
Figur 8: Websiden hvor logoafstemningen foregår.



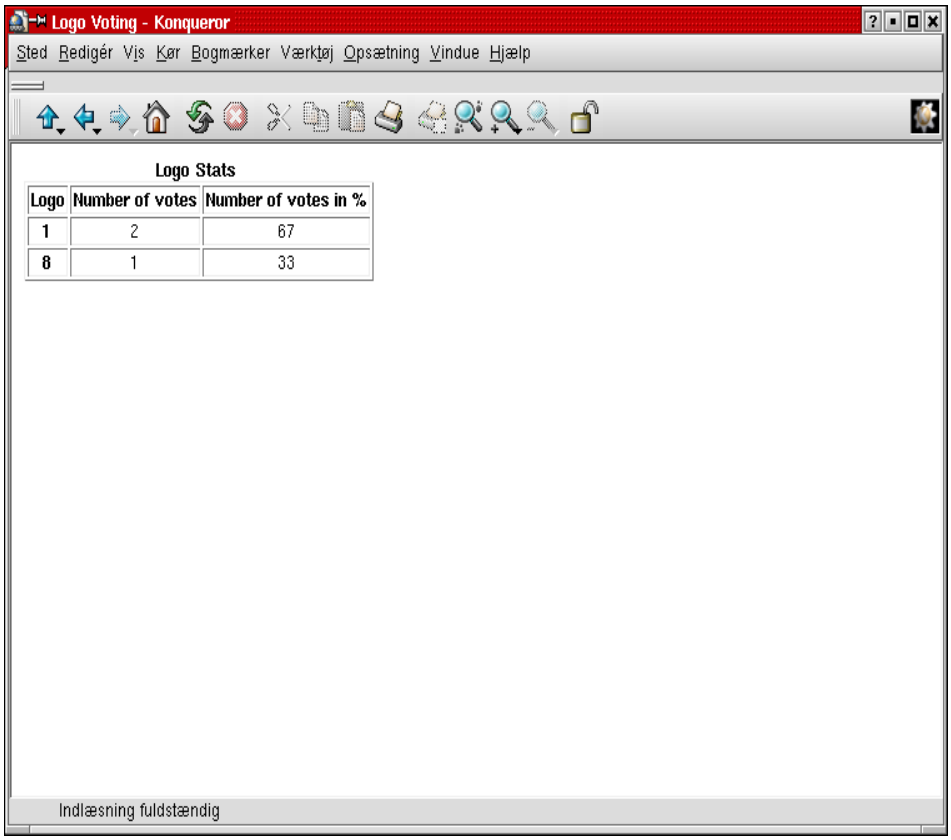
Figur 9: Websiden som vises når brugeren har stemt.



Figur 10: Websiden hvor brugeren kan tilføje egne logoer.



Figur 11: Websiden som vises når brugeren har tilføjet et logo.



Figur 12: Statistiksiden.