BACHELOROPPGAVE:

# RSS

# Repository Self Service

FORFATTERE:
Ron J. Stangvik
Christer Berg

Dato: 20.05.2010

# Sammendrag av Bacheloroppgaven

| | | |
|---|---|---|
| Tittel: | Repository Self Service (RSS) | Nr. : |
| | | Dato : 20.05.10 |
| Deltaker(e): | Christer Berg | |
| | Ron J. Stangvik | |
| Veileder(e): | Erik Hjelmås, Høgskolen i Gjøvik | |
| Oppdragsgiver: | Høgskolen i Gjøvik v/IT-tjenesten | |
| Kontaktperson: | Anders Wiehe | |
| Stikkord (4 stk) | Subversion, Automatisering, Web-applikasjon, CentOS | |

| | | |
|---|---|---|
| Antall sider: 57 + vedlegg | Antall bilag: 8 | Tilgjengelighet: Åpen |


Repository Self Service er et system som automatiserer opprettelse av Subversion repositorier. Gjennom et webgrensesnitt kan brukere av systemet selv opprette, konfigurere og slette sine repositorier. Dette letter arbeidsmengden for IT-tjenesten ved Høgskolen i Gjøvik

Prosjektet består av 2 deler. Teknisk og presentasjon. Den tekniske delen går ut på å sette opp og konfigurere server-miljøet applikasjonen skal kjøre på. Applikasjonen står for presentasjonen mot brukeren, i form av ett webgrensesnitt.

Applikasjonen er utviklet i PHP med MySQL-database i bakhånd som lagrer konfigurasjons-informasjon om repositoriene. Serveren kjører på CentOS 5.

Brukeren av vår løsning vil få mulighet til å legge til samarbeidspartnere på sine repositorier. Det kan da legges til grupper eller enkeltpersoner fra Høgskolens LDAP-database, eller man kan invitere eksterne brukere. Tilgangsrettigheter blir satt til systemets standardinnstillinger, men eier av repositoriet har mulighet til å overstyre dette i konfigurasjon av repository.

Vi har utviklet dette systemet for vår oppdragsgiver på IT-tjenesten ved høgskolen. Derfor er systemet tilpasset høgskolens systemer. Dette kan med litt modifisering av koden enkelt tilpasses andre høgskoler/universiteter. Men også til privat bruk.

# Forord

Denne bacheloroppgaven representerer 3 års studie innen bachelor i drift av nettverk og datasystemer. Oppgaven utgjør 20 studiepoeng. Oppgaven har som formål å lage et system som automatiserer prosessen med å opprette Subversion repositorier. Det som fattet vår interesse fra denne oppgaven var det å lære om prosessen man må igjennom for å automatisere den daglige driften av datasystemer. Automatisering er noe vi vil strekke oss etter så langt det er mulig, når vi kommer ut i arbeid.

Gjennom studiet har vi opparbeidet oss kunnskap om prosjektarbeid, systemutvikling, programmering, databaser og oppsett og konfigurasjon av servere. Du kan se i denne rapporten hvordan vi har benyttet denne kunnskapen til for eksempel valg av systemutviklingsmodell, design av databasen og konfigurasjon av vår utviklingsserver.

Arbeidet med oppgaven har vært svært interessant og lærerik, men også utfordrende. Det har hele tiden kommet opp problemer som har krevd gode undersøkelser for å løse. Det har vært mye nytt fagstoff å sette seg inn i, og vi har brukt mange timer på å lese manualer til ulike applikasjoner og rammeverk vi har benyttet oss av.

Det har hele tiden vært ett ønske fra oppdragsgivers og fra vår side at vi skulle bli ferdig med utviklingen av systemet, slik at dette kan settes i produksjon. Vi har jobbet veldig hardt gjennom dette semesteret for å oppnå denne målsettingen. Med deltidsjobb, familie og andre fag ved siden av, har det blitt meget lange dager. Det er vanskelig å gjenspeile alt arbeidet som er lagt ned i denne rapporten, men vi har lagt vekt på å få med det viktigste.

Gruppen har 2 medlemmer. Selv med en så liten gruppe, har vi valgt å fordele ansvarsforhold for bedre styring. Christer Berg har vært rapport og dokumentansvarlig, mens Ron Stangvik har vært gruppeleder og hatt hovedansvaret for kode. Fordelingen av arbeidet har i praksis vært fordelt likt oss i mellom, hvor begge har kodet og begge har skrevet rapport. Det har ikke vært noe problem å jobbe som team under prosjektperioden. Vi har tatt opp små-problemer i plenum og fått disse avklart muntlig og uformelt underveis.

Vi vil takke IT-tjenesten ved Høgskolen i Gjøvik og vår kontaktperson Anders Wiehe. Det har ikke vært noe å utsette på samarbeidet og vi har fått svært god oppfølging underveis.

Vår veileder har vært Erik Hjelmås, førsteamanuensis. Vi vil takke for et godt samarbeid med gode råd og god veiledning.

Vi vil også rette en stor takk til Ivar Farup, Arne Magnus Bakke, Monica Strand, Lars Arne Sand, Anders Sand Frogner og Gaute Bjørklund Wangen som tok seg tid til å teste systemet og komme med konstruktive tilbakemeldinger.

Gjøvik, 20.05.2010

Ron J. Stangvik

Christer Berg

_____

_____

# Innholdsliste

# 1. Innledning

Bakgrunnen for denne oppgaven er at IT-tjenesten ved Høgskolen i Gjøvik til nå har måtte avsatt sine ressurser for å opprette ***Subversion*** repositorier. Dette er en enkel, men tidkrevende oppgave. IT-tjenesten ønsker så mye ***automatisering*** som mulig, slik at de begrensede ressursene de har til rådighet kan benyttes til andre formål. Utfordringene så langt har vært at de ansatte ved IT-tjenesten er opptatt med daglige gjøremål. De har da ikke selv tid til å utvikle ett slikt system. Det er her vi kommer inn i bildet. Som bacheloroppgave våren 2010 har vi valgt å utvikle ett system som automatiserer opprettelse av Subversion repositorier. Gjennom denne oppgaven får vi praktisk erfaring i systemutvikling og IT-tjenesten får reduserte kostnader.

Vi vil gjennom denne rapporten bruke uttrykkene ***repository*** og repositorier siden det ikke finnes et godt norsk ord som beskriver dette. Vi har sett på ulike oversettelser av dette utrykket, men finner ingen tilfredsstillende oversettelser. En direkte oversettelse av utrykket, vil være "depot". Dette minner mer om ett lagringsdepot, noe som blir feil da Subversion er ett versjonskontroll system. Derfor er det besluttet at vi benytter det opprinnelige utrykket.

## *1.1 Problemområde*

Studenter og ansatte ved Høgskolen i Gjøvik (fra nå av HiG) bruker repositorier for versjonstyring av kildekodefiler og dokumenter, ved prosjekter og skolerelaterte oppgaver. IT-tjenesten tilbyr i dag Subversion repositorier til disse brukerne. Dette gjøres i dag ved at det sendes en forespørsel til IT-tjenesten om å få opprettet et slikt repository. Deretter må en av IT-tjenestens ansatte manuelt taste inn de kommandoene som må kjøres for at ett repository skal bli opprettet, for så sende link til repositoriet via epost til forespørreren.

Repository Self Service er et system som skal lette denne oppgaven for IT-tjenesten slik at brukerne skal selv via en nettside kunne opprette og konfigurere et slikt repository basert på om det skal være et personlig repository eller om det skal brukes i et fag/prosjekt. Oppgaven vår går ut på lage dette systemet.

## *1.2 Målgrupper*

### 1.2.1 Rapporten

Målgruppe for prosjektrapporten vil være sensor, veileder, oppdragsgiver, lærere innen fagfeltet ved HiG og datastudenter.

### 1.2.2 Systemet

Målgruppen for systemet vil i hovedsak rette seg mot IT-tjenesten, ansatte og studenter ved HiG, som en enklere løsning i forhold til den måten det i dag opprettes repositorier på.

IT-tjenesten vil stå for distribusjon og drift av løsningen.

## 1.3 Formål

Formålet med prosjektet er å utvikle ett system som automatiserer dagens løsning med opprettelse av repositorier, som i dag gjøres manuelt av IT-tjenesten.

Systemet skal gjøre det lettere for ansatte og studenter å opprette og administrere sine egne repositorier og gi tilgang til de som eventuelt skal samarbeide på samme oppgave/prosjekt. Det vil lette arbeidsoppgavene for IT-tjenesten og gi dem tid til andre gjøremål. IT-tjenesten med sin administratortilgang vil kunne gjøre utvidede endringer som brukerne ikke kan. Eksempelvis endringer av kvote og antall repositorier per bruker.

Dette vil øke brukernes mulighet for selvhjelp og redusere ventetid på at deres forespørsel skal godkjennes og opprettes. Dette fører til at brukerne med en gang kan få tilgang til sitt repository.

## 1.4 Gruppens bakgrunn og kompetanse

Gruppen har 2 medlemmer. Begge tar studiet bachelor i drift av nettverk og datasystemer, så kunnskaper innen basis nettverk og server-konfigurasjon sitter begge inne med. Vi sitter med lite erfaring innenfor systemutvikling.

I studiet har vi vært igjennom grunnleggende programmering og utvikling, men innenfor temaet vi nå holder på med har vi kun vært igjennom ***SQL***-kode og veldig enkel ***PHP*** kode.

Begge tar faget WWW-teknologi som går ut på å kode webapplikasjoner i PHP og ***AJAX***. Dette kom godt med, da løsningen skulle lages i PHP.

Når det kom til miljøet som løsningen skulle kjøre på måtte vi sette oss inn i et helt nytt operativsystem. HiG bruker for det meste ***CentOS*** på sine servere og det var ukjent område for vår del, men siden det er et ***UNIX***-basert operativsystem var mye kjent fra før med tanke på at vi har jobbet mye med Debian og Ubuntu fra før.

Vi måtte også sette oss nøye inn i ***LDAP***, som vi hadde vært såvidt innom tidligere, siden all brukerautentisering foregår gjennom LDAP. Her hadde PHP egne funksjoner for å løse dette. Denne kompetansen tilegnet vi oss gjennom samtaler med oppdragsgiver og veileder, samt mye lesing av dokumentasjon.

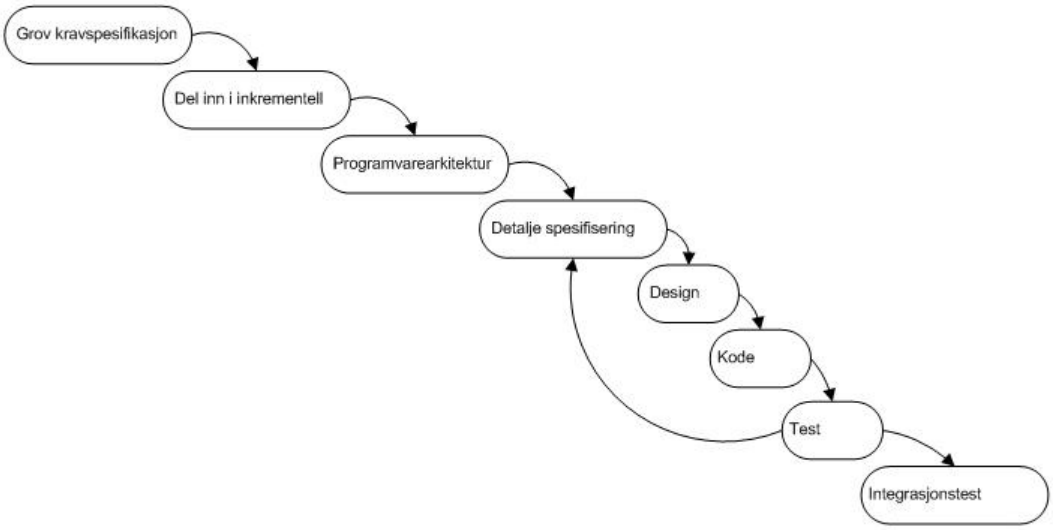
## 1.5 Utviklingsmodellen

Kravene til systemet var allerede kommet fra oppdragsgiver i en grovutgave ved oppstart av prosjektet. Disse kravene viste klart at systemet burde bygges opp av moduler. Så da valget av systemutviklingsmodell skulle tas, måtte den støtte modul-basert utvikling og at kravene allerede var tilstede. Den måtte også kunne behandle endringer underveis. Vi ville også ha muligheter for å ha overlappende moduler siden noen av modulene bygger på hverandre.

Vi styrte fort unna en evolusjonær modell fordi den har en tendens til å bli uoversiktlig i større prosjekter. Og siden denne modellen tar for seg alle moduler på en gang var dette også et minus.

Vi satt igjen med fossefallmodellen og inkrementell-modellen. Når det kommer til fossefallmodellen så passet ikke den vårt prosjekt, siden den tok for seg endringer dårlig underveis, og den er noe treg.

Valget falt da på inkrementell-modellen(figur 1-1) siden den egner seg best på modul-basert programvare da den deler opp i inkrementer. Siden den i hvert inkrement har testing før den integreres finner man lett hva som må endres. Derfor takler denne modellen endringer som dukker opp underveis lett. Når man bruker en inkrementell-modell ser den på prioriteten av funksjonaliteten og produsere den modulen med størst prioritet først. Dette gjør at den mest kritiske funksjonaliteten kommer på plass først og det fører til lavere risiko når det kommer til feil som dukker opp i programvaren.




*Figur 1-1 - Inkrementellmodellen*

## *1.6 Rammer*

### 1.6.1 Arbeidsmetoder

Det første vi gjorde i prosjektet var å planlegge løpet fremover ved å lage en fremdriftsplan. Vi har delt hovedansvaret for de ulike oppgavene oss i mellom. I forbindelse med implementering av ulike funksjoner i kildekoden og når vi har satt opp nye system, har vi måtte lese manualer for å få oversikt over oppsett og funksjonalitet. Dette er en tidkrevende prosess, men dette er nødvendig for å sikre kvaliteten på prosjektet.

I forbindelse med denne rapporten har vi sett på strukturen til tidligere oppgaver som har gjort det bra, og bygget opp vår egen rapport ut fra disse. Det er plukket det beste fra de forskjellige rapportene, så vi har ikke sett på kun én spesifikk rapport.

Ved prosjektstart fikk vi tildelt grupperom A018C som har vært det rom som har blitt benyttet under hele prosjektperioden, noe som ble avtalt på forhånd av prosjektet.

I starten av prosjektet hadde en ansvaret for kodingen mens den andre skulle få i gang test-serveren. Så fort dette var gjort skulle andremann også bidra videre med koding resten av prosjektet.

### 1.6.2 Fremdriftsplan

Gruppens fremdriftsplan vil bli presenter i vedlegg B i form av et gant-skjema for estimert tidsplan og faktisk tidsplan.

## *1.7 Roller*

### 1.7.1 Gruppen

Ron Stangvik – Prosjektleder, kodeansvarlig og ansvaret for gruppens hjemmeside.

Christer Berg – Rapport- og dokumentansvarlig. Sammensetting av rapporter.

Gruppen har hatt like stort ansvar for at alt skal være levert i henhold til tidsfristene, men for å få bedre oversikt har vi valgt hvert vårt område vi skal ha oversikt over, hva som mangler og hva som må gjøres for å rekke unna det forskjellige arbeidet. Vi har hele vegen samarbeidet som et team, og har i fellesskap jobbet sammen for å bli ferdig med prosjektet.

### 1.7.2 Veileder

Erik Hjelmås – Gruppens veileder. Ressursperson på fagområder som systemadministrasjon.

### 1.7.3 Oppdragsgiver

Anders Wiehe – Kontaktperson ved IT-tjenesten. Ressursperson på serveroppsett og skolens systemer.

## *1.8 Rapporten*

Rapporten vil følge malen for utviklingsprosjekter som ligger ferdig fra HiG sine retningslinjer for bachelorprosjekt.

Uttrykk som er skrevet ***fet og kursiv*** er uttrykk som blir nærmere forklart i Vedlegg A – Definisjoner.

Nummerering av figurer og tabeller er på formatet:

– Fig/Tabell. Kapittelnummer – figur/tabellnummer(eks. Fig. 1-1)

Mappe- og fil-stier vil bli skrevet i *kursiv skrift,* mens innhold i filer blir skrevet i **fet skrift.**

Kommandoer brukt i Linuxterminal presenteres på denne måten: eks. **`rpmbuild -ba <navnet på spec-filen>`**

### Kapitteloversikt:

Rapporten inneholder 9 kapitler og for å få en rask oversikt over hva de forskjellige kapitlene omhandler følger en kort oppsummering av hvert enkelt kapitel.

Kapitel 1: Innledning.

Her blir en grov oversikt over formålet med prosjektet beskrevet, rammene rundt oppgaven og de forskjellige involverte parters roller.

Kapitel 2: Kravspesifikasjon.

Inneholder krav til oppgaven som måtte på plass før prosjektet kunne påbegynnes.

Kapitel 3: Design.

Her beskrives aspektene rundt hvordan designet på løsningen skulle bli.

Kapitel 4: Implementering.

Her beskrives utviklingsmiljøet, de programmer som er brukt under utviklingen og eksempler på skjermbilder, kodeeksempler til disse og rammeverk som har blitt brukt.

Kapitel 5: Testing.

Inneholder egentester gjort underveis i utviklingen, aspekter rundt første beta-utgivelse og brukertester med brukere.

Kapitel 6: Realisering.

Inneholder de aspekter rundt levering av det ferdige systemet. Herunder RPM-pakken og

installasjonen av denne.

Kapitel 7: Avslutning

Inneholder drøftinger rundt prosjektet, egne evalueringer, forslag til forbedringspotensiale og videreutvikling og konklusjon,

Kapitel 8: Referanser.

Inneholder referanser til brukt litteratur og nettsider.

Kapitel 9: Vedlegg.

# 2. Kravspesifikasjon

## *2.1 Introduksjon*

I dette kapitlet vil kravene til systemet presenteres. Opplysninger om krav har blitt sendt til oss fra oppdragsgiver hvor det er presisert hvordan de ønsker oppsett og funksjonalitet. Det er også blitt presisert krav muntlig ved møter og gjennom lynmeldingsnettverket ***IRC***.

### 2.1.1 Kort om krav til systemet

Systemet skal ved utrulling bli den automatiserte utgaven av den manuelle jobben IT-tjenesten i dag gjør ved opprettelse av repositorier. Dette skal være et webgrensesnitt, kodet i PHP, og skal fungere på en server med CentOS 5 operativsystem. Siden det også er en betydelig mengde data som bygger seg opp over tid, så vil også serveren trenge godt med lagringsplass.

Brukere skal kunne logge seg inn, opprette og generelt administrere egne repositorier.

IT-tjenesten vil med sin egen admin-modul kunne holde oversikt over alle repositorier som er opprettet, endre egenskaper på utover standard oppsett. Modulen skal liste repositorier som har gått ut på dato, slik at det er enklere for administrator å fjerne disse.

### 2.1.2 Kort om systemets omgivelser

Systemet skal kjøre på en fysisk dedikert server plassert i HiG's server rom. Direkte vedlikehold og tilgang til serveren vil bli gjort av administratorene, mens brukere kun vil bruke webgrensesnittet til å vedlikeholde sine repositorier.

## *2.2 Brukerbeskrivelse*

### 2.2.1 Omgivelser

Systemet skal kjøre på en dedikert server som benytter Linux-distribusjonen CentOS 5, som er gratisversjonen av Red Hat Enterprise Linux uten kommersiell kundestøtte. De fleste modulene vårt system avhenger av er allerede installert på serveren fra før, eller finnes i de globale repositoriene serveren har tilgang til. Systemet skal benytte autentisering fra den eksisterende LDAP- databasen HiG allerede kjører.

Det skal også være mulig å legge til brukere som ikke ligger i LDAP-databasen. Informasjon om disse brukerne vil bli lagret på en lokal ***MySQL*** database. Disse brukerne skal ikke ha tilgang til webgrensesnittet, men kun ha tilgang til selve repositoriet.

Oppdragsgiver sørger selv for valg av maskinvarekonfigurasjon, fysisk plassering og sikring av serveren. Systemet blir skrevet så generelt som mulig, slik at det kan kjøres på de fleste Unix-baserte maskiner som støtter ***Apache*** og PHP.

### 2.2.2 Systemets brukere

Systemet skal i hovedsak benyttes av studenter og ansatte ved HiG. Systemet skal ikke være et sted hvor brukeren kan søke dokumentasjon og veiledning. Studenter og ansatte som er aktuelle brukere av systemet får tilbud om kurs med innføring i bruk av Subversion av HiG. Annen dokumentasjon er linket fra http://minside.hig.no

Ettersom brukerne av systemet kommer til å være fra alle samfunnsklasser og i alle aldre, vil vi legge vekt på at som normal bruker skal det være minimalistisk design med få valg som er tydelige og enkle å forstå. Informasjon om eksisterende repositorier vil også være tilgjengelig for brukeren.

Operatørene av admin-modulen skal også få et enkelt brukergrensesnitt, men med flere valg som brukeren ikke skal ha tilgang til. Det er tenkt at IT-tjenesten ved HiG er de som skal administrere systemet. Ansatte ved IT-tjenesten har litt dypere teknisk innsikt enn normale brukere, og vil derfor få tilgang til litt mer avanserte funksjoner.

### 2.2.3 Funksjon

#### 2.2.3.1 Login modul

Denne modulen skal være enkel og lettfattet. Brukeren taster inn gyldig brukernavn og passord for så å bli sendt videre til brukermodul eller administrasjons modul, alt ettersom hvilke rettigheter brukeren har. Dersom brukernavn og passord ikke er gyldig vil det bli gitt en beskjed om dette, og brukeren vil ikke få tilgang.

#### 2.2.3.2 Brukermodul

Her vil brukeren få listet repositoriene han har tilgang til. Dersom brukeren har lov å opprette nytt repository vil brukeren få mulighet for dette. Dersom maks antall repositorier er nådd, vil ikke brukeren lengre ha muligheten til å opprette repository.

Dersom brukeren ønsker å opprette et gruppe-repository kan han selv velge gruppemedlemmer som finnes i LDAP- eller lokal database.

#### 2.2.3.3 Administrasjons-modul

På dette webgrensesnittet skal aktørene med administratorrettigheter kunne endre standardbegrensninger for hele systemet eller bare enkelte brukere.

Systemet skal liste alle utgåtte repositorier slik at administrator har mulighet til å endre egenskapene for disse, eller fjerne repositoriene helt.

### 2.2.4 Operasjon

Systemet skal ha 100% oppetid så lenge modulene systemet er avhengig av fungerer som de skal. Apache må lastes på nytt hver gang repositorier er endret på. Dette skal ikke ha noen påvirkning av brukeropplevelsen.

Alle passord som skal lagres i lokal database skal krypteres på en sikker måte, slik at dette ikke kan leses i klartekst dersom ett angrep på databasen gjennomføres suksessfullt.

Systemadministrator konfigurer hvem som har administratortilgang ved å endre konfigurasjonsfilen for systemet.

### 2.2.5 Aspekter omkring livssyklus

Systemet er designet for å kjøre på én maskin. En administratorbruker og/eller administratorgruppe skal kunne vedlikeholde systemet fra et webgrensesnitt ved kontinuerlig drift. Kildekoden i prosjektet skal kommenteres på en forståelig måte slik at systemet kan videreutvikles ved ett senere tidspunkt dersom dette er ønskelig.

Det er mulig å flytte hele systemet på en annen fysisk maskin. Det eksisterer ingen automatisk ***rutine*** for dette, så flytting av hele systemet til en ny maskin må gjøres og verifiseres manuelt. Dersom databasen og mappestrukturen forholder seg likt, behøver man bare måtte kopiere over repository-mappene og importere databasen. For en systemadministrator er dette en forholdsvis enkel operasjon.

### 2.2.6 Ytelse

Det er vanskelig å måle ytelsen til systemet i tall, da det er mange faktorer som spiller inn. Nettverk, andre prosesser på maskinen, backup og lignende. Systemet skal ikke være tregere enn andre store webapplikasjoner som for eksempel nettaviser. Ved stor last i enkelte perioder kan responstiden være noe høyere. Ved oppstart av fag hvor det anbefales bruk av repository ser vi for oss at systemet kan være noe tregere enn normalt, da pågangen ofte øker ved slike situasjoner. Responstiden kan minskes ved at oppdragsgiver oppgraderer maskinvare. Lasten vil være ***dynamisk*** etter når brukerne benytter seg av sitt repository. Vi ser for oss at lasten vil være størst på dagtid, slik at oppdragsgiver bør kjøre oppdateringer på operativsystemet på kveld/natt.

## *2.3 Detaljert kravspesifikasjon*

### 2.3.1 Funksjonell spesifikasjon

For å få en god grafisk oversikt over de ulike funksjonalitetene i systemet ble det utarbeidet et Use case diagram. Dette er presentert i figur 2-1. Forklaringer på de ulike funksjonalitetene følger i tabellene under figur 2-1.

Eier innehar funksjonalitet spesifikk for seg selv i tillegg til det en vanlig bruker har. Administrator innehar funksjonalitetene som eier og bruker i tillegg til sine egne. Forskjellen på bruker og eier er at en bruker bare er medlem av et repository mens en eier er medlem og i tillegg kan endre rettighetene på repositoriet.

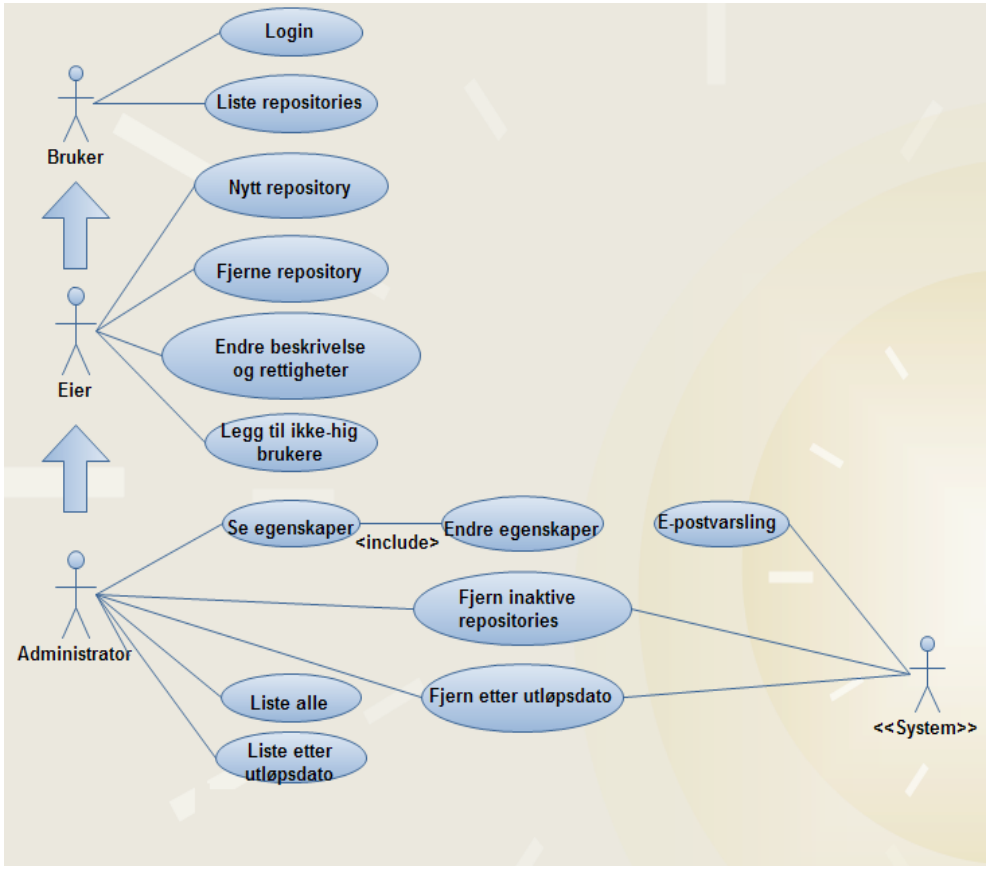



*Fig. 2-1: Use case diagram*

**Overordnet Use case:**

| | |
|---|---|
| **Use case:** | Login |
| **Aktør:** | Bruker, eier, administrator |
| **Mål:** | Få tilgang til å opprette repository, oversikt over repositorier og endre innstillinger. |
| **Beskrivelse:** | Aktør logger seg inn via nettsiden for så å få mulighet til å opprette og gjøre de innstillinger som kreves for å gjøre endringer. |

| | |
|---|---|
| **Use case:** | Nytt repository |
| **Aktør:** | Eier, administrator |
| **Mål:** | Opprette nytt repository for eget eller gruppe- bruk. |
| **Beskrivelse:** | Aktør vil her opprette nytt repository og sette innstillinger som beskriver repository og rettigheter settes. |

| | |
|---|---|
| **Use case:** | Fjern repository |
| **Aktør:** | Eier, administrator |
| **Mål:** | Aktør skal kunne slette egne repositorier. |
| **Beskrivelse:** | Aktør har mulighet til å slette repositorier han ikke har behov for lenger. Dette kan bare gjøres om aktør er eier av repository eller administrator. |

| | |
|---|---|
| **Use case:** | Liste repositories |
| **Aktør:** | Eier, bruker, administrator |
| **Mål:** | Få oversikt over tilhørende repositorier. |
| **Beskrivelse:** | Aktør vil her få listet opp alle repositorier som han er eier av eller har rettigheter til. Det vil også bli listet opp hvor mye plass som er brukt og totalkvoten på hvert repository. |

| | |
|---|---|
| **Use case:** | Endre beskrivelse og rettigheter |
| **Aktør:** | Eier, administrator. |
| **Mål:** | Få endret tilgjengelige innstillinger |
| **Beskrivelse:** | Aktør kan her endre beskrivelser, utløpsdato og tilgangsrettigheter ved eventuelle endringer. |

| | |
|---|---|
| **Use case:** | Liste alle repositories |
| **Aktør:** | Administrator. |
| **Mål:** | Få oversikt over alle opprettede repositorier. |
| **Beskrivelse:** | Aktør vil her få listet opp alle repositorier som er opprettet opprettet alfabetisk og med stien til disse. |

| | |
|---|---|
| **Use case:** | Liste repositories etter utløpsdato. |
| **Aktør:** | Administrator. |
| **Mål:** | Få oversikt over når repositorier går ut på dato. |
| **Beskrivelse:** | Aktør vil her få listet opp alle repositorier etter når de nærmer seg utløpsdato satt ved opprettelse. |

| | |
|---|---|
| **Use case:** | Legge til ikke-HiG brukere |
| **Aktør:** | Eier, Administrator |
| **Mål:** | Gi tilgang til repository for brukere utenfor HiG |
| **Beskrivelse:** | Aktør oppretter «profil» på brukeren og gir tilgang til det repository han skal jobbe med. |

| | |
|---|---|
| **Use case:** | Se egenskaper |
| **Aktør:** | Administrator |
| **Mål:** | Se hvilke egenskaper og innstillinger som er satt. |
| **Beskrivelse:** | Aktør går inn på ønsket repository og får opp de egenskaper og innstillinger som er satt på hvert enkelt repository. |

| | |
|---|---|
| **Use case:** | Endre egenskaper |
| **Aktør:** | Administrator |
| **Mål:** | Endre egenskaper og innstillinger som er satt. |
| **Beskrivelse:** | Aktør kan fra «se egenskaper» gå inn og endre de innstillinger og egenskaper utover det vanlige brukere kan sette selv. |

| | |
|---|---|
| **Use case:** | E-postvarsling |
| **Aktør:** | System |
| **Mål:** | Varsle via e-post ved check-in og ved nær oppbrukt kvote. |
| **Beskrivelse:** | Systemet vil sende e-post til eier om «stien» til repositoriet ved check-in og en e-post når det begynner å nærme seg limit på lagringskvoten på det aktuelle repository. |

| | |
|---|---|
| **Use case:** | Fjern etter utløpsdato |
| **Aktør:** | System, administrator |
| **Mål:** | Fjerne utgåtte repositorier. |
| **Beskrivelse:** | Systemet varsler på utløpsdato og sletter automatisk etter 3 mnd. og sender varsel. Administrator kan slette før dette. |

| | |
|---|---|
| **Use case:** | Fjern inaktive repositories |
| **Aktør:** | System, administrator |
| **Mål:** | Fjerne repositorier hvor eier og brukere ikke er ved HiG lenger eller bare de ikke-aktive deltagere.. |
| **Beskrivelse:** | Systemet vil sjekke jevnlig om eier og/eller brukere er tilstede ved HiG og sende mail til administrator om at repository har ikke-aktive deltagere. Administrator logger seg inn og sletter de aktuelle repositorier. |

### 2.3.2 Funksjonell struktur og tverr-relasjoner

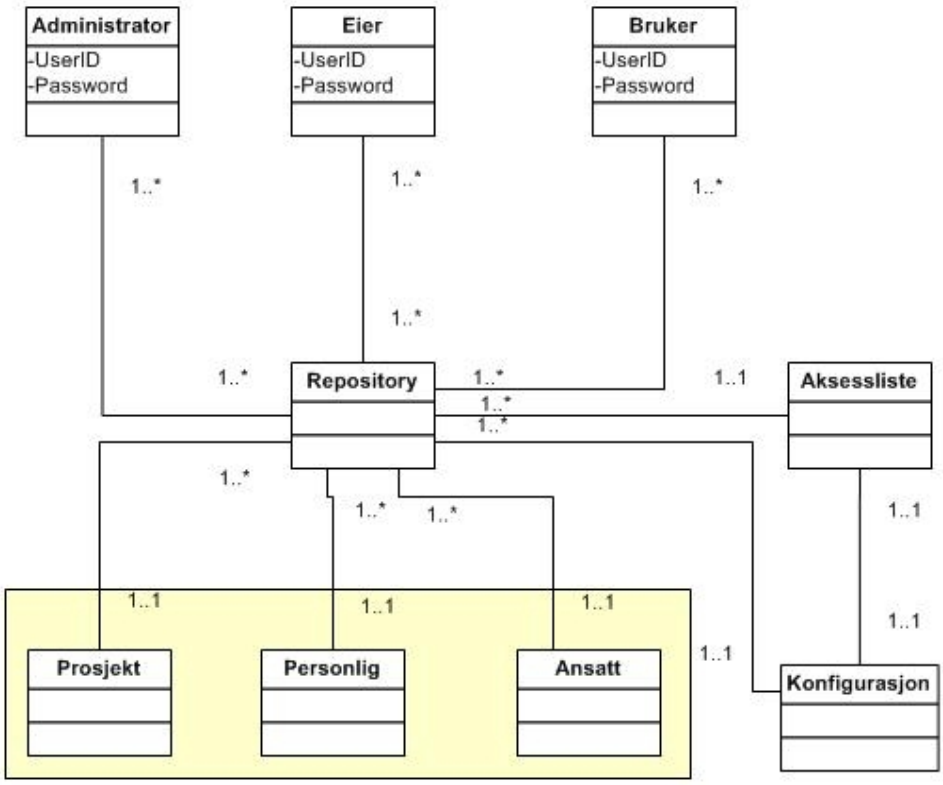



*Fig. 2-2: Konseptuelt klassediagram*

Som man kan se av figur 2-2 har en administrator tilgang til alle repositorier i systemet og kan være medlem av flere repositorier. En eier og/eller en bruker kan være medlem av flere repositorier.

Et repository kan kun ha en konfigurasjon og kun en liste som styrer aksessen. Et repository kan bare være av en type, mens en spesifikk type repository kan være en samling av flere repositorier.

### 2.3.3 Funksjonelle krav

#### 2.3.3.1 Risikoanalyse

*Teknologisk risiko:*

Hvor vanskelig vil det være å produsere enheten og har vi tilstrekkelig med teknologi til å gjøre det.

*Forretningsmessig risiko:*

Hvor viktig vil det være at de ulike Use casene fungerer som de skal i forhold til våre og kundens krav.

*Prosjektmessig risiko:*

Kan Use casene føre til problemer med tanke på organisering, samarbeid og koordinering av andre Use case/deler av prosjektet.

| **Use case** | **Teknologisk** | **Forretning** | **Prosjekt** | **Total** |
|---|---|---|---|---|
| Login | Lav | Høy | Høy | **Høy** |
| Liste repositories | Lav | Lav | Middels | Lav |
| Nytt repository | Lav | Høy | Middels | **Middels** |
| Fjerne repository | Middels | Middels | Lav | Middels |
| Endre beskrivelse og rettigheter | Middels | Høy | Høy | **Høy** |
| Legg til ikke-hig brukere | Lav | Middels | Middels | Middels |
| Se egenskaper | Middels | Middels | Lav | Middels |
| Endre egenskaper | Middels | Middels | Lav | Middels |
| Liste alle | Lav | Lav | Middels | Lav |
| Liste etter utløpsdato | Lav | Lav | Middels | Lav |
| E-postvarsling | Middels | Middels | Lav | Middels |
| Fjern inaktive repositories | Høy | Middels | Lav | Middels |

| | | | | |
|---|---|---|---|---|
| Fjern etter utløpsdato | Middels | Middels | Lav | Middels |

*Tabell 2-1: Risikoanalyse Use case*

En gjennomgang av samtlige i en Risikoanalyse(Tabell 2-1) av Use Casene og en påfølgende diskusjon har ført til at vi ønsker å presentere disse tre Use Casene i Expanded Use Case.

Valget av disse er grunnet med at de er essensielle når det gjelder generell administrering og opprettelse av repositorier.

### 2.3.3.2 Expanded Use case

| | |
|---|---|
| **Use case:** | Login |
| **Aktør:** | Bruker, eier, administrator |
| **Mål:** | Få tilgang til å opprette repository, oversikt over repositorier og endre innstillinger. |
| **Beskrivelse:** | Aktør logger seg inn via nettsiden for så å få mulighet til å opprette og gjøre de innstillinger som kreves for å gjøre endringer. |
| **Type:** | Høy |
| **Pre:** | Må være en gyldig bruker |
| **Post:** | Får tilgang til funksjoner tilsvarende sine rettigheter. |
| **Spesielle krav:** | |
| **Hendelseflyt** | |
| **Aktør** | **System** |
| 1. Aktør sender brukernavn og passord. | |
| | 2. Sjekker om gyldig brukerID og sjekker rettigheter.<br>3. Laster inn side med funksjoner tilsvarende de rettigheter brukeren har. |
| | |
| **Alternativ hendelsesflyt** | |
| Feil brukernavn/passord. Gjennomfør Hendelse 1 på nytt. | |

| | |
|---|---|
| **Use case:** | Endre beskrivelse og rettigheter |
| **Aktør:** | Eier, administrator. |
| **Mål:** | Få endret tilgjengelige innstillinger |
| **Beskrivelse:** | Aktør kan her endre beskrivelser, utløpsdato og tilgangsrettigheter ved eventuelle endringer. |
| **Type:** | Høy |
| **Pre:** | Må være innlogget via nettsiden. |
| **Post:** | Får tilgang til å endre beskrivelser og rettigheter etter tilgangsrettigheter de har. |
| **Spesielle krav:** | Må være en gyldig bruker med riktige rettigheter. |
| **Hendelsesflyt** | |
| **Aktør** | **System** |
| 1. Velger endre repository | |
| | 2. Laster ny side med repositorier aktøren har tilgang til. |
| 3. Velger repository som han vil endre på. | |
| | 4. Henter innstillinger til repositoriet og viser det for aktør. |
| 5. Gjør de endringer som trengs. | |
| | 6. Gjør endringer på de nødvendige konfigurasjonsfilene og databasen. |
| **Alternativ hendelsesflyt** | |
| Velger repository han ikke har tilgang til. Gjennomfør hendelse 3 på nytt.<br>Klarer ikke liste repositorier. Gjennomfør hendelse 1 på nytt.<br>Gjør ugyldig endringer. Gjennomfør hendelse 5 på nytt. | |

| | |
|---|---|
| **Use case:** | Nytt repository |
| **Aktør:** | Eier, administrator |
| **Mål:** | Opprette nytt repository for eget eller gruppe- bruk. |
| **Beskrivelse:** | Aktør vil her opprette nytt repository og sette innstillinger som beskriver repository og rettigheter settes. |
| **Type:** | Middels |
| **Pre:** | Må være innlogget via nettsiden |
| **Post:** | Får tilgang til repository og begynt å lagre arbeid. |
| **Spesielle krav:** | Må ikke ha brukt opp kvoten for antall repository. |
| **Hendelseflyt** | |
| **Aktør** | **System** |
| 1. Velger å opprette nytt repository. | |
| | 2. Sjekker om forespørrer kan opprette flere repositorier.<br>3. Laster ny side for opprettelse av nytt repository. |
| 4. Setter innstillinger og beskrivelser for repository.<br>5. Trykker opprett. | |
| | 6. Sender innstillinger til konfigurasjonsfiler, aksessliste og database blir oppdatert.<br>7. Sender mail til eier om stien til forespørrer. |
| **Alternativ hendelsesflyt** | |
| Ikke lov til å opprette nytt repository. Feilmelding vises.<br>Manglende innstillinger, gjør om hendelse 4. | |

## *2.4 Begrensninger*

### 2.4.1 Software design begrensninger

#### 2.4.1.1 Software standarder og språk

Vi har sammen med oppdragsgiver valgt å benytte PHP5 som kodespråk og MySQL5 som intern database. Dette er en kombinasjon som er meget utbredt blant utviklere som benytter åpen kildekode og det finnes masse ressurser på nettet som vi kan benytte oss av i prosjektet. Oppdragsgiver foretrekker denne kombinasjonen også, da dette er noe han selv er vant til å jobbe med.

Rapporter vi genererer under prosjektperioden skal følge fastsatt formatering som ligger på gruppens Subversion-område. På denne måten vil vi ha et konsistent oppsett gjennom alle rapporter.

Versjonkontrollsystemet vil ta seg av revisjoner på de forskjellige dokumentene og kildekoden som blir laget under prosjektperioden.

#### 2.4.1.2 Software grensesnitt

PHP-rammeverket tilbyr en mengde moduler for grensesnitt mot andre applikasjoner. Vi vil benytte noen av disse grensesnittene for å koble oss opp mot HiG's LDAP database for autentifikasjon og mot vår interne MySQL database for lagring av informasjon om repositoriene.

#### 2.4.1.3 Software pakker/verktøy

Vår oppdragsgiver krever at systemet skal være kompatibelt med operativsystemet CentOS 5. Det endelige produktet skal leveres som en pakke som kan installeres.

Ettersom applikasjonene vårt system benytter seg av allerede finnes i det globale repositoriet til CentOS, trenger vi kun å legge inn instruksjoner om hvilke applikasjoner systemet avhenger av, i tillegg til vår egen kode, i den ferdige pakken. Når pakken vi leverer installeres vil de nødvendige pakkene bli automatisk nedlastet og installert, så sant de ikke er installert korrekt fra før. Pakken skal være av typen ***RPM*** som er standard på CentOS.

# 3. Design

## *3.1 Innledning*

Dette kapitlet gir en oversikt over hele designdokumentet. Dette dokumentet beskriver alle data, arkitektur, grensesnitt og komponent-nivå design for ***GUCRSS***.

### 3.1.1 Mål og målsettinger

GUCRSS skal automatisere oppgaven IT-tjenesten ved Høgskolen i Gjøvik har i dag når det gjelder opprettelse av Subversion repositorier.

### 3.1.2 Systemet i bruk

GUCRSS vil fungere slik at studenter og ansatte logger seg inn med sitt ansatt-brukernavn eller studentnummer. Brukeren får da listet opp alle repositoriene han tilhører. Er brukeren eier av ett repository kan han endre egenskaper og tilgangskontroll på gitt repository.

Brukeren kan også selv opprette repository på denne siden, så lenge kvoten for antall repositorier ikke er overskredet.

Dersom den innloggede brukeren har administrator rettigheter, vil han få tilgang til flere funksjoner enn en vanlig bruker, som for eksempel å liste alle repositorier som eksisterer, slette utdaterte repositorier og endre egenskaper og tilgang på alle repositorier.

### 3.1.3 Programvare kontekst

Programvaren er tenkt brukt på Høgskolen i Gjøvik for å forenkle IT-tjenestens hverdag. Vi ser for oss at programvaren kan ved små endringer benyttes på andre Høgskoler eller universitet, men også i andre sammenhenger.

### 3.1.4 Begrensninger

Det var ønskelig fra oppdragsgivers side at systemet skulle inneha muligheten for å støtte andre versjonsstyringskontroll-programvare basert på Apache, enn Subversion. Da vi kun er to personer som jobber med denne oppgaven, ser vi det blir for dårlig tid for å implementere dette på en tilfredsstillende måte.

## *3.2 Datadesign*

En beskrivelse av datastrukturer inkludert interne og globale datastrukturer.

### 3.2.1 Databasebeskrivelse

Database med konfigurasjon og eksterne brukere vil basere seg på MySQL. Denne databasen er godt utbredt og har støtte på mange ulike systemer. Autentisering av lokale brukere gjøres via den eksisterende LDAP-databasen Høgskolen i Gjøvik allerede kjører.

Databasen systemet benytter for å holde orden på de ulike repositoriene er bygget opp slik:

**Tabell: repository**
*Inneholder informasjon om de forskjellige repositoriene.*

| **Attributt:** | **Beskrivelse:** |
|---|---|
| shortdescription | Én linjers beskrivelse av repositoriet |
| qutoa | Kvoten på repositoriet (kan kun endres av administratorbruker) |
| createddate | Datoen repositoriet ble opprettet på |
| enddate | Datoen repositoriet kan slettes |
| path* | Stien repositoriet befinner seg på. (Denne er unik) |

**Tabell: repositorypath**
*Inneholder de ulike undermappene på gitt repository.*

| **Attributt:** | **Beskrivelse:** |
|---|---|
| path* | Stien repositoriet befinner seg på. (Denne er unik) |
| subpath | Stien innenfor repositoriet denne undermappen ligger på. (Denne er unik) |

**Tabell: repositoryowner**
*Inneholder informasjon om hvem som er eier av gitt repository.*

| **Attributt:** | **Beskrivelse:** |
|---|---|
| path* | Stien repositoriet befinner seg på. (Denne er unik) |
| uID | Bruker id til personen som er eier (denne er unik) |

**Tabell: repositoryacluser**
*Inneholder tilganginformasjon for gitt bruker på gitt repository.*

| **Attributt:** | **Beskrivelse:** |
|---|---|
| path* | Stien repositoriet befinner seg på. (Denne er unik) |
| subpath* | Stien innenfor repositoriet som tilgangsoppføringen gjelder for. |
| access | Tilgangen brukeren har på denne stien. (Ingen tilgang/lese/lese og skrive) |
| uID | Bruker id til personen som denne tilgangsoppføringen gjelder for (denne er unik) |

**Tabell: repositorygroup**
*Inneholder et unikt gruppenavn på gitt repository.*

| **Attributt:** | **Beskrivelse:** |
|---|---|
| path* | Stien repositoriet befinner seg på. (Denne er unik) |
| groupname | Gruppenavn. Denne er unik innenfor dette repositoriet. Samme gruppenavn kan forekomme på forskjellige repositorier. |

**Tabell: repositorygroupmember**
*Inneholder informasjon om hvem som er medlem av gitt repositorygruppe.*

| **Attributt:** | **Beskrivelse:** |
|---|---|
| path* | Stien repositoriet befinner seg på. (Denne er unik) |
| groupname* | Gruppenavn. Denne er unik innenfor dette repositoriet. Samme gruppenavn kan forekomme på forskjellige repositorier. |
| uID | Brukerid til personen som er gruppemedlem (denne er unik). |

**Tabell: repositoryaclgroup**
*Inneholder tilganginformasjon for gitt gruppe på gitt repository.*

| **Attributt:** | **Beskrivelse:** |
|---|---|
| path* | Stien repositoriet befinner seg på. (Denne er unik) |
| subpath* | Stien innenfor repositoriet som tilgangsoppføringen gjelder for. |
| access | Tilgangen gruppen har på denne stien. (Ingen tilgang/lese/lese og skrive) |
| groupname* | Gruppenavn på gruppen ssom tilgangsoppføringen gjelder for (denne er unik). |

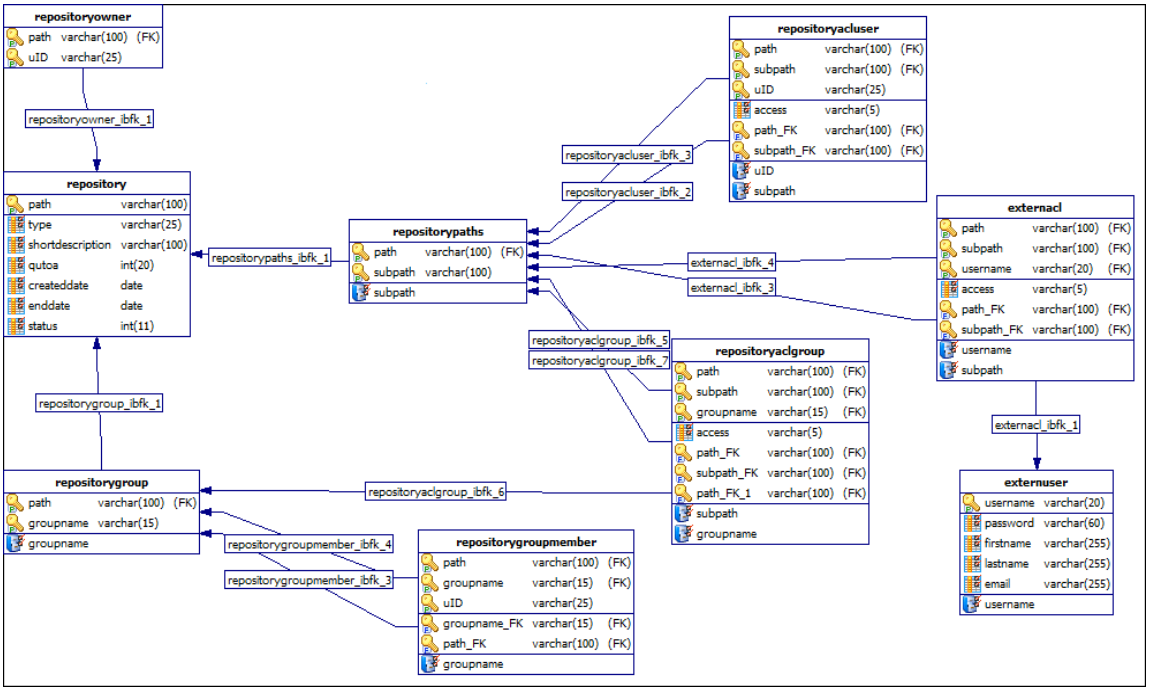
**Tabell: externuser**
*Inneholder brukerdata for brukere som ikke ligger i LDAP-databasen. Disse vil kun få tilgang til selve repositoriet, og ikke kontrollpanelet for repositoriet.*

| **Attributt:** | **Beskrivelse:** |
|---|---|
| username | Brukernavnet til den eksterne brukeren. |
| password | Den eksterne brukerens passord. (enveis-kryptert med PHPs MD5 funksjon) |
| firstname | Den eksterne brukerens fornavn. |
| lastname | Den eksterne brukerens etternavn. |
| email | Epost til den eksterne brukeren |

**Tabell: externacl**
*Inneholder tilganginformasjon for gitt ekstern bruker.*

| **Attributt:** | **Beskrivelse:** |
|---|---|
| path* | Stien repositoriet befinner seg på. (Denne er unik) |
| subpath* | Stien innenfor repositoriet som tilgangsoppføringen gjelder for. |
| access | Tilgangen den eksterne brukeren har på denne stien. (Ingen tilgang/lese/lese og skrive) |
| username* | Brukernavn på den eksterne brukeren som tilgangsoppføringen gjelder for (denne er unik). |

#### 3.2.1.1 Databasefigur




## *3.3 Systemarkitektur og komponentdesign*

En beskrivelse av systemets arkitektur presenteres i dette kapitelet.

### 3.3.1 Systemarkitektur

Vi har valgt å benytte oss av lag-modellen i dette prosjektet. Denne modellen deler systemet opp i forskjellige lag, der de forskjellige lagene vil tilby forskjellige tjenester. Dette er en modell som vil passe best til systemer der forskjellige lag i systemet har forskjellige krav til sikkerhet og tilgjengelighet, fordi det er lett å konfigurere krav til sikkerhet i de ulike lagene. Dette er også en modell der det er lett å gjøre forandringer.

Lag-modellen passer inkrement-metoden, da etterhvert som hvert enkelt lag utvikles, vil dette gjøres tilgjengelig for brukerne. Her benyttes et system i bunnen som grunnlag for de andre lagene. For eksempel LDAP-databasen til skolen må være operativ for at brukerne skal kunne logge seg inn å få tilgang til de funksjonene høyere opp i lag-inndelingen. Prosjektets lag-modell er presentert i figur 3-1.

### 3.3.1.1 Arkitektur diagram

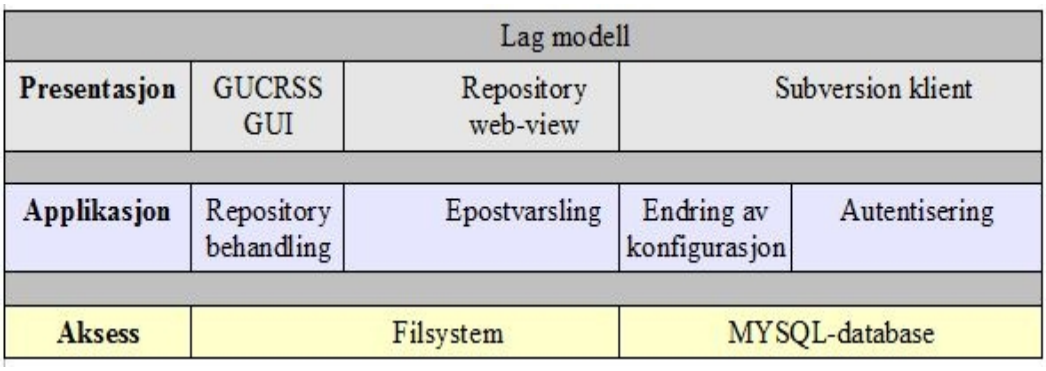



Fig. 3-1 – Lag-modellen

# 4. Implementering

I dette kapitlet vil vi ta for oss hvilket utviklingsmiljø, diverse programmer som er brukt og eksempler på de rammeverk som har blitt brukt.

## 4.1 Utviklingsmiljø

Ved start av prosjektet var utviklingsmiljøet allerede bestemt fra oppdragsgivers side. Hele systemet skulle være en PHP basert løsning og fungere på en CentOS 5.4 distribusjon. Apache-2.2 ble brukt som webserver siden den allerede var implementert i distribusjonen ved installering.

Databasen som løsningen bruker er en relasjonsdatabase. Valget falt etter samarbeid med oppdragsgiver på MySQL som database siden det er god støtte for denne i UNIX-systemer og i PHP. En funksjonalitet som også var med på å bestemme dette var at eksterne bruker utenfor HiG skal få tilgang til repositoriene. Subversion bruker gjennom en modul i Apache som heter «mod_auth_mysql» for MySQL autentisering.

## 4.2 Kodestil

Ved å basere kildekode på en kodestil vil gjøre det lettere for en programmerer å lese og forstå koden [2].

For å få en så oversiktlig og god lesbar kode som mulig har det gjennom hele utviklingen blitt brukt en innrykkbasert kodestil. Dette er en kodestil som er lett å lese og definere hvilken del av koden som hører innenfor en bestemt kodeblokk. Kodeutsnittet under viser et eksempel på hvordan denne kodestilen er blitt brukt:

```
function is_repo_owner($user, $path) {

        if ($_SESSION["is_admin"]) {
                return TRUE;
        }
        mysql_db_connect();
        $query = sprintf("SELECT uID FROM repositoryowner WHERE path='%s'",
        mysql_real_escape_string($path));

        $result = mysql_query($query);
        if($result) {
                while ($row = mysql_fetch_assoc($result)) {
                        // if given user is listed as owner on the given repository?
                        if ($row['uID'] == $user) {
                                return true;
                        }
                }
        }
        else {
                // mysql error
```

```
                return false;
        }
        return false;
}
```

Navngiving av funksjoner er gjort på den måten som vist under slik at navnet gjenspeiler oppgaven til funksjonen.

```
function delete_repository($repopath) {

        $error = FALSE;
        global $rsspath;

        $filename = str_replace("/", ".", $repopath);
```

Slett gitt repository medsendt som variabel.

```
function is_repo_owner($user, $path) {

        if ($_SESSION["is_admin"]) {
                return TRUE;
```

Sjekke om gitte bruker innlogget er eier av et repository.

Globale variable er lagt i egen PHP fil og er navngitt etter hvilken funksjonalitet de skal brukes til. Eksempler følger under:

```
// mysql host
$mysqlconfig['host'] = 'localhost';

// mysql database
$mysqlconfig['database'] = 'gucrss';

// LDAP-protocol (ldap, ldaps)
$ldapconfig['protocol'] = 'ldaps://';

// Domain for LDAP
$ldapconfig['basedn'] = 'dc=hig,dc=no';

// Path to the folder with repository plugins
$rsspath['pluginpath'] = "plugins/";

// Path for external users. Ex: http://repository.example.com/<external>/myrepository
//$rsspath['external'] = 'external';
$rsspath['external'] = 'external';
```

## 4.3 Programvare

### 4.3.1 Programmer brukt ved utvikling

Brukt ved koding:

| Program | Formål |
|---|---|
| Notepad++ | Skriving av PHP kode |
| Firebug | Tilleggsprogram til Firefox for feilsøking av kode i nettleser |

### 4.3.2 Programmer brukt til andre formål

| Program | Formål |
|---|---|
| Microsoft Office Project | Utvikling av Gant-skjema |
| Open Office Writer | Skriving av rapport |
| Edraw | Utvikling av Use case diagram |
| Microsoft Office Visio | Utvikling av klassediagram |
| MicroOLAP Database Designer for MySQL | Utvikling av databasemodell |
| Subversion (TortoiseSVN) | Versjonskontroll |

## 4.4 Versjonskontroll

Når det kom til det å holde kontrollen på alle kildekodefiler og alt annet som hørte oppgaven vår til så tok vi å benyttet oss av skolens tilbud av Subversion repository. For å enkelt få tilgang til dette repository så brukte vi ***GUI***-programmet TortoiseSVN.

Siden vi også skulle utvikle et system som skulle gjøre det lettere for studenter og ansatte å opprette repositorier falt dette seg naturlig. Det var også en hjelp for oss for å forstå mer av funksjonen bak Subversion.

## 4.5 Webapplikasjonen

I denne delen vil vi gå nærmere inn på enkelte funksjonaliteter i systemet, kodeeksempler og løsninger.

### 4.5.1 Innlogging


*Fig. 4-1 Innlogging*

Figur 4-1 viser skjermbildet som brukeren vil få presentert for å få logget inn på repository administrasjonen. Det er bare brukere som er registrert i HiG's LDAP-database som kan logge seg inn her. PHP har egne innebygde LDAP-funksjoner for å koble seg opp, autentisere seg for serveren, søke og hente ut data.

```
if(!($ds=gucldapconnect($ldapconfig['host'], $ldapconfig['readusr'], $ldapconfig['readpass']))) {
                // host or ldap readuser is wrong
                return FALSE;
        }
        $sr = ldap_search($ds, $ldapconfig['basedn'], "sAMAccountName=".$userid);
        $entries = ldap_get_entries($ds, $sr); // fetch result

        if (ldap_count_entries($ds,$sr) != 1) {
                // User don't exists
                return FALSE;
        }
        @ldap_close($ds);

        if(($ds=gucldapconnect($ldapconfig['host'], $entries[0]['dn'], $password))) {
```

```
                // Login OK!
                return TRUE;
        }
        else {
                // Wrong password
                return FALSE;
        }
}

function gucldapconnect($servers, $userdn, $password) {
        global $ldapconfig; // Global ldapconfig to fetch config-parameters
        $server = explode(";", $servers); // Split server parameters if more than one

        foreach($server as $host) {
                // try to connect all servers
                if(!($ds=@ldap_connect($ldapconfig['protocol'].$host.':'.$ldapconfig['port']))) { continue; }
                // try to bind to all servers
                if(!($r=@ldap_bind($ds, $userdn, $password))) { continue; }
                // return resource
                return $ds;
        }
        return FALSE;
}
```

I kodeutsnittet ovenfor er hovedessensen i autentiseringen ved innlogging.

*ldap_connect* åpner koblingen opp mot LDAP databasen. Tar parameterne(hvilken protokoll som skal brukes, adressen til serveren og hvilken port som skal brukes). Det brukes ldaps (LDAP over ***SSL***) som protokoll for å få en sikker kommunikasjon. All kommunikasjon her foregår kryptert.

For at tilkobling via SSL mot LDAP-serveren måtte vi bruke et sertifikat som vi fikk overlevert fra oppdragsgiver. Dette sertifikatet kommer vi nærmere inn på i kapitel 6.

*ldap_bind* brukes for å få lov til å sende data om autentiseringen [1]. Den bruker parameterne fra *ldap_connect* og i tillegg brukernavn og passord for en LDAP lese-bruker for å se om brukeren som logger seg inn finnes. Finnes den brukes da denne brukerens brukernavn og passord for å sette denne som en gyldig bruker.

*ldap_search* søker etter den gitte bruker som logger seg inn. Bruker parameterne fra *ldap_connect* samt hoveddomenet til LDAP databasen og brukerID for brukeren som prøver å logge seg inn.

Både *ldap_get_entries* *og* *ldap_count_entries* bruker parameterne fra *ldap_connect* og *ldap_bind* for å hente ut data. Get bruker dem for å hente ut data, mens count verifiserer at brukeren finnes.

Det var også et krav fra oppdragsgiver at dersom ***cookies*** ikke er påslått i nettleseren måtte brukeren informeres om at dette var nødvendig [9]. Denne sjekken vises i kodesnutten nedenfor:

```
// Set a cookie to check if cookies is enabled
if(!isset($_GET['redirected'])) {
    setcookie('gucrsscookiecheck', 'cookiecheck', time() + 360);
    header('location:'.$_SERVER['PHP_SELF'].'?redirected=1');
}

// Is our cookie set?
if(isset($_GET['redirected']) and $_GET['redirected']==1) {
    if(!isset($_COOKIE['gucrsscookiecheck'])) {
        $no_cookie_message = _('Cookies are not enabled. You must enable cookies to use theese pages.');
    }
            }
```

### 4.5.2 Opprettelse

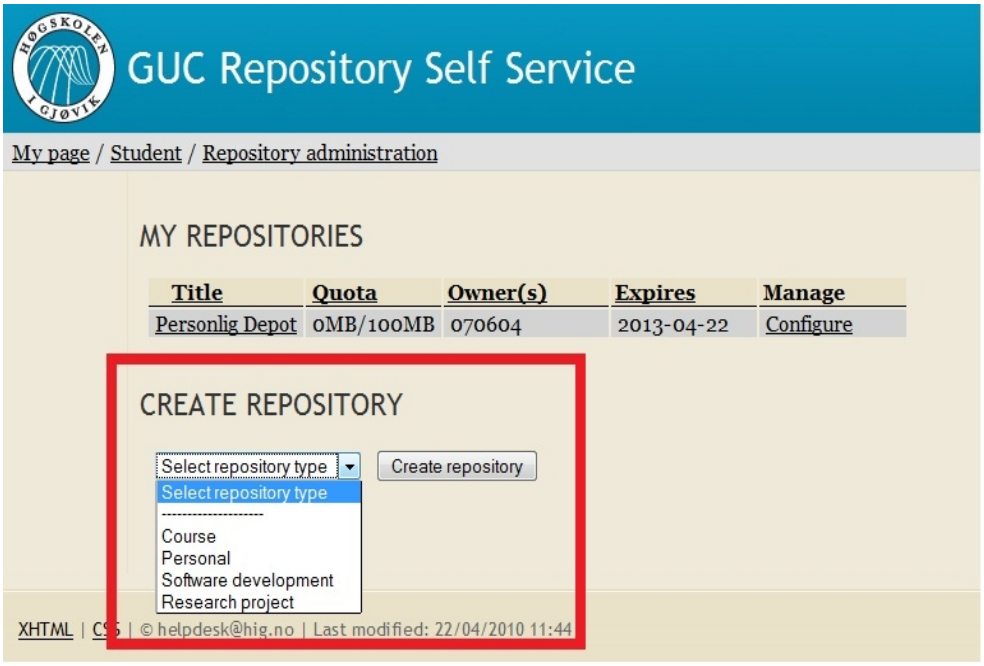



*Fig. 4-2 Opprette nytt repository*

Når man skal opprette et nytt repository vil det være flere ulike typer å velge mellom. Disse valgene vil komme i form av en nedtrekksmeny (Fig. 4-2). Hvilke valg som kommer i denne nedtrekksmenyen styres av «plugins». Dette vil bli nærmere forklart i kapitel 4.6.

### 4.5.3 My repositories


*Fig. 4-3 Liste mine repositoies*

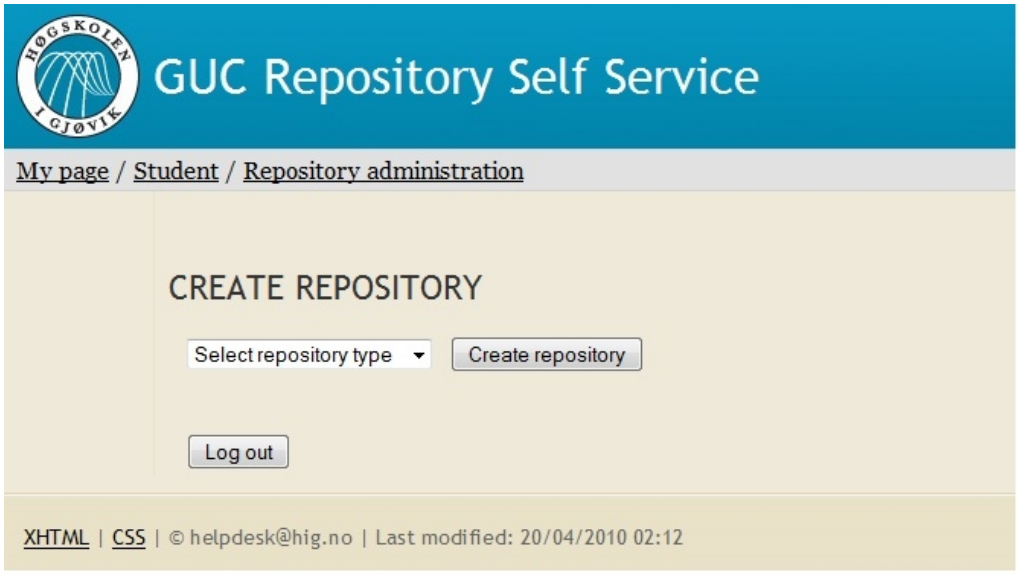



*Fig. 4-4 Ingen repositorier*

Når brukeren har logget inn vil de repositorier som denne personen er medlem av. Som vist i figur 4-3. Om brukeren står som eier av repositoriet vil det under manage-kolonnen i oversikten dukke opp en trykkbar link som gjør det mulig for brukeren å endre konfigurasjonen til repositoriet. Hvis brukeren ikke er eier vil ikke denne linken dukke opp.

Om brukeren ikke er medlem av noen repository vil det som vist i figur 4-4 bare vises muligheten for å opprette et nytt repository.

Når en administrator logger seg inn vil det bli vist det samme som for en vanlig bruker(figur 4-3). Men i tillegg til de repositorier som administrator er medlem av vil også alle

repositorier som er opprettet vises. Antall repositorier som er gått ut på dato vil også dukke opp i dette skjermbilde, se figur 4-5. Administrator har også mulighet til å endre konfigurasjon på alle opprettede repositorier.

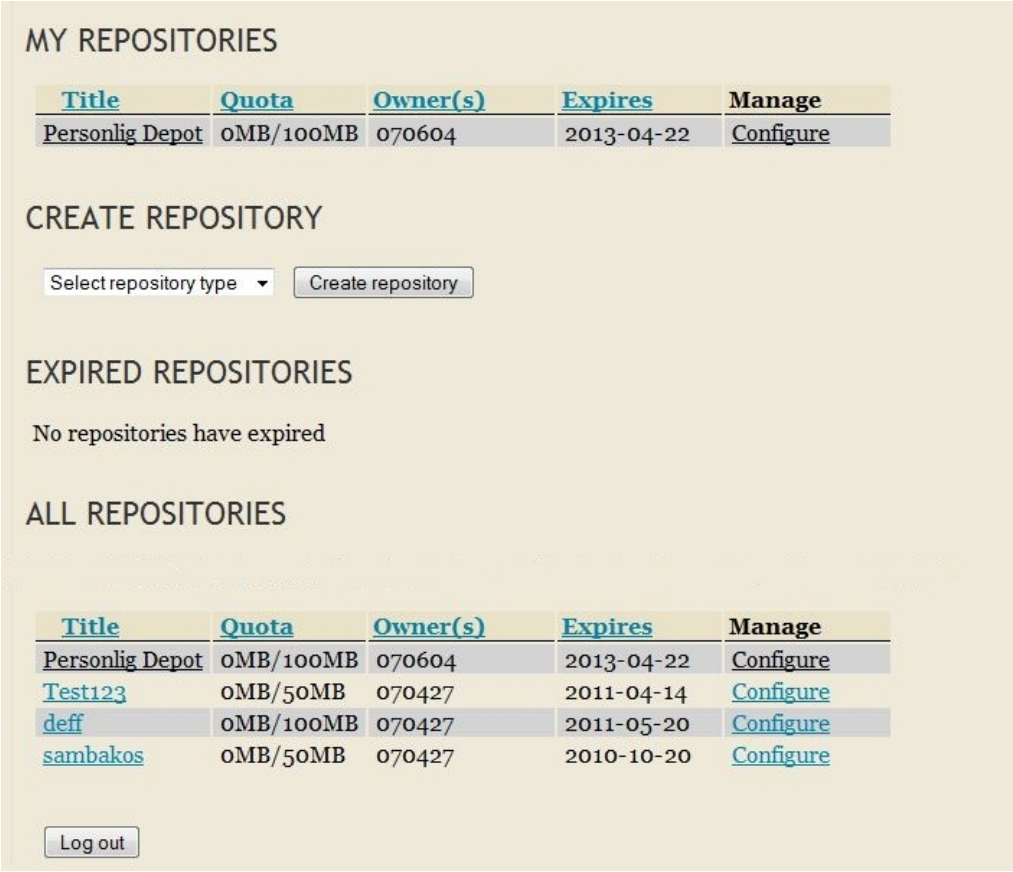



Fig. 4-5 Repositoryoversikt for administrator.

## 4.5.4 Endre konfigurasjon

Konfigurasjonen styrer alt av levetid, brukere og tilgang for hvert enkelt repository. For å holde styr på all denne dataen blir alt lagret i systemets MySQL-database. Ut fra denne databasen blir kofigurasjonsfilene bygget opp.

**Eiere:**


*Fig. 4-6 Eiere*

Den som oppretter et repository blir automatisk satt som eier(figur 4-6) og kan om det trengs eller er ønskelig å sette andre brukere som eiere. Koden nedenfor viser hvordan listen i figur 4-6 bygges opp:

```
echo "<table>\n";
echo "<tr>\n";
echo "<th>Owner</th>\n";
echo "<th>Action</th>\n";
echo "</tr>\n";
$counter = 0; // counter for background color
while ($row = mysql_fetch_assoc($result)) {
        echo "<form action=\"editrepository.php?path=".$_GET['path']."#owners\" method=\"post\">\n";
        if ($counter % 2 == 1) {
                echo "<tr style=\"background-color: lightgray\">\n";
        }
        else {
                echo "<tr>\n";
        }
        echo "<td>".$row['uid']."</td>\n";
        echo "<td>";
        if ($row['uid'] != $_SESSION["valid_user"]) {
                echo "<input type=\"hidden\" name=\"RemoveOwnerUid\" value=\"".$row['uid']."\" />";
                echo "<input type=\"submit\" id=\"RemoveOwner\" name=\"RemoveOwner\"
value=\""._('Remove')."\" />";
        }
        echo "</td>\n";
        echo "</tr>\n";
        echo "</form>";
        $counter++;
}
```

Funksjonen for å legge til eiere vil bli nærmere forklart i kapitel 4.6.3, i figur 4-6 vises også skjemaet for å søke etter brukere å legge til. Dette er koden for skjemaet:

```
echo "<form action=\"editrepository.php?path=".$_GET['path']."#owners\" method=\"post\">\n";
echo "<tr valign=\"bottom\">\n";
echo "<td><input type=\"text\" id=\"AddOwnerUserId\" name=\"AddOwnerUserId\" /></td><td><input
type=\"submit\" name=\"AddNewOwner\" value=\""._('Add Owner')."\"></td>\n";
echo "</tr>\n";
echo "</table>\n";
echo "</form>\n";
```

**Innstillinger:**
For endringer på innstillinger på repositoriet vil eiere få i skjermbildet som vist i figur 4-7. Kun tillatelse til å endre tittel og sluttdato for repositoriet.
Administrator vil i tillegg (se figur 4-8) også kunne endre hvor stor lagringskvote hvert enkelt repository skal få tildelt utover det som standard blir satt under opprettelse.

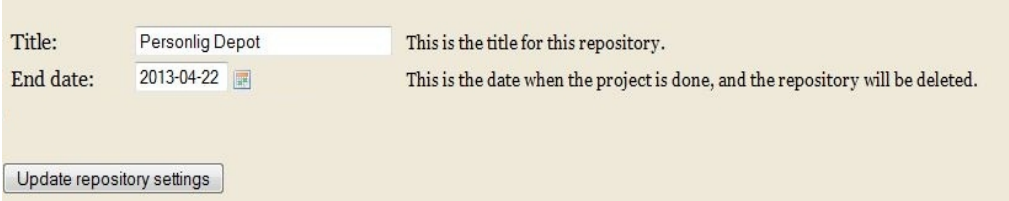



*Fig. 4-7 Endring av innstillinger for eier*

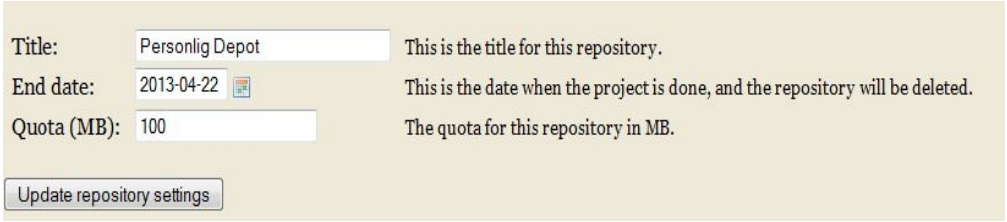



*Fig. 4-8 Endring av innstillinger for administrator*

### 4.5.5 Informasjon til bruker

Når brukeren gjør endringer, gjør noe feil, skal gjøres oppmerksom på noe eller lignende kommer det opp en meldingsboks [10] med informasjon om hva det gjelder. Ut i fra hva slags type melding det er har disse boksene forskjellige farger og symboler. Eksempler på disse meldingen ser du i figur 4-9.

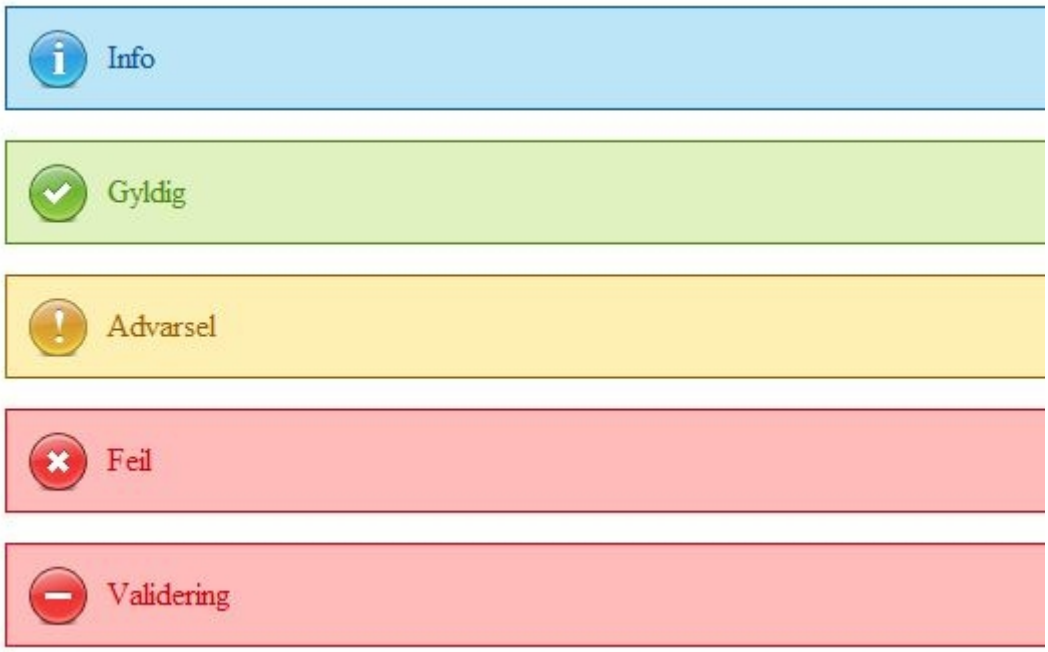



Fig. 4-9 Informasjonsbokser til bruker.

– Info brukes til generell informasjon ut mot brukeren. Eksempelvis ved endringer av et repository så vises den og sier hvilket repository som det endres på.

– Gyldig vises når operasjoner/endringer er vellykket. Eksempelvis ved vellykkede endringer.

– Advarsel vises når brukeren gjør noe som han må være helt sikker på at han vil gjøre. Eksempelvis å slette repositoriet.

– Feil vises når en feil har oppstått. Eksempelvis ved at brukeren oppretter et repository som allerede opprettet eller ved feil inntasting av brukernavn og/eller passord.

– Validering vises når ikke alle felter er fylt ut riktig og ikke kan gå videre. Eksempelvis hvis det ved endringer av informasjon på et repository ikke er fylt ut.

## *4.6 Rammeverk, standarder og konfigurasjon*

### 4.6.1 Repository autentisering

For at brukerne skal få tilgang til sine repositorier så må det være en konfigurasjonsfil som sier hva slags type autentisering som skal brukes og hvor dette ligger. I tillegg må det også være en (***ACL***) aksessliste fil som sier hvilke brukere som har tilgang til hvor i repositoriet og hva slags tilgang de har. Disse blir opprettet under (*/etc/httpd/conf.d/gucrss*)**.** For at disse filene skal bli tatt i bruk etter opprettelse blir denne linjen lagt inn i (*/etc/httpd/conf/httpd.conf)* , **(Include conf.d/*.conf).** Dette medfører da at vår egen konfigurasjonsfil (*/etc/httpd/conf.d/gucrss.conf*) som inneholder (**Include conf.d/gucrss/*.conf**) blir tatt med i konfigurasjonen.

Dette medfører at når Apache restartes, så blir de nyopprettede filene del av webserverkonfigurasjonen. Det blir opprettet en konfigurasjonsfil for hvert repository med konfigurasjonen for både LDAP og MySQL.

Filene for konfigurasjon og aksess til hvert enkelt repository blir opprettet henholdsvis i (*/etc/httpd/conf.d/gucrss/*) og (*/etc/httpd/conf.d/gucrss/acl*). For å få en bedre forståelse av hva de ulike elementene i filene har som oppgave se kapitlene 4.6.1.1 og 4.6.1.2.

For å kunne bruke LDAP autentisering mot SVN må «mod_authnz_ldap» [8]. Dette er en Apache modul som brukes for å kunne bruke LDAP som autentiseringsmekanisme.

Det må også for å kunne bruke HiG's LDAP-server ha høgskolens ***sertifikat*** på serveren. Det blir i Subversion sin konfigurasjonsfil lagt inn disse to linjene som sier hvor sertifikatet ligger:

**LDAPVerifyServerCert Off**

**LDAPTrustedGlobalCert CA_BASE64 /etc/openldap/cacerts/inu.pem**

Disse endringene og installasjonen blir gjort når RPM-pakken kjøres.

#### 4.6.1.1 Repository konfigurasjonsfiler.

**LDAP**

```
// Config for GUC users stored in Ldap
 "<Location /".$path.">\n";
= "    DAV svn\n";
= "    SVNPath ".$rsspath['filesystem'].$path."\n";
= "    SVNReposName \"".$row['shortdescription']."\"\n";
= "    SVNIndexXSLT /svnindex.xsl\n";
= "    SetOutputFilter xslt\n";
= "    SSLRequireSSL\n";
= "    AuthzSVNAccessFile ".$rsspath['svnaclfile'].$filename.".acl\n";
= "    Satisfy All\n";
= "    Require valid-user\n";
= "    AuthType Basic\n";
= "    AuthBasicProvider ldap\n";
= "    AuthName \"".$rssmsg['logintxt']."\"\n";
= "    AuthLDAPURL \"".$ldapconfig['authldapurl']."\"\n";
= "    AuthLDAPBindDN \"".$ldapconfig['readusr']."\"\n";
= "    AuthLDAPBindPassword ".$ldapconfig['readpass']."\n";
= "</Location>\n";
```

**MySQL**

```
 // Config for non-hig users stored in MySQL-DB
"\n<Location /".$rsspath['external']."/".$path.">\n";
"    DAV svn\n";
"    SVNPath ".$rsspath['filesystem'].$path."\n";
"    SVNReposName \"".$row['shortdescription']."\"\n";
"    SVNIndexXSLT /svnindex.xsl\n";
"    SetOutputFilter xslt\n";
"    SSLRequireSSL\n";
"    AuthzSVNAccessFile ".$rsspath['svnaclfile'].$rsspath['external']."."$filename.".acl\n";
"    Satisfy All\n";
"    Require valid-user\n";
"    AuthType Basic\n";
"    AuthName \"".$rssmsg['logintxt']."\"\n";
"    AuthMySQLHost ".$mysqlconfig['host']."\n";
"    AuthMySQLUser ".$mysqlconfig['user']."\n";
"    AuthMySQLPassword ".$mysqlconfig['password']."\n";
"    AuthMYSQLEnable On\n";
"    AuthMySQLAuthoritative On\n";
"    AuthMySQLDB ".$mysqlconfig['database']."\n";
"    AuthMySQLUserTable ".$mysqlconfig['usertable']."\n";
"    AuthMySQLNameField ".$mysqlconfig['usernamefield']."\n";
"    AuthMySQLPasswordField ".$mysqlconfig['passwordfield']."\n";
"    AuthMySQLNoPasswd Off\n";
"    AuthMySQLPwEncryption md5\n";
"</Location>\n";
```

**Felles elementer for LDAP og MySQL:**

**Location:** Sier stien som brukes i ***URL***'en for å få tilgang til det gitte repository.

**DAV:** Sier hvilken Apache-modul som skal aktiveres. I dette tilfelle «mod_dav_svn».

***SVNpath:*** Hvor i filsystemet repositoriet ligger.

**SVNReposName:** Tittelen på repositoriet.

**SVNIndexXSLT:** Hvilket stilark som skal brukes for utseende som presentasjonen av strukturen av repositoriet får i nettleseren.

**SetOutputFilter:** Hvilket filter serveren skal bruke for å behandle data før det blir sendt til klienten.

**SSLRequireSSL:** Krever SSL kobling for å beskytte passord.

**AuthSVNAcessFile:** Hvilken fil som sier hvem som har hvilke tilgangsrettigheter på repositoriet.

**Satisfy All:** At alle tilgangsrestriksjoner skal oppfylles. Det vil si gyldig kobling mot autentiseringsserver og at gyldig brukernavn og passord er tastet inn.

**Require valid-user:** Må være en gyldig bruker som kan autentisere seg.

**AuthType Basic:** Metoden som blir brukt for autentiseringen. Basic er den metoden som er standard men man bør være klar over at denne sender passord ukryptert. Siden vi også har **SSLRequireSSL** så blir allikevel passordet beskyttet.

**AuthName:** Tekst som dukker opp i boksen som krever brukernavn og passord når du vil åpne repositoriet i en nettleser.

**Elementer spesifikk for LDAP:**

**AuthBasicProvider:** Hvilken type autentisering som skal brukes. I dette tilfellet LDAP.

**AuthLDAPURL:** Adressen til LDAP-serveren pluss de søkeparametere for å autentisere den som prøver å få tilgang til repositoriet.

Syntaks: `ldaps://host:port/basedn?attribute?scope?filter`

protokoll://adressen:port/baseDN det skal starte søket fra?hvilket attributt det skal søkes etter?omfang av søket(sub)?ldapfilter

**AuthLDAPBindDN:** Lese-brukeren som får tilgang til å søke i LDAP basen.

**AuthLDAPBindPassword:** Passordet til lese-brukeren.

**Elementer spesifikk for MySQL:**

**AuthMySQLHost:** Tjeneren for MySQL databasen.

**AuthMySQLUser:** Brukeren som har rettigheter til å søke i MySQL databasen.

**AuthMySQLPassword: AuthMySQLUser** sitt passord.

**AuthMySQLEnable:** Om autentiseringen skal gjøres med MySQL autoriserings modulen til Apache.

**AuthMySQLAuthoriative:** Om andre Apache autoriserings moduler skal tas i bruk for denne plasseringen om MySQL ikke klarer det. Siden den hos oss er satt til On vil bare MySQL bli brukt.

**AuthMySQLDB:** Databasen som inneholder autoriseringsdata.

**AuthMySQLUserTable:** Tabellen i **AuthMySQLDB** som inneholder brukernavn og passord til autentiseringen.

**AuthMySQLNameField:** Raden i **AuthMySQLUserTable** som inneholder brukernavnet.

**AuthMySQLPasswordField:** Raden i **AuthMySQLUserTable** som inneholder passordet.

**AuthMySQLNoPasswd:** Om passord skal brukes eller ikke ved autoriseringen. Er den satt til On vil hvilket som helst passord fungere om brukeren finnes i databasen. Er den satt til Off som i vår konfigurasjon må brukernavn og passord være lagret i databasen.

**AuthMySQLPwEncryption:** Typen kryptering som er brukt til å kryptere passordet til brukerne lagret i databasen.

#### 4.6.1.2 ACL

**LDAP: (***/etc/httpd/conf.d/gucrss/acl/type.navn.acl)*

**[groups]**

**owners = brukerid**

**utvikling = ludvig, anders, knut**

**[/]**

**@owners = rw**

**@utvikling = r**

**roar = rw**

I aksessliste filen for interne HiG-brukere vil det være mulig å lage grupper med brukere slik at det vil bli lettere å gruppere de som skal ha like rettigheter på samme område. Det vil også være mulighet for å legge til enkeltbrukere. Når gruppene og brukerne er opprettet vil de bli lagt under riktig undermappe i aksessliste filen. Alt dette skjer i administrasjonen av hvert enkelt repository via webgrensesnittet.

**MySQL: (***/etc/httpd/conf.d/gucrss/acl/external.type.navn.acl)*

**[/]**

**brukernavn = rw**

**brukernavn3 = rw**

**[/oppgave]**

**brukernavn2 = r**

I aksessliste filen for eksterne brukere er det bare en og en bruker som blir registrert av gangen. Derfor blir bare brukernavnene med rettigheter lagt inn under riktig undermappe i repositoriet.

### 4.6.2 Plugins

Etter ønske fra oppdragsgiver er det laget ett rammeverk for plugins. Rammeverket er designet slik at man enkelt kan legge til repository typer ved å legge den ferdige pluginen i plugin-mappen, uten å måtte oppdatere de øvrige komponentene i systemet.

Logikken bak pluginsene er veldig enkel, men effektiv. Hvis det tas utgangspunkt i fag/course-pluginen, kan vi ta en nærmere kikk på oppbygningen av hvordan plugins fungerer. La oss starte med navnet. Navnet er bygget opp på denne måten: *plugin_<type>_<etikett på plugin>.php*. I course-plugin blir navnet da: *plugin_rt_course.php*.

Type *rt* står for «repository type». Dette feltet er lagt til for fremtidig utvidelse av plugins, slik at man kan ha f.eks. Repository system plugins. Alle systemets plugins er av typen *rt*.

Innholdet i pluginen er bygget opp av tre funksjoner og noen justerbare variabler. I course pluginen heter disse funksjonene

- plugin_rt_course_getinputtypes($user)
- plugin_rt_course_getpathfrominput($user, $inputArray)
- plugin_rt_course_userallowed($user)

Som man kan se, er navnet på funksjonene bygget opp slik: *<navnet på plugin>_<funksjon>*. Slik skilles de ulike pluginsene fra hverandre.

Funksjonen *getinputtypes* tar med brukerID som parameter og returnerer en array med input-feltene en bruker får opp når han skal opprette ett repository. Dette kan være kortnavn, fag og lignende. Tanken er at feltene funksjonen returnerer skal være komponenter som brukes i stien til repositoriet.

Funksjonen *getpathfrominput* genererer stien til repositoriet som skal opprettes. Denne stien blir brukt i hyperlenken til GUCRSS og til stien på det lokale filsystemet. Funksjonen tar brukerID og en input-array som parametere. Input-arrayen inneholder data brukeren har matet feltene som er generert i funksjonen nevnt over. Når funksjonen har behandlet dataene den har fått fra sine to parametere, returnerer den stien.

Funksjonen *userallowed* tar brukerID som parameter og returnerer *true* eller *false*. Dersom funksjonen returnerer *true*, har brukeren lov å opprette denne type repository. I motsatt fall har brukeren ikke lov, og muligheten til å velge denne type repository faller bort.

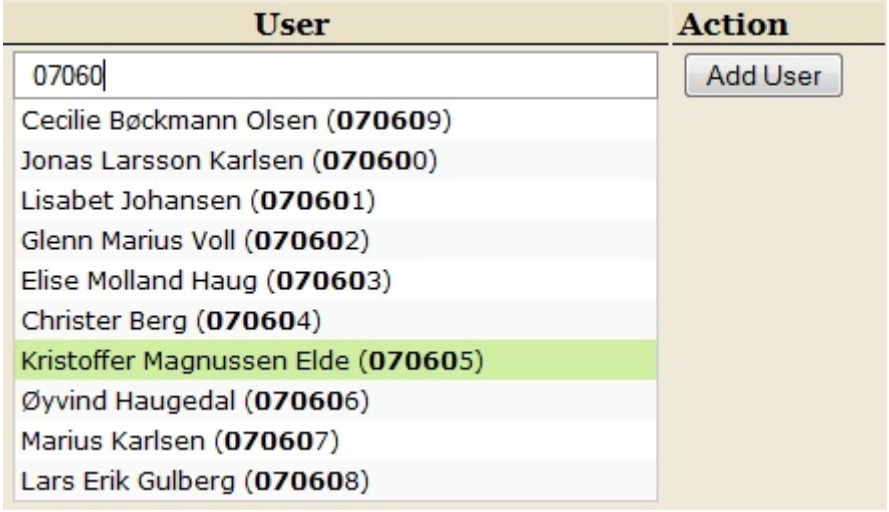
Variablene en plugin inneholder er informasjon om hver enkelt plugin, som brukeren får opp. Igjen ser man bruken av <navnet på plugin>_<variabelnavn>:

- $plugin_rt_course_description_short – dette er navnet som dukker opp i listen når en bruker skal opprette repository. I course-plugin vil det stå «Course».
- $plugin_rt_course_description_long – denne beskrivelsen forteller hva denne type repository skal brukes til. I course-plugin vil det stå noe som «denne repository-typen brukes til prosjekter knyttet opp mot ett fag».
- $plugin_rt_course_path – dette er navnet på plugintypen, i ett format som passer til bruk i hyperlenker. I course-plugin vil det stå «course».
- $plugin_rt_course_months_alive – denne variabelen inneholder antall måneder denne type repository skal være aktiv som standard. Ettersom fag normalt går over et halvt år, vil innholdet i course-plugin være «6».

### 4.6.3 Søke/legge til brukere

Ettersom det er lagt vekt på brukervennlighet gjennom prosjektet, har det blitt implementert noen web 2.0 funksjoner ved hjelp av javascript og XMLHttpRequest-objekter (AJAX). Det finnes ulike javascript-bibliotek man kan benytte for å oppnå de ønskede effektene. Det har blitt utforsket og vurdert ulike bibliotek deriblant Dojo, MooTools og Prototype, men den mest utbredte er den valget falt på; ***jQuery***. Grunnen til at valget falt på jQuery er for det første at den er meget veldokumentert. Men tilgjengelige tilleggspakker er også mye større for jQuery enn de øvrige. Over 30% av de 10.000 mest besøkte nettstedene på Internett i dag benytter seg av jQuery [3]. Av de mest kjente sidene som benytter et eller annet kjent javascript-bibliotek, bruker 69% jQuery [4].

En av tilleggspakkene til jQuery er «Tokenizing Autocomplete Text Entry» skrevet av James Smith [5]. Denne innehar den funksjonaliteten som vi på forhånd hadde sett det var ønskelig å tilby til våre brukere. Dessverre er denne meget dårlig dokumentert og skaperen av pakken er opptatt med andre prosjekter og tilbyr derfor ingen support på denne. Derfor har vi måttet sette oss grundig inn i koden på denne og rette opp ulike småfeil som originalutgaven hadde.



Funksjonaliteten for å søke opp og legge til brukere var i tidlig betaversjon meget tungvinn, og man måtte igjennom flere steg for å finne og velge riktig person. For hvert steg ble siden lastet på nytt, eller så ble brukeren sendt til en ny side. Med jQuery og James Smiths tilleggspakke ble denne prosessen kortet ned betraktelig og gav et mer intuitivt brukergrensesnitt. Brukeren starter å skrive en brukerID eller ett navn, så kommer automatisk forslag på de mest relevante treffene opp. En liten detalj som var implementert i originalkoden var at man kunne velge flere personer på engang. Tilleggspakken fungerer slik at resultater fra søk blir lagret i ett buffer lokalt på datamaskinen. Dette medfører raskere søk for brukeren når han søker etter flere personer på engang, og databasen slipper å bli stresset. Ettersom resultatene blir hentet rett fra HiG's LDAP-database ble forslagslisten etterhvert lang, så det ble valgt å begrense resultatet til de 10 mest relevante treffene.


Ett av problemene vi måtte få orden på var at det var opprinnelig mulig å velge samme person mer enn en gang. Vi forandret på koden slik at man bare kan velge samme person én gang på samme søk. Det var også problemer når man valgte person med musen i enkelte nettlesere. Dette fungerer nå på den måten som var ønskelig.

### 4.6.4 Sluttdato/kalender

En annen tilleggspakke til jQuery som ble benyttet var «date picker plug-in» laget av Kelvin Luck [6]. Dette er en kalender som spretter opp når man skal skrive inn datoen repositoriet kan fjernes.


Denne tilleggspakken var så veldokumentert at man kun trengte å inkludere pakken og sette de riktige opsjonene, så fungerte kalenderen på den måten som var ønskelig.

# 5. Testing

Som i alle andre utviklingsprosjekter er det veldig viktig å teste applikasjonen for å avdekke feil og/eller mangler. For å få en så omfattende og realistisk test av systemet som mulig, holder det ikke at vi som utviklere er de eneste testerne. Derfor er det besluttet at hvis det tidsmessig er mulig, skal det utføres virkelige brukertester.

## *5.1 Egentest underveis*

Gjennom hele utviklingsperioden har applikasjonen blitt testet etterhvert som ny funksjonalitet er lagt til. Åpenbare feil er blitt rettet undervegs.

Vi har benyttet oss av Firebug, en enkel og rask debugger som fåes som plugin til nettleseren Firefox. Denne gjør det svært enkelt å se igjennom HTML-koden PHP genererer, og forenkler arbeidet med å feilsøke i kildekoden.

## *5.2 Første beta-utgivelse*

Fredag før påske, 26.03.2010, leverte vi vår første betaversjon til oppdragsgiver. Vi hadde jobbet lenge med denne innleveringen og var veldig spent på tilbakemeldingen. Dessverre gikk ikke installasjonen for oppdragsgiver så feilfritt som vi hadde håpet. Maskinen vi hadde testet installasjonen på, hadde programmer liggende inne fra utviklingsperioden. Disse programmene var ikke i listen over avhengigheter til GUCRSS. Derfor oppdaget vi ikke dette selv før vi leverte installasjonspakken til oppdragsgiver. Dette medførte at en ny utgivelse av versjon 1 kom ut veldig raskt, denne gangen med korrekt installasjon.

Da vi klarte målet med å levere første beta-utgivelse før påske, kunne vi ta oss tid til å utføre brukertester.

## *5.3 Faktiske brukertester*

Vår veileder, Erik Hjelmås, ga oss en liste over ansatte som benytter seg av Subversion til prosjekter. Noen av disse opplyste oss om andre ansatte som var flittige brukere av Subversion. På denne måten ble det veldig enkelt for oss å finne ulike ansatte som var potensielle brukere av GUCRSS. Av medstudenter benyttet vi oss av personer som allerede hadde kjennskap til Subversion.
Testene ble utført ved at testbrukeren fikk en liste med oppgaver (se vedlegg F) som han skulle fullføre, deretter ble det notert hvordan testbrukeren avvek fra tenkt bruksmønster.

Fordelen med å benytte forskjellige folk til testing er at alle bruker programmet annerledes og på måter vi ikke forventer. Dette resulterte i at det ble avdekket flere merkelige mangler og feil. Vi ble oppmerksomme på hvordan brukerne oppfattet systemet og måtte forandre litt på designet.

Noe som gikk igjen var at testbrukerne gikk rett inn på webvisningen av repositoriet etter opprettelse. Vi vurderte om vi skulle gjøre URL'en til repositoriet ikke klikkbar. Det viste seg at samtlige hadde lite eller ingen problem med å komme til administrasjonssiden ved å benytte menylinken under ***headeren***. Likevel ble det besluttet å ta bort muligheten til å trykke på linken, da det er lite hensiktsmessig å komme inn på et tomt repository. Link tilbake til administrasjonssiden blir gjort om til en knapp for bedre synlighet.

En annen ting som gikk igjen var at testbrukerne var usikre på om man måtte bekrefte brukere etter disse var søkt opp. Derfor måtte knappen for å bekrefte brukere lagt til bli mer synlig.

## 5.4 Oppsummering

Endringene det vil fokuseres på før innlevering er oppsummeringssiden etter vellykket opprettelse av repository og få til en enklere oversikt av konfigurasjonssiden til repositoriene. I våre øyne er dette viktige endringer som vil forbedre brukeropplevelsen betraktelig og vi tror dette er «pynten på kaka» som vil gi GUCRSS en stor suksess.

# 6. Realisering

Seksjonene i dette kapitlet omhandler de driftsmessige aspektene rundt realiseringen av prosjektet.

Kapitlet forklarer hvordan GUCRSS pakken er bygget opp og hvordan vi gikk frem for å realisere dette.

## 6.1 RPM-pakken

En RPM-pakke er en måte å samle ett program, dette kan være en eller flere kjørbare binære filer. Man har også SRPM-pakker som inneholder kildekoden til programmet slik at dette kan kompileres selv. Dette er en god måte å forenkle installasjon og distribusjon. Pakken kan brukes alene, eller knyttes opp mot en pakkebehandler som koordinerer installasjon av programvare, dokumentasjon, grafikk og annen informasjon.

For å tilegne oss kunnskap rundt bygging av RPM-pakke ble mye tid brukt på å studere manualen utgitt av GURULABS [11].

Det er en rekke egenskaper RPM-pakker innehar. De viktigste egenskapene er versjon, arkitektur, beskrivelse og endringslogg. RPM-pakken inneholder alle filer GUCRSS avhenger av for å fungere og en liste over applikasjoner som må være installert.

For å kunne opprette en RPM-pakke må man først ha oppsett og verktøy for å bygge pakken. Det første som må gjøres er å installere pakkene *rpm-build* og *redhat-rpm-config*. Pakken *redhat-rpm-config* inneholder endel makroer og hjelpescript som ikke følger med *rpm-build*-pakken til CentOS. Etter å ha lagt inn disse verktøyene for bygging, må mappestrukturen til prosjektet opprettes. I hjemmemappen lagde vi mappen *rpmbuild* med sine undermapper; *BUILD*, *RPMS*, *SOURCES*, *SPECS*, *SRPMS*. Disse mappene er strukturert på denne måten for å ivareta sine funksjoner:

- BUILD – Denne mappen brukes til midlertidig lagring under byggingen av pakken. Rpmbuild vil faktisk installere GUCRSS sine filer i denne mappen, for å verifisere at filene ikke inneholder kritiske feil.
- RPMS – Under denne mappen blir den ferdige RPM-pakken lagt.
- SOURCES – Her ligger filene som skal være med i RPM. De ligger pakket i et .tar.gz filformat.
- SPECS – Her ligger .spec filen til RPM-pakken. Denne fungerer som en oppskrift på hvordan RPM skal installere filer og programmer.
- SRPMS – I denne mappen blir ferdigbygde SRPM-pakker lagt.

Når prosjekts filer er pakket i .tar.gz formatet og lagt i SOURCES-mappen, og .spec-filen er ferdig skrevet, ble pakken bygget med følgende kommando:

```
rpmbuild -ba <navnet på spec-filen>
```

Man finner da den ferdige RPM-pakken i *RPMS*-mappen og den ferdige SRPM-filen i *SRPMS*-mappen. Etter pakken er bygget er den for sikkerhets skyld digitalt signert, slik at

det kan verifiseres at det faktisk er vår pakke som blir installert på systemet. Det har i dette prosjektet blitt generert en egen ***GPG***-nøkkel for dette formålet. Dette gjøres ved å kjøre:

```
gpg --gen-key
```

Når nøkkelen er generert, signeres pakken med

```
rpm --addsign <rpm-pakken>
```

Deretter eksporteres den offentlige nøkkelen som må benyttes for å verifisere signaturen på RPM-pakken med kommandoen

```
gpg --export --armor > gpg-pub-key
```

## 6.2 Kompilering av binærpakke

GUCRSS består av kun en binærpakke, ettersom systemet baserer seg på PHP som prosesserer koden i sanntid uten å måtte kompilere koden på forhånd. Binærpakken er ett lite script(se kode under) som laster konfigurasjonsfilene til ***httpd*** på nytt. Scriptet blir kompilert under installasjonen.

```
/************************************************************
 * make sure to chmod +s the binary after compiling:
 * cc -o httpd_reload httpd_reload.c ; chmod +s httpd_reload
 * source from: http://adamyoung.net/Reload-httpd-via-PHP
 ************************************************************/

#include <stdio.h>
#include <stdlib.h>

int main() {
  if (!setuid(geteuid())) {
    system("/bin/echo '/sbin/service httpd reload > /dev/null 2>&1' | /usr/bin/at now");
  } else {
    printf("Couldn't set UID to effective UID\n");
    return 1;
  }
  return 0;
}
```

## 6.3 Installasjonen

En av pakkene GUCRSS avhenger av, php-pecl-json [7], ligger ikke i standard repositoriet til CentOS. Derfor er det ett krav at før installasjonen av GUCRSS kjøres, må ***EPEL*** være installert. Php-pecl-json blir installert når RPM-pakken kjøres. Dersom systemet ikke har installert EPEL på forhånd, kan dette enkelt gjøres ved å kjøre kommandoen

```
[root@public-82-171 rss]# su -c 'rpm -Uvh http://download.fedora.redhat.com/pub/epel/5/i386/epel-release-5-3.noarch.rpm'
Retrieving http://download.fedora.redhat.com/pub/epel/5/i386/epel-release-5-3.noarch.rpm
warning: /var/tmp/rpm-xfer.vsk2fS: Header V3 DSA signature: NOKEY, key ID 217521f6
Preparing...                ########################################### [100%]
   1:epel-release           ########################################### [100%]
[root@public-82-171 rss]#
```

**su -c 'rpm -Uvh http://download.fedora.redhat.com/pub/epel/5/i386/epel-release-5-3.noarch.rpm**

Når EPEL er installert på systemet må man verifisere pakken. Først må den offentlige GPG-nøkkelen legges inn i RPM GPG-nøkkelringen. Dette gjøres ved å kjøre følgende kommando som superbruker

**rpm --import gpg-pub-key**

Deretter sjekker man signaturen på RPM-pakken ved å kjøre

**rpm -K gucrss-1.0-3.noarch.rpm**

Man vil da få opp en melding som bekrefter at pakken er signert med den private nøkkelen som tilhører den offentlige nøkkelen som ble lagt inn i nøkkelringen over.

**gucrss-1.0-3.noarch.rpm: rsa sha1 (md5) pgp md5 OK**

Nå som pakken er verifisert, kan man trygt installere den på systemet. Den enkleste måten å gjøre dette på er ved å bruke et pakkebehandlingsverktøy som Yum. Eksempel på hvordan dette gjøres følger under.

**yum install gucrss-1.0-3.noarch.rpm**

Undervegs vil man få listet opp programmer som kreves for at GUCRSS skal kjøre. Man trenger da bare å svare ja på spørsmålene om man vil legge inn disse, så gjøres dette automatisk.

```
================================================================================================
 Package                                Arch                           Version
================================================================================================
Installing:
 gucrss                                 noarch                         1.0-2
Installing for dependencies:
 autoconf                               noarch                         2.59-12
 automake                               noarch                         1.9.6-2.1
 gcc                                    i386                           4.1.2-46.el5_4.2
 glibc-devel                            i386                           2.5-42.el5_4.3
 glibc-headers                          i386                           2.5-42.el5_4.3
 imake                                  i386                           1.0.2-3
 kernel-headers                         i386                           2.6.18-164.15.1.el5
 libgomp                                i386                           4.4.0-6.el5
 mod_auth_mysql                         i386                           1:3.0.0-3.2.el5_3
 mod_dav_svn                            i386                           1.4.2-4.el5_3.1
 mysql                                  i386                           5.0.77-4.el5_4.2
 openssl-perl                           i386                           0.9.8e-12.el5_4.6
 perl-DBI                               i386                           1.52-2.el5
 perl-URI                               noarch                         1.35-3
 php                                    i386                           5.1.6-24.el5_4.5
 php-cli                                i386                           5.1.6-24.el5_4.5
 php-common                             i386                           5.1.6-24.el5_4.5
 php-devel                              i386                           5.1.6-24.el5_4.5
 php-ldap                               i386                           5.1.6-24.el5_4.5
 php-mysql                              i386                           5.1.6-24.el5_4.5
 php-pdo                                i386                           5.1.6-24.el5_4.5
 php-pear                               noarch                         1:1.4.9-6.el5
 php-pecl-json                          i386                           1.2.1-4.el5
 subversion                             i386                           1.4.2-4.el5_3.1
Updating for dependencies:
 cpp                                    i386                           4.1.2-46.el5_4.2
 glibc                                  i686                           2.5-42.el5_4.3
 glibc-common                           i386                           2.5-42.el5_4.3
 libgcc                                 i386                           4.1.2-46.el5_4.2
 nscd                                   i386                           2.5-42.el5_4.3
 openssl                                i686                           0.9.8e-12.el5_4.6

Transaction Summary
================================================================================================
Install     25 Package(s)
Update       6 Package(s)
Remove       0 Package(s)

Total size: 49 M
Total download size: 49 M
Is this ok [y/N]:
```

Når installasjonen er fullført vil det se slik ut.

```
Installed:
  gucrss.noarch 0:1.0-2

Dependency Installed:
  autoconf.noarch 0:2.59-12                      automake.noarch 0:1.9.6-2.1
  gcc.i386 0:4.1.2-46.el5_4.2                    glibc-devel.i386 0:2.5-42.el5_4.3
  glibc-headers.i386 0:2.5-42.el5_4.3            imake.i386 0:1.0.2-3
  kernel-headers.i386 0:2.6.18-164.15.1.el5      libgomp.i386 0:4.4.0-6.el5
  mod_auth_mysql.i386 1:3.0.0-3.2.el5_3          mod_dav_svn.i386 0:1.4.2-4.el5_3.1
  mysql.i386 0:5.0.77-4.el5_4.2                  openssl-perl.i386 0:0.9.8e-12.el5_4.6
  perl-DBI.i386 0:1.52-2.el5                     perl-URI.noarch 0:1.35-3
  php.i386 0:5.1.6-24.el5_4.5                    php-cli.i386 0:5.1.6-24.el5_4.5
  php-common.i386 0:5.1.6-24.el5_4.5             php-devel.i386 0:5.1.6-24.el5_4.5
  php-ldap.i386 0:5.1.6-24.el5_4.5               php-mysql.i386 0:5.1.6-24.el5_4.5
  php-pdo.i386 0:5.1.6-24.el5_4.5                php-pear.noarch 1:1.4.9-6.el5
  php-pecl-json.i386 0:1.2.1-4.el5               subversion.i386 0:1.4.2-4.el5_3.1

Dependency Updated:
  cpp.i386 0:4.1.2-46.el5_4.2                   glibc.i686 0:2.5-42.el5_4.3
  glibc-common.i386 0:2.5-42.el5_4.3            libgcc.i386 0:4.1.2-46.el5_4.2
  nscd.i386 0:2.5-42.el5_4.3                    openssl.i686 0:0.9.8e-12.el5_4.6

Complete!
[root@public-82-171 rss]#
```

Etter installasjonen må man sette opp den lokale databasen for konfigurasjon og informasjon om repositorier og eksterne brukere. Det er valgt å benytte MySQL som database som beskrevet i kapitel 4.1. Dersom MySQL ikke ligger inne, kan dette lett legges inn.

```
Total download size: 9.9 M
Is this ok [y/N]: y
Downloading Packages:
(1/2): perl-DBD-MySQL-3.0007-2.el5.i386.rpm                    | 148 kB
(2/2): mysql-server-5.0.77-4.el5_4.2.i386.rpm                  | 9.8 MB
--------------------------------------------------------------------------------
Total                                                 4.6 MB/s | 9.9 MB
Running rpm_check_debug
Running Transaction Test
Finished Transaction Test
Transaction Test Succeeded
Running Transaction
  Installing     : perl-DBD-MySQL
  Installing     : mysql-server

Installed:
  mysql-server.i386 0:5.0.77-4.el5_4.2

Dependency Installed:
  perl-DBD-MySQL.i386 0:3.0007-2.el5

Complete!
[root@public-82-171 rss]#
```

**`yum install mysql-server`**

Når MySQL er ferdig satt opp kan man enkelt importere databasen ved å legge inn SQL-filen som blir installert og som standard legger seg på */usr/share/doc/gucrss-1.0/gucrss.sql* på ett CentOS 5 operativsystem.

```
[root@public-82-171 rss]# mysql -u root -p < /usr/share/doc/gucrss-1.0/gucrss.sql
Enter password:
[root@public-82-171 rss]#
```

**mysql -h *<host>* -u root -p < gucrss.sql**

Importering av databasen kan gjøres automatisk i RPM-pakken, men da databasen kan ligge på en annen maskin blir dette satt som en manuell oppgave for systemadministrator. På denne måten blir GUCRSS mer fleksibelt.

# 7. Avslutning

## 7.1 Resultater

Nå som vi er ferdig med utviklingsperioden og ser tilbake på produktet vi har produsert, føler vi at hovedkravene til at systemet (presentert i kapitel 2 Kravspesifikasjon) skal kunne tas i bruk er oppnådd. Med vårt produkt kan en bruker opprette den typen repository som er ønskelig, om det er til personlig bruk eller til prosjektjobbing i faglig sammenheng. Det kan gis tilgang til de personer man ønsker skal få ta del i prosjektet og med de rette rettighetene.

Ikke bare skulle det utvikles et webgrensesnitt men også et serversystem tilpasset CentOS slik at Apache, Subversion og MySQL skulle kunne samkjøre. Det var en tilfredsstillelse for oss da vi endelig hadde fått all serverkonfigurasjon på plass og kunne teste Web-grensesnittet av løsningen og se at det som før var gjort manuelt nå ble automatisert og satt i system.

Det vi er fornøyde med er at vi har fått på plass den viktigste funksjonaliteten, slik at systemet kan settes i drift. Totalt sett er vi veldig fornøyde med helheten av prosjektet. Vi har holdt en god struktur og jobbet veldig bra gjennom hele prosjektperioden.

Vi er stolte av at vi kan presentere en ferdig løsning som kan benyttes av oppdragsgiver ved endt prosjekt. Men med mer jobb enn tid til rådighet har vi vært nødt til å prioritere, noe vi føler at vi har gjort riktig.

Oppsummert er vi veldig fornøyde med hva vi har fått til med tanke på vår bakgrunn. Siden vi har utdannet oss som driftere, har vi størst kunnskap med konfigurasjon og funksjonalitet. Derfor har det til tider vært vanskelig å jobbe med et utviklingsprosjekt. Vi føler vi har fått kjempe god erfaring i å jobbe med større prosjekter.

## 7.2 Forbedringspotensiale og videre arbeid

### 7.2.1 Forbedringspotensiale

Selv om vi har klart å levere ett produkt som kan settes i drift, ønsker vi å fremheve forbedringspotensiale i vår løsning.

Vi har lagt rette for at vår løsning enkelt kan oversettes til andre språk. Vi har laget løsningen på engelsk. Dette fordi HiG er en institusjon med mange internasjonale studenter og ansatte. Dessverre har vi ikke fått oversatt løsningen til vårt morsmål.

Kildekoden kunne vært bygget opp mer objekt orientert. Vi hadde lite kunnskaper om objekt orientert programmering i PHP i starten av prosjektet. Når vi fant ut at man kunne programmere på denne måten i PHP, var vi allerede kommet så langt at det var uaktuelt å omstrukturere koden.

Brukergrensesnittet har store forbedringspotensiale med tanke på fargevalg, skriftstørrelse og plassering av objekter. Med vår bakgrunn har vi veldig liten erfaring på dette område, så vi har prøvd å legge til rette for at løsningen skal være enkel og intuitiv for folk flest.

Vi ser det er mulighet for å legge til ytterligere funksjonalitet opp mot vår løsning. Det har

vært tidspress hele vegen, og vi skulle ønske at vi hadde hatt litt bedre tid til å implementere ekstra funksjonalitet som for eksempel web-basert Subversion-klient og andre typer versjons-kontrollsystemer, men det må vurderes hvor mye dette vil bli brukt.

### 7.2.2 Videre arbeid

Først og fremst bør punktene i kapitel 7.2.1 prioriteres. Videre kan det være fornuftig å ta ut data fra GUCRSS til logwatch.

Ellers kan det implementeres flere funksjoner som blir utløst av hendelser som skjer i Subversion. Eksempler på dette kan være epost-varsling ved opprettelse av repository, ved checkin og ved checkout.

Det kan være ønskelig å ha bedre kontroll på opprettelse av repositorier i fremtiden, derfor kan funksjonalitet for godkjennelse før opprettelse på enkelte typer repositorier være nødvendig.

Da vi har lagt vekt på at systemet kan bli utvidet senere, har vi kommentert koden hele vegen, slik at det skal være enkelt for en programmerer å fortsette arbeidet videre.

## *7.3 Evaluering av gruppens arbeid*

### 7.3.1 Organisering

Organisering av prosjektet beskrives i vedlegg E og rollefordelingen har fungert uten problemer. Avgjørelser har foregått i plenum siden vi bare har vært 2 medlemmer i prosjektgruppen. Siden avgjørelsene har forekommet i felles diskusjoner innad i gruppen har det ikke vært store uenigheter og ingen behov for megling for å komme til enighet.

### 7.3.2 Fordeling

I starten av utviklingsperioden ble ansvaret fordelt slik at en skulle stå for koding og en for serverkonfigurasjonen. Dette løp over en periode på 2 uker og når serveren var klar fortsatte begge videre på kodingen. Siden gruppen bare bestod av 2 medlemmer ble ansvaret fordelt på begge to slik at man hele tiden hadde kontroll over hvem som jobbet hvilken del. Dette forenklet arbeidet slik at ingen jobbet på samme del samtidig.

I to bolker på 4 uker har det vært parallelle prosjekter i fag vi har tatt ved siden av Bachelorprosjektet. Dette medførte til en liten endring i organiseringen av arbeidet med tanke på fordeling av tid, men fikk ikke store konsekvenser for det totale resultatet av oppgaven.

### 7.3.3 Prosjektet som arbeidsform

Prosjektet har gitt oss en forsmak på hvordan det fungerer i det virkelige arbeidsmarkedet. I stedet for å jobbe med en oppgave som ikke er så farlig om ikke blir ferdig har vi her hatt ett press på oss tidsmessig for å få ferdig et fungerende system. Det å kunne planlegge og sette av tid til de ulike områdene i prosjektet har vært en test på vår evne til å estimere og realisere bruk av tiden. Dette har vært en god lærdom i forhold til det som venter oss når vi skal ut i arbeid.

### 7.3.4 Subjektiv opplevelse av bacheloroppgaven

Bacheloroppgaven har vært en god lærdom når det kommer til jobbing med utviklingsprosjekter. Dette har styrket vår samarbeidsevne betraktelig og lært oss å jobbe systematisk under press. Dette har vært en positiv opplevelse for oss med tanke på veien ut i arbeidslivet siden det i størst grad jobbes i grupper under prosjekter.

Siden vi har måttet sette oss inn i mye nytt stoff tilhørende webutvikling og server-konfigurering har dette medført til stor kompetanseheving innad i gruppen.

## *7.4 Konklusjon*

Det å gjennomføre et stort prosjekt, som reelt er tenkt å bli satt i bruk har vært en veldig lærerik og nyttig erfaring som vi begge vil ta med oss videre i livet. Vi har begge hatt viktige ansvarsforhold i prosjektet, samtidig som vi har jobbet sammen hele vegen. Vi har fått prøvd et utviklings-løp med delvis kravspesifisering, design og implementering.

I tillegg til utviklings-delen har vi lært oss å skrive rapport og dokumentere utviklingen undervegs. Det er første gang vi har skrevet en så omfattende rapport som dette, og vi føler det har gitt oss kunnskaper som kan brukes senere, uavhengig om vi skal ut i arbeid eller studere videre.

Siden kunnskapen om PHP og web-utvikling har vært liten fra start, har læringskurven vært desto større. Dette gir mersmak når vi ser tilbake på prosjektet.

I skrivende stund er GUCRSS i versjon 1.0-4.

Konklusjonen blir at vi har hatt en utfordrende og morsom tid i denne perioden når vi har jobbet med Bacheloroppgaven. Denne oppgaven har vært en veldig god måte å bygge på de kunnskapene vi allerede har og utvikle nye på områdene vi ikke kunne. Vi er i etterkant enige, begge to, at dette prosjektet har vært en fin avslutning på vårt 3-årige studieløp her på HiG.

# 8. Referanser

[1] Rfc 4511 - Lightweight Directory Access Protocol (LDAP): The Protocol. Bind operation. http://tools.ietf.org/html/rfc4511#section-4.2 , pr. Juni 2006

[2] Wikipedia, Programming style, http://en.wikipedia.org/wiki/Programming_style (Sist besøkt 18 Mai 2010)

[3] Jquery Usage Statistics, http://trends.builtwith.com/javascript/JQuery, pr. 27 april 2010.

[4] W3techs, Usage of javascript libraries for websites, http://w3techs.com/technologies/overview/javascript_library/all (Sist besøkt 18 Mai 2010)

[5] James Smith, jQuery Plugin: Tokenizing Autocomplete Text Entry, http://loopj.com/2009/04/25/jquery-plugin-tokenizing-autocomplete-text-entry (Sist besøkt 5 April 2010)

[6] Kelvin Luck, jQuery date picker plug-in, http://www.kelvinluck.com/assets/jquery/datePicker/v2/demo/index.html (Sist besøkt 18 Mai 2010)

[7] Gargiullo Family, Installing PECLjson on CentOS, http://gargiullo.com/?tag=json_encode , pr. 7 Juni 2009.

[8] Apache HTTP Server Project, Apache Module mod_authnz_ldap, http://httpd.apache.org/docs/2.2/mod/mod_authnz_ldap.html (Sist besøkt 18 Mai 2010).

[9] Code Remix, PHP cookies-enabled check, http://coderemix.com/php-cookies-enabled-check , pr. 15 Januar 2010.

[10] Web Builder Zone, CSS Message Boxes, http://css.dzone.com/news/css-message-boxes-different-me , pr. 26 Mai 2008.

[11] GURULABS, GURULABS-RPM-GUIDE-v1.0, http://www.gurulabs.com/downloads/GURULABS-RPM-LAB/GURULABS-RPM-GUIDE-v1.0.PDF

# 9. Vedlegg

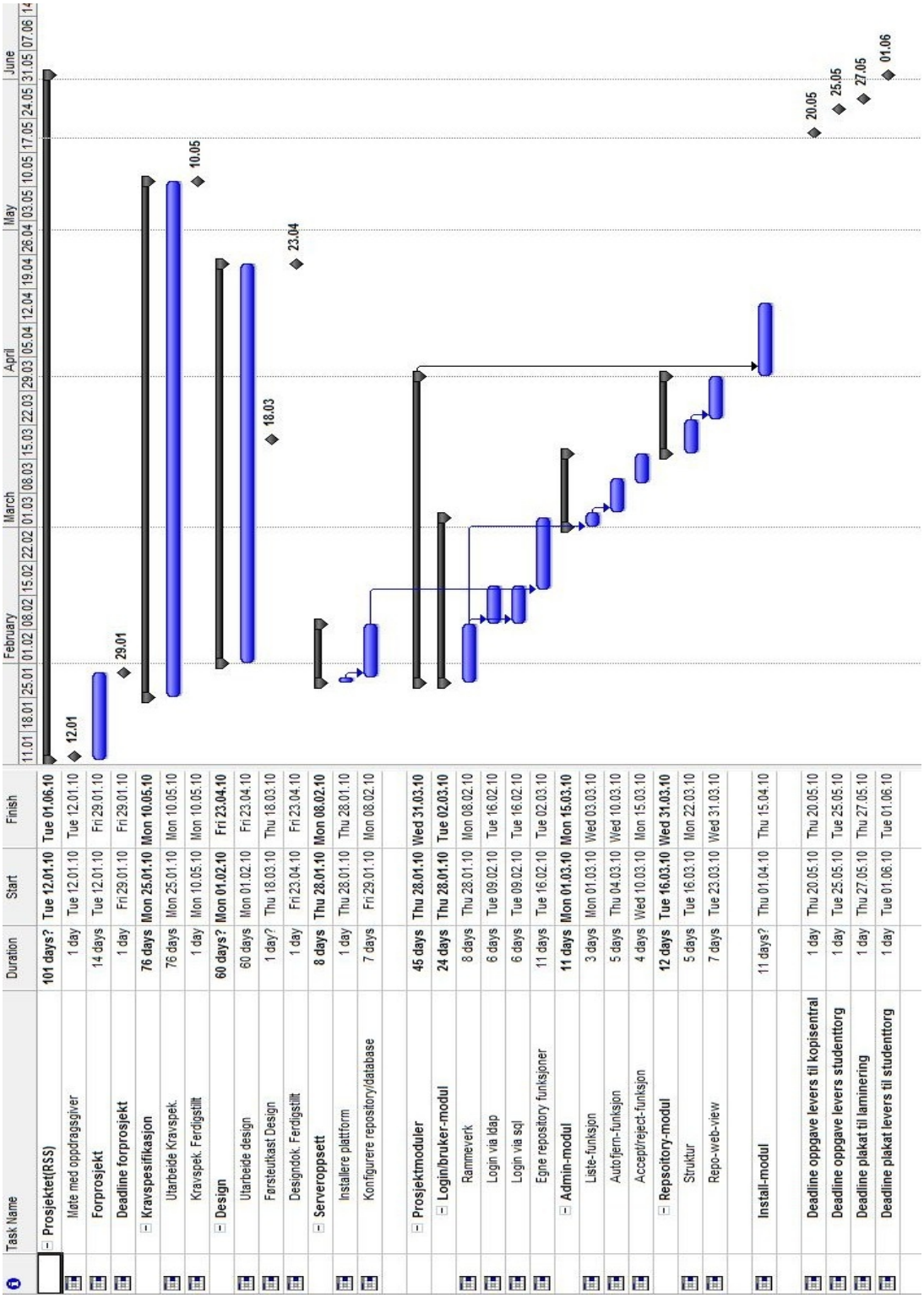
## *Vedlegg A – Definisjoner*

| Terminologi | Definisjon |
|---|---|
| Subversion | Versjonskontrollsystem for å holde rede på revisjonen til filer og mapper. |
| Repository | Versjonskontrollsystem-lagringsområde for filer og dokumentasjon. |
| Automatisering | At ting styrer mer eller mindre seg selv uten menneskelig innblanding. |
| SQL | Structured Query Language: databasespråk for behandling av data. |
| PHP | PHP: Hypertext Preprocessor. Programmeringsspråk for utvikling av dynamiske nettsider. |
| AJAX | Asynchronous JavaScript and XML: utviklingsteknikk for interaktive nettsider. |
| CentOS | Community ENTerprise Operating System: gratis operativsystem basert på Red Hat Enterprise Linux. |
| UNIX | Tverrplattform-operativsystem programmert i C. |
| LDAP | Lightweight Directory Access Protocol: brukes til oppslag i en servers katalogtjeneste. |
| IRC | Internet Relay Chat: protokoll som sender lynmeldinger i sanntid. |
| MySQL | Databasetjener basert på SQL. |
| Apache | Webserver primært for Unix-baserte operativsystemer. |
| Rutine | En fast måte å gjøre operasjoner på. |
| Dynamisk | I dette prosjektet, variabel belastning av systemet. |
| RPM | Red Hat Package Manager: pakkebehandlingssystem, kan bli referert som filformat eller programmet som håndterer dette formatet. |

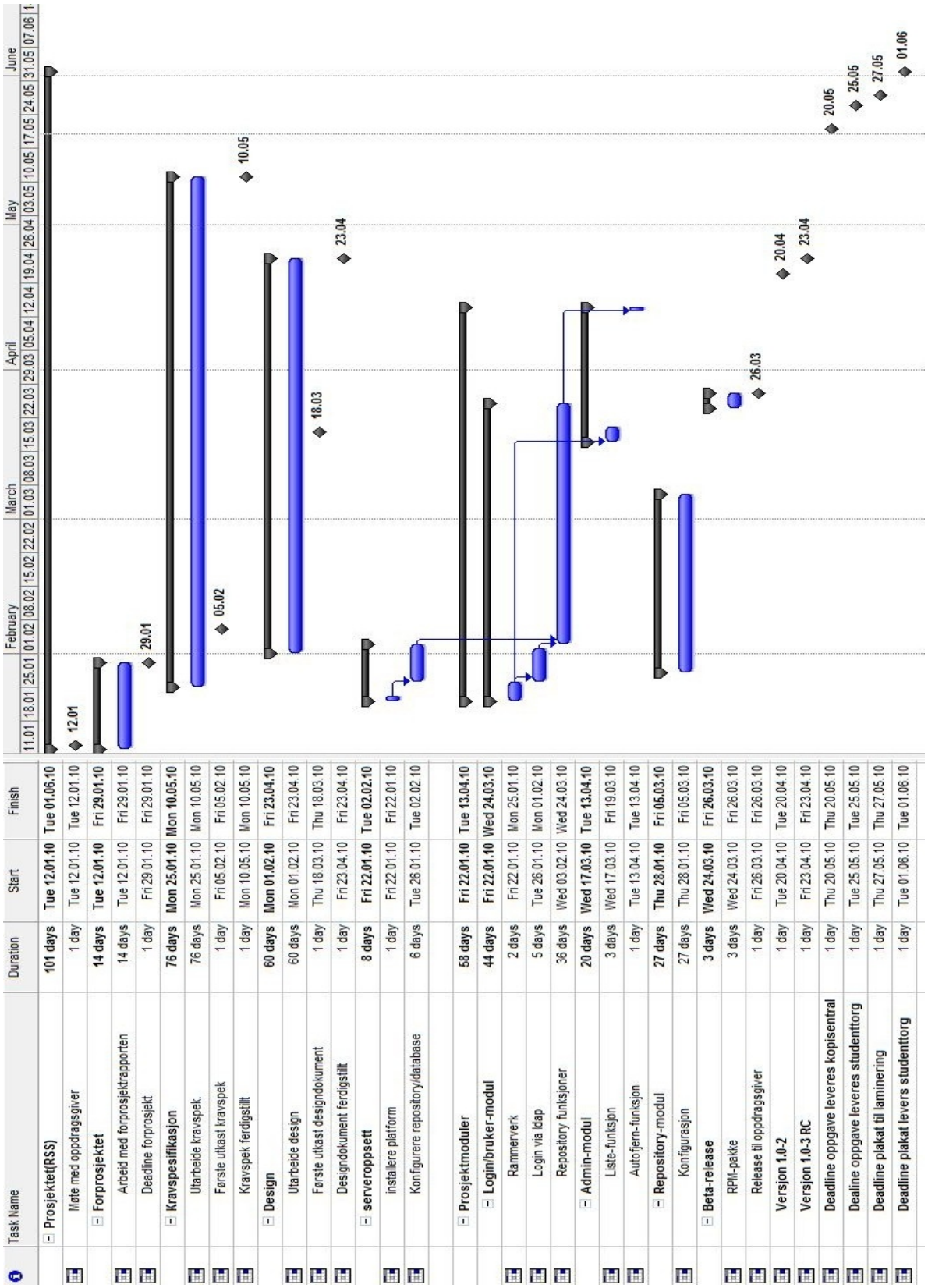
| | |
|---|---|
| GUCRSS | Gjøvik University College Repository Self Service: Det offisielle navnet på prosjektet og systemet som blir utviklet i prosjektet. |
| GUI | Graphical User Interface: Brukergrensesnittet til et program som dukker opp på skjermen. I motsetning til kommandolinje kan man klikke seg rundt i stedet for å bruke kommandoer. |
| SSL | Secure Sockets Layer: krypteringsprotokoll for sikker kommunikasjon over Internett. |
| Cookies | Informasjonskapsel brukt av nettlesere for lagring av personlig informasjon brukt ved forskjellige webtjenere. |
| Acl | Access control list: fil/liste som styrer hvem som har hvilken tilgang til hva på systemet. |
| Sertifikat | En fil som blir brukt som en digital legitimasjon når det kommuniseres over Internett. |
| URL | Uniform Resource Locator: adresse/sti til ressurs. |
| jQuery | Javascript-bibliotek brukt til forenkling av funksjonalitet i webapplikasjoner. |
| GPG | GNU Privacy Guard: Bruker asymmetriske nøkkelpar til kryptering. |
| HTTPD | Hyper Text Transfer Protocol Daemon: Apache/webserveren sitt program som kjører i bakgrunnen av systemet. |
| EPEL | Extra Packages for Enterprise Linux: repository for tilleggspakker for Enterprise Linux operativsystemer som for eksempel CentOS. |
| Headeren | Tittel/logo på en nettside. |

# *Vedlegg B – Fremdriftsplan*

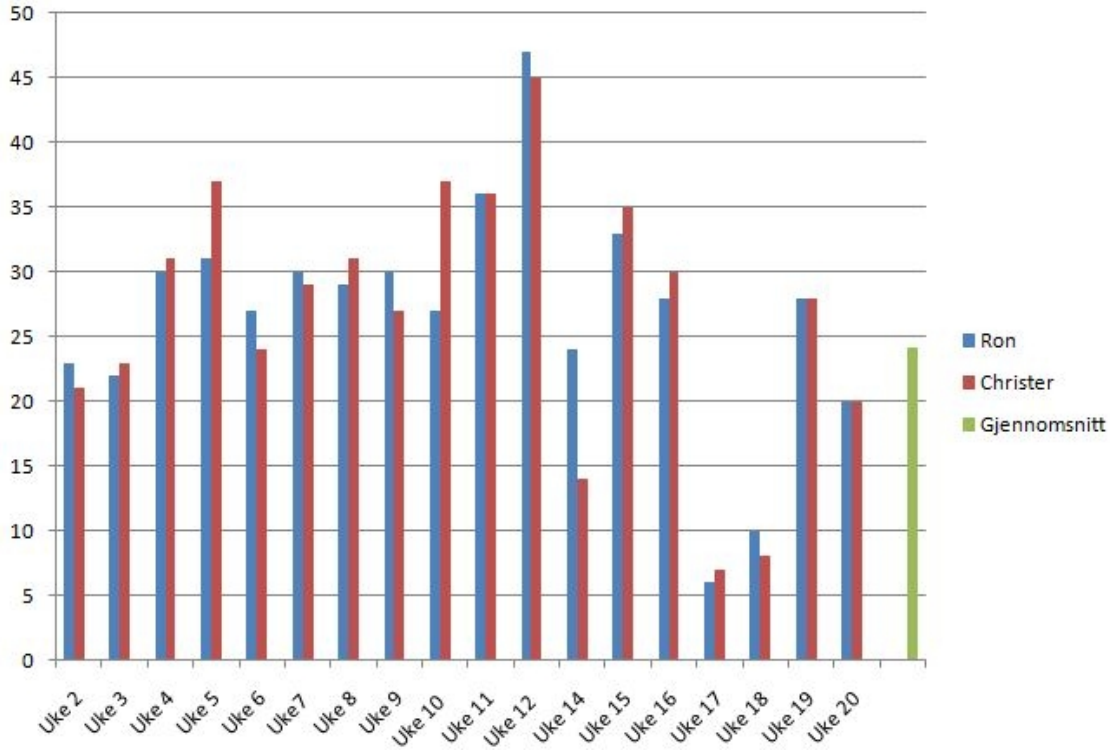
**Estimert fremdriftsplan**



## Faktisk fremdriftsplan



## *Vedlegg C – Logg, statusrapport, møtereferat*




Gruppens brukte arbeidstid i timer hver uke. Uke 17 og 18 ble mye av tiden brukt til å arbeide med et parallelt prosjekt i et valgfag som viste seg å være mer omfattende enn først antatt.

### Utdrag av møtereferater:

**Dato: 22/2-2010**

Sted: Veilederes kontor, K210

Tilstede:
- Erik Hjelmås(Veileder)
- Ron Stangvik(Prosjektleder)
- Christer Berg(referent)

Hva er gjort til nå:
- Plugins-rammeverk og test.
- Rettet bugs i kode

Hva skal fokuseres på fremover:
- Gjøre ferdig plugins
- Strukturere databasen

- Skrive videre på rapporten

Hva kom ut av dagens møte:
Vi hadde en liten missforståelse med oppdragsgiver angående rammer rundt plugins.
- fokus på retting av rammeverket der.

Veileder ga oss navn på ansatte vi kunne ha god nytte av under brukertesten i tillegg til studenter. Disse var: Rune Hjeldsvold, Monica Strand, Jayson Mackie.
Dette var også ressurspersoner som kan bistå ved problemer rundt databaser.

Vi skal fra dette punktet og frem til påske ha hovedfokus på kodingen men ha i bakhodet å ta notater til rapporten underveis. Dette gjør da oppgaven enklere når vi til slutt skal ta å nøste opp trådene før levering.

---

## Dato: 12/4-2010

Merknad: Møte fant sted rett etter tilbakemelding av betaversjon var levert og tilbakemeldinger trengte gjennomgang.

Sted: A018C, Gruppens arbeidsrom.

Tilstede:
- Erik Hjelmås(Veileder)
- Ron Stangvik(Prosjektleder)
- Christer Berg(referent)
- Anders Wiehe(Oppdragsgiver)

Hva er gjort til nå:
- Beta-utgivelse av løsningen levert og fått tilbakemeldinger på.
- Første utkast av rapport levert for vurdering/tilbakemelding

Hva skal fokuseres på fremover:
- Gjøre de mest kritiske endringene på løsningen
- Fortsette skrivingen av rapporten.

Hva kom ut av dagens møte:
Hadde levert et lite utkast av prosjektrapporten for tilbakemelding.
Fikk mange gode tilbakemeldinger rundt dette og noen tips:
- Se på stopmotion-prosektet for eksempel på god teknisk skrivemåte.
- Sette opp alle punktene i rapporten slik at det var lettere å se hva som skulle hvor.

Oppdragsgiver går igjennom fokuspunkter og endringspotensiale for koden. Ønsker ny versjon klar så fort som mulig.
Når det kommer til koden så skal vi fokusere på de mest kritiske endringene slik at den kunne tas i bruk til brukertesting.

## Statusrapporter


**HØGSKOLEN I GJØVIK**

# STATUSRAPPORT #2 ved Repository Self Service (RSS) Februar 2010

Status for:

- Tidsplan (fremdriftsplan).
- Database
- Koding

Totalstatus:

- Tidsplan (fremdriftsplan).
- Har måttet omrokkere på rekkefølgen i tidsskjemaet. Merker at tiden begynner å bli knapp men det går måtelig framover.
- Database
- Er oppe å går. Har hatt noen prøveversjoner før endelig versjon ble installert og tatt i bruk på serveren. Mulig det blir gjort noen små endringer for å få den slik at den passer prosjektet. Tilgangskontroll for ikke-hig brukere er lagt inn.
- Koding
- Login-modul ferdigstilt. Rammeverk for plugins ved create-repo er så godt som klare. Prioritet frem mot beta-levering er å ferdigstille plugins og få til edit repo.

Problemer oppstått:

- Reload apache fra php, altså tilgang til systemkommando fra php-kode.
- Plugins, førsteimplementering gjorde at plugins ble lastet to ganger som førte til at php krasjet. Dettet medførte full omskriving av plugins.
- For det meste generelle problemer rundt plugins har vært det største problemet.

Tidsfrister:

- Første modul(login) ble ferdigstilt i god tid før tiden. Gjorde at vi kunne starte på modul 3 før en beregnet. På grunn av noe lik kodestruktur har vi bestemt å bytte om på modul 2 og 3 slik at vi kan ferdigstille brukeres funksjoner og ta koden derfra og implementere den i admin-modulen.
- Har til nå ikke sprukket noen tidsfrist men sånn det ligger an nå så ser det ut til å sprekke på både modul 2 og 3 pga. mye unødvendig tid til søking etter løsninger

Motivasjon:

- Den siste måneden har motivasjon vår ligget ganske høyt med tanke på at vi parallelt med bacheloroppgaven har hatt et prosjekt i et annet fag. For å prøve å holde tiden har vi jobbet strukturert og oversiktlig for å skille mellom de to oppgavene.

Hva er ferdig:

- Login, create repository og databasen er det som til nå er ferdigstilt og fungerer. Data hentes fra databasen for opprettelse av config- og acl-filer.

Veilederkontakt:

- Vi føler at vi kunne tenke oss mer konkret tilbakemelding på veiledermøtene. Har virket som at veileder har for mye å gjøre i forhold til tid slik at det går utover vår møtetid.

Ron Stangvik, gruppeleder – Gjøvik, 09.03.2010


**HØGSKOLEN I GJØVIK**

# STATUSRAPPORT #3 ved Repository Self Service (RSS) Mars/April 2010

Status for:

- Tidsplan
- Koding
- Rapport

Totalstatus:

- Tidsplan
- Etter noe omprioritering av tidsplanen har vi klart å holde oss så vidt innenfor tidsplanen for fastsatt tidspunkt for levering av en ferdig løsning. Men merker også at tiden puster oss i nakken når det kommer til levering av ferdig oppgave. Blir mye hard jobbing fremover.
- Driver også parallelt med bachelorprosjektet med et enda et prosjekt i valgfaget vi tar ved siden av. Dette vil ha en ganske stor innvirkning på tidsskjemaet fremover. Gjelder for oss å planlegge tiden godt for å være effektive.
- Koding
- Der er løsningen ferdigstilt med de forandringer som oppdragsgiver hadde å komme med etter beta-utgivelse rundt påsketider.
- Rapport:
- Rapporten bygger seg sakte men sikkert opp. Driver å forbereder et utkast som vil bli sendt til vurdering.

Hva jobbes med nå:

- Ferdigstillelse av rapporten er det som jobbes med for fullt. Først og fremst et utkast som skal vurderes av veileder slik at vi kan forbedre punkter før endelig levering.
- Sette opp en oppgave som skal brukes til å se hvordan brukere vil bruke systemet i forhold til hvordan vi tror de vil bruke det.

Motivasjon:

- Arbeidsmoralen er på topp, merker at tidsfristen for levering nærmer seg og at vi jobber iherdig for å få produktet klart.
- Selv om også denne gang har et prosjekt gående parallelt prøver vi å jobbe så strukturert som mulig.

Hva er ferdig:

- Versjon 1.0-3 er ferdig og levert til oppdragsgiver. Det er den foreløpig endelige

versjonen av systemet og den som vil bli tatt i bruk under brukertesten.

Ron Stangvik, gruppeleder – Gjøvik, 26.04.2010

# *Vedlegg D – Design og spesifiseringsfiler*

## Stilskjema for repository-view i nettleser

```
<?xml version="1.0"?>

<!-- A sample XML transformation style sheet for displaying the Subversion
  directory listing that is generated by mod_dav_svn when the "SVNIndexXSLT"
  directive is used. -->
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform" version="1.0">

  <xsl:output method="html"/>

  <xsl:template match="*"/>

  <xsl:template match="svn">
    <html>
      <head>
        <title>
          <xsl:if test="string-length(index/@name) != 0">
            <xsl:value-of select="index/@name"/>
            <xsl:text>: </xsl:text>
          </xsl:if>
          <xsl:value-of select="index/@path"/>
        </title>
        <link rel="stylesheet" type="text/css" href="/svnindex.css"/>
        <link rel="stylesheet" type="text/css" href="https://minside.hig.no/hig-web.css"/>
                    <link rel="icon" href="https://minside.hig.no/favicon.ico" />
      </head>
      <body>
        <div id="head">
          <h1>GUC Subversion</h1>
        </div>
        <div id="navigation">
                      <xsl:element name="a">
                        <xsl:attribute name="src">
                                https://minside.hig.no/images/higlogo.png
                              </xsl:attribute>
                        <xsl:attribute name="width">100</xsl:attribute>
                        <xsl:attribute name="height">100</xsl:attribute>
                      </xsl:element>
            <div>
                                <xsl:element name="p">
                              <xsl:element name="a">
                                      <xsl:attribute name="href">https://minside.hig.no/</xsl:attribute>
                                      <xsl:text>My page</xsl:text>
                              </xsl:element>
                                      <xsl:text> / </xsl:text>
                              <xsl:element name="a">
                                      <xsl:attribute
name="href">https://minside.hig.no/student/</xsl:attribute>
                                      <xsl:text>Student</xsl:text>
                              </xsl:element>
                                      <xsl:text> / </xsl:text>
                              <xsl:element name="a">
                                      <xsl:attribute name="href">https://svn.hig.no/</xsl:attribute>
                                      <xsl:text>Repository administration</xsl:text>
```

```
                        </xsl:element>
                                <xsl:text> / </xsl:text>
                        <xsl:element name="a">
                                <xsl:attribute name="href"></xsl:attribute>
                                <xsl:text>Repository browser</xsl:text>
                        </xsl:element>
                         </xsl:element>
        </div>
                          <xsl:element name="br" />
        </div>
        <div class="content">
        <div class="svn">
                <xsl:apply-templates/>
        </div>
        </div>
    <div id="footer" class="footer">
                 <xsl:element name="p">

                        <xsl:text>&#xA9; </xsl:text>
                        <xsl:element name="a">
                                <xsl:attribute name="href">mailto:helpdesk@hig.no</xsl:attribute>
                                <xsl:text>helpdesk@hig.no</xsl:text>
                        </xsl:element>
                        <xsl:text> | </xsl:text>
                         <xsl:text>Powered by </xsl:text>
                <xsl:element name="a">
                        <xsl:attribute name="href">
                                <xsl:value-of select="@href"/>
                        </xsl:attribute>
                        <xsl:text>Subversion</xsl:text>
                </xsl:element>
                <xsl:text> </xsl:text>
                <xsl:value-of select="@version"/>
                 </xsl:element>
    </div>
  </body>
 </html>
</xsl:template>

<xsl:template match="index">
 <div class="rev">
        <xsl:element name="h2">
                <xsl:if test="string-length(@name) != 0">
                        <xsl:value-of select="@name"/>
                                <xsl:if test="string-length(@rev) != 0">
                                <xsl:text> &#8212; </xsl:text>
                                </xsl:if>
                </xsl:if>
                <xsl:if test="string-length(@rev) != 0">
                        <xsl:text>Revision </xsl:text>
                        <xsl:value-of select="@rev"/>
                </xsl:if>
        </xsl:element>
 </div>
 <div class="path">
         <xsl:element name="p">
                <xsl:value-of select="@path"/>
         </xsl:element>
```

```
  </div>
  <xsl:element name="p"><xsl:apply-templates select="updir"/>
  <xsl:apply-templates select="dir"/>
  <xsl:apply-templates select="file"/></xsl:element>
</xsl:template>

<xsl:template match="updir">
  <div class="updir">
    <xsl:text>[</xsl:text>
    <xsl:element name="a">
      <xsl:attribute name="href">..</xsl:attribute>
      <xsl:text>Parent Directory</xsl:text>
    </xsl:element>
    <xsl:text>]</xsl:text>
  </div>
  <!-- xsl:apply-templates/ -->
</xsl:template>

<xsl:template match="dir">
  <div class="dir">
    <xsl:element name="a">
      <xsl:attribute name="href">
        <xsl:value-of select="@href"/>
      </xsl:attribute>
      <xsl:value-of select="@name"/>
      <xsl:text>/</xsl:text>
    </xsl:element>
  </div>
  <!-- <xsl:apply-templates/ -->
</xsl:template>

<xsl:template match="file">
  <div class="file">
    <xsl:element name="a">
      <xsl:attribute name="href">
        <xsl:value-of select="@href"/>
      </xsl:attribute>
      <xsl:value-of select="@name"/>
    </xsl:element>
  </div>
  <!-- xsl:apply-templates/ -->
</xsl:template>

</xsl:stylesheet>
```

Kodelistingen over er for å styre hvordan repositoryoversikten i nettleseren blir presentert. Det var ønske fra oppdragsgiver at utseende skulle være som utseende på http://minside.hig.no .

## CSS som styrer utseende på info-boksene

```
/*********************************************************
* GUCRSS CSS - GUC 2010
* Some parts of this css is taken from:
* http://css.dzone.com/news/css-message-boxes-different-me
*********************************************************/
.info, .success, .warning, .error, .validation {
   border: 1px solid;
   margin: 15px;
   padding:15px 10px 15px 50px;
   background-repeat: no-repeat;
   background-position: 10px center;
}
.info {
   color: #00529B;
   background-color: #BDE5F8;
   background-image: url('info.png');
}
.success {
   color: #4F8A10;
   background-color: #DFF2BF;
   background-image:url('success.png');
}
.warning {
   color: #9F6000;
   background-color: #FEEFB3;
   background-image: url('warning.png');
}
.error {
   color: #D8000C;
   background-color: #FFBABA;
   background-image: url('error.png');
}
.validation {
   color: #D8000C;
   background-color: #FFBABA;
   background-image: url('validation.png');
 }
```

## *Vedlegg E – Kontrakter*

Avtale om Student

HØGSKOLEN I GJØVIK

### PROSJEKTAVTALE

mellom Høgskolen i Gjøvik (HiG) (utdanningsinstitusjon),

Høgskolen i Gjøvik v/it-tjenesten

(oppdragsgiver), og

Christer Berg og
Ron Stangvik

(student(er))

Avtalen angir avtalepartenes plikter vedrørende gjennomføring av prosjektet og rettigheter til anvendelse av de resultater som prosjektet frembringer:

1. Studenten(e) skal gjennomføre prosjektet i perioden fra 11/1-2010 til 3/6-2010.

   Studentene skal i denne perioden følge en oppsatt fremdriftsplan der HiG yter veiledning. Oppdragsgiver yter avtalt prosjektbistand til fastsatte tider. Oppdragsgiver stiller til rådighet kunnskap og materiale som er nødvendig for å få gjennomført prosjektet. Det forutsettes at de gitte problemstillinger det arbeides med er aktuelle og på et nivå tilpasset studentenes faglige kunnskaper. Oppdragsgiver plikter på forespørsel fra HiG å gi en vurdering av prosjektet vederlagsfritt.

2. Kostnadene ved gjennomføringen av prosjektet dekkes på følgende måte:
   - Oppdragsgiver dekker selv gjennomføring av prosjektet når det gjelder f.eks. materiell, telefon/fax, reiser og nødvendig overnatting på steder langt fra HiG. Studentene dekker utgifter for trykking og ferdigstillelse av den skriftlige besvarelsen vedrørende prosjektet.
   - Eiendomsretten til eventuell prototyp tilfaller den som har betalt komponenter og materiell mv. som er brukt til prototypen. Dersom det er nødvendig med større og/eller spesielle investeringer for å få gjennomført prosjektet, må det gjøres en egen avtale mellom partene om eventuell kostnadsfordeling og eiendomsrett.

3. HiG står ikke som garantist for at det oppdragsgiver har bestilt fungerer etter hensikten, ei heller at prosjektet blir fullført. Prosjektet må anses som en eksamensrelatert oppgave som blir bedømt av faglærer/veileder og sensor. Likevel er det en forpliktelse for utøverne av prosjektet å fullføre dette til avtalte spesifikasjoner, funksjonsnivå og tider.

4. Den totale besvarelsen med tegninger, modeller og apparatur så vel som programlisting, kildekode, disketter, taper mv. som inngår som del av eller vedlegg til besvarelsen, gis det en kopi av til HiG, som vederlagsfritt kan benyttes til undervisnings- og forskningsformål. Besvarelsen, eller vedlegg til den, må ikke nyttes av HiG til andre formål, og ikke overlates til utenforstående uten etter avtale med de øvrige parter i denne avtalen. Dette gjelder også firmaer hvor ansatte ved HiG og/eller studenter har interesser.

   Besvarelser med karakter C eller bedre registreres og plasseres i skolens bibliotek. Det legges også ut en elektronisk prosjektbesvarelse uten vedlegg på bibliotekets del av skolens Internett-sider. Dette avhenger av at studentene skriver under på en egen avtale hvor de gir biblioteket tillatelse til at deres hovedprosjekt blir gjort tilgjengelig i papir og nettutgave (jfr. Lov om opphavsrett). Oppdragsgiver og veileder godtar slik

offentliggjøring når de signerer denne prosjektavtalen, og må evt. gi skriftlig melding til studenter og dekan om de i løpet av prosjektet endrer syn på slik offentliggjøring.

5. Besvarelsens spesifikasjoner og resultat kan anvendes i oppdragsgivers egen virksomhet. Gjør studenten(e) i sin besvarelse, eller under arbeidet med den, en patentbar oppfinnelse, gjelder i forholdet mellom oppdragsgiver og student(er) bestemmelsene i Lov om retten til oppfinnelser av 17. april 1970, §§ 4-10.

6. Ut over den offentliggjøring som er nevnt i punkt 4 har studenten(e) ikke rett til å publisere sin besvarelse, det være seg helt eller delvis eller som del i annet arbeide, uten samtykke fra oppdragsgiver. Tilsvarende samtykke må foreligge i forholdet mellom student(er) og faglærer/veileder for det materialet som faglærer/veileder stiller til disposisjon.

7. Studenten(e) leverer 3 - tre - eksemplarer av oppgavebesvarelsen med vedlegg til Studenttorget. I tillegg leveres et eksemplar til oppdragsgiver. HiG kan stille til disposisjon ytterligere eksemplar(er) for oppdragsgiver mot at denne godtgjør produksjonskostnadene.

8. Denne avtalen utferdiges med et eksemplar til hver av partene. På vegne av HiG er det dekan som godkjenner avtalen.

9. I det enkelte tilfelle kan det inngås egen avtale mellom oppdragsgiver, student(er) og HiG som nærmere regulerer forhold vedrørende bl.a. eiendomsrett, videre bruk, konfidensialitet, kostnadsdekning og økonomisk utnyttelse av resultatene.

   Dersom oppdragsgiver og student(er) ønsker en videre eller ny avtale, skjer dette uten HiG som partner.

10. Når HiG også opptrer som oppdragsgiver trer HiG inn i kontrakten både som utdanningsinstitusjon og som oppdragsgiver.

11. Eventuell uenighet vedrørende forståelse av denne avtale løses ved forhandlinger avtalepartene i mellom. Dersom det ikke oppnås enighet, er partene enige om at tvisten løses av voldgift, etter bestemmelsene i tvistemålsloven av 13.8.1915 nr. 6, kapittel 32.

12. Deltakende personer ved prosjektgjennomføringen:

HiGs veileder (navn): Erik Hjelmås

Oppdragsgivers kontaktperson (navn): Anders Wiehe

Student(er) (signatur): _____ dato 21/01-10

_____ dato 21/1-2010

_____ dato _____

_____ dato _____

Oppdragsgiver (signatur): _____ dato 21/1-2010

Dekan (signatur): _____ dato 25/1-2010

Revidert 11.10.07, Ivar Moe


## *Vedlegg F – Brukertest dokument*

### RSS (Repository Self Service)

Instruction in english will follow.

*Merk: Hvis du er ansatt, hopp over punkt 2 og 3, og referer til punkt 1 på de øvrige punktene.*

Gå inn på http://svn.hig.no og logg deg inn med ditt studentnummer/ansattbrukernavn.

1. Opprett ett personlig repository
2. Opprett et repository til ett fag du tar, og legg til to personer i klassen som eiere.
3. Slett repositoryet du opprettet i punkt 1.
4. Fjern en av eierne fra repositoriet du opprettet i punkt 2.
5. Legg til foreleser og personen du fjernet i punkt 4 som en gruppe på repositoriet du opprettet i punkt 2.
6. Endre slutt-dato på repositoriet du opprettet i punkt 2.
7. Slett repositoriet du opprettet i punkt 2.
8. Logg deg ut.

*Note: If you are employee, skip step 2 and 3, and refer to step 1 in the other steps.*
Go to http://svn.hig.no and log in using your student number / employee username.

1. Create a personal repository
2. Create a repository to a course you take, and add two people in the class as owners.
3. Delete the repository you created in step 1
4. Remove one of the owners from the repository you created in step 2
5. Add the lecturer and the person you removed in step 4 as a group of the repository you created in step 2
6. Change end date on the repository you created in step 2
7. Delete the repository you created in step 2
8. Log out

Takk for din hjelp!

Vennelig hilsen Ron og Christer

---

Thank you for your time!

Sincerely Ron and Christer

## *Vedlegg G – CD-rommens innhold*

- Oppgavens kildekode
- Oppgavens rapport inkl. Vedlegg
- Plakat

***Vedlegg H – Forprosjektrapport***

# Forprosjektrapport for Repository Self Service

# Hovedoppgave våren 2010

**Christer Berg (070604 – 07HBDRA)**

**Ron Stangvik (070427 – 07HBDRA)**

# Innholdsfortegnelse

# 1. MÅL OG RAMMER

## *1.1. Bakgrunn*

Subversion ble utviklet av CollabNet i 1999/2000 og er et versjonskontrollsystem som hjelper enkeltepersoner og/eller grupper å holde versjoner av filer «up-to-date». Subversion holder kontroll på tidligere versjoner av dokumentene for mulighet til å rulle tilbake til en versjon som fungerte dersom noe går galt.

IT-tjenesten ved Høgskolen i Gjøvik (fra nå av HiG) tilbyr subversion-repositoryer til ansatte og studenter ved HiG. I dag blir administrering av repositoryer i stor grad manuelt behandlet av IT-tjenestens ansatte ved henvendelse til IT-tjenesten. Det er ønskelig å ha et system hvor studenter og ansatte i stor grad kan opprette og administrere sine repositoryer selv og generell administrasjon av systemet kan gjøres av IT-tjenesten.

## *1.2. Prosjektmål*

### 1.2.1 Resultatmål

Resultatet som skal oppnås med dette prosjektet er at det skal fjerne manuelt arbeid helt eller delvis for IT-tjenesten når det kommer til opprettelse av repositorier. Studenter og ansatte skal ved hjelp av et webgrensesnitt kunne opprette og administrere egne repositorier. IT-tjensten vil på sin side kunne gjøre utvidede endringer på repositorier ved hjelp av en administrasjonsmodul. Løsningen skal lages for å være skalerbar med tanke på utvidelse til andre versjonsstyringsverktøy. Da systemet er tenkt å settes i produksjon vil vi utvikle en sikker og stabil løsning, der vi også vil fokusere på brukervennlighet.

### 1.2.2 Effektmål

Effekten av dette systemet skal redusere manuelle oppgaver som kan automatiseres. Det vil gi IT-tjenesten tid til andre gjøremål. Studenter og ansatte vil slippe å vente på at deres repositorier skal godkjennes og opprettes manuelt, slik at de kan ta i bruk sitt repository med en gang.
Dette vil effektivisere oppdragsgivers arbeidshverdag og øke mulighetene for selvhjelp hos brukerene av IT-tjenesten.

### 1.2.3 Læringsmål

Vi ønsker å få benyttet oss av det vi har lært i løpet av de 3 årene vi har gått på HiG. I dette prosjektet har vi fått en utviklingsoppgave, selv om vår studieretning er rettet mot drift av nettverk- og datasystemer. Vi tror dette blir en veldig god erfaring å ha med seg ut i arbeidslivet, da drift i praksis vil ha noe utvikling når vi kommer ut i jobb. Vi ser allerede nå at vi kommer til å få god bruk for programmering, systemadministrasjon, WWW-teknologi og ikke minst systemutvikling. Dette er fag som ikke er typisk driftsrelatert, men som vi ser nytten av å ha vært igjennom.
Vi vil med dette prosjektet tilegne oss ferdigheter innenfor strukturert og systematisk jobbing, i tillegg til erfaring fra gruppearbeid over en lengre periode. Dette er ferdigheter som vi mener er viktige å ha med seg i arbeidslivet.

## *1.3. Rammer*

Prosjektet skal ferdigstilles til levering 20 Mai 2010. Systemet skal kunne kjøre på server med Centos 5 med subversion-repository. Det skal utvikles i PHP, MYSQL og autentisering skjer ved sjekk mot HiGs eksisterende LDAP-database, i tillegg til lokal database for andre brukere. Ferdig system skal leveres som RPM-fil.
Det er i utgangspunktet ikke avtalt noen faste tidspunkter for møte med oppdragsgiver, men vi vil varsle oppdragsgiver dersom det tas opp noe relevant på de ukentlige veiledermøtene, slik at oppdragsgiver har mulighet til å følge fremgangen. Ellers vil møte med oppdragsgiver bli avtalt etter behov.

# 2. OMFANG

## *2.1. Oppgavebeskrivelse*

IT-tjenesten ønsker at det utvikles et websystem hvor brukere kan opprette og endre repositoryinnstillinger selv. Med innstillinger menes repositoryadgangskontroll og hendelser ved innsjekking som et minimum. Det er også ønskelig at det utvikles en administrasjonsmodul tiltenkt ITtjenestens ansatte hvor man setter opp innstillinger for repositorylagring, kvoter, brukerkilder, standardbegrensninger og fjerning av repositoryer. Administrasjonsmodulen bør også kunne liste repositoryer, deres innstillinger og deres plassforbruk.

Systemet må kunne kjøre på IT-tjenestens valg av operativsystem, CentOS 5 og det må kunne bruke IT-tjenestens eksisterende brukerdatabase via LDAP. Det er en fordel hvis systemet kan skrives så generelt at det er lett å legge til støtte for andre repositoryer enn subversion, det kan kjøres på flere typer operativsystemer og bruke data fra flere forskjellige brukerdatakilder.

## *2.2. Avgrensning*

Det er ønskelig fra oppdragsgivers side at det skal være mulig å implementere flere typer versjonskontrollsystemer i systemet. Men systemet vil primært forholde seg til systemet Subversion(SVN), hvis tiden tilsier det vil vi også ta for oss denne oppgaven.

# 3. PROSJEKTORGANISERING

## *3.1 Ansvarsforhold og roller*

Gruppens prosjektleder er Ron Stangvik. Rapport- og dokumentansvarlig er Christer Berg. Prosjektleder har hovedansvaret for opprettelse av SVN struktur og er også hjemmesideansvarlig.

Christer Berg har som rapport- og dokumentansvarlig ansvaret for loggføring, men Ron Stangvik er pliktig til å levere antall timer og hva som er gjort til logg-ansvarlig.

Ron Stangvik plikter å holde prosjekt-siden oppdatert fortløpende når ny informasjon blir opprettet.

Når vi kommer til kodingen, vil vi dele ansvaret for hver modul. Vi ser da for oss at vi har hovedansvaret for annenhver modul. Selv om hovedansvaret for hver modul er fordelt, har begge ansvaret for at modulene skal ferdigstilles til rett tid.

## *3.2 Rutiner og regler i gruppa*

- Uoverenstemmelser skal diskuteres saklig for å komme til en løsning. Dersom vi ikke kommer til enighet, skal møte med veileder bli foretatt så snart dette er mulig.
- Alle gruppemedlemmer har ansvar for å loggføre sitt arbeid, med antall timer.
- Det fastsettes at gruppemedlemene skal bruke i snitt 30 timer i uka på oppgaven gjennom hele prosjektperioden.
- Det kan være perioder der det jobbes ekstra mye på prosjektet. Det gis da mulighet for å ta en roligere periode når dette er mulig, slik at gruppemedlemmene får mulighet til å jobbe seg opp med annet arbeid enn prosjektet.
- Fravær skal dokumenteres med årsak.
- Arbeid med oppgaven skal i hovedsak foregå på rom A018C.
- All dokumentasjon skal være formatert i henhold til fastsatte maler som finnes på gruppens subversion område.

# 4. PLANLEGGING, OPPFØLGING OG RAPPORTERING

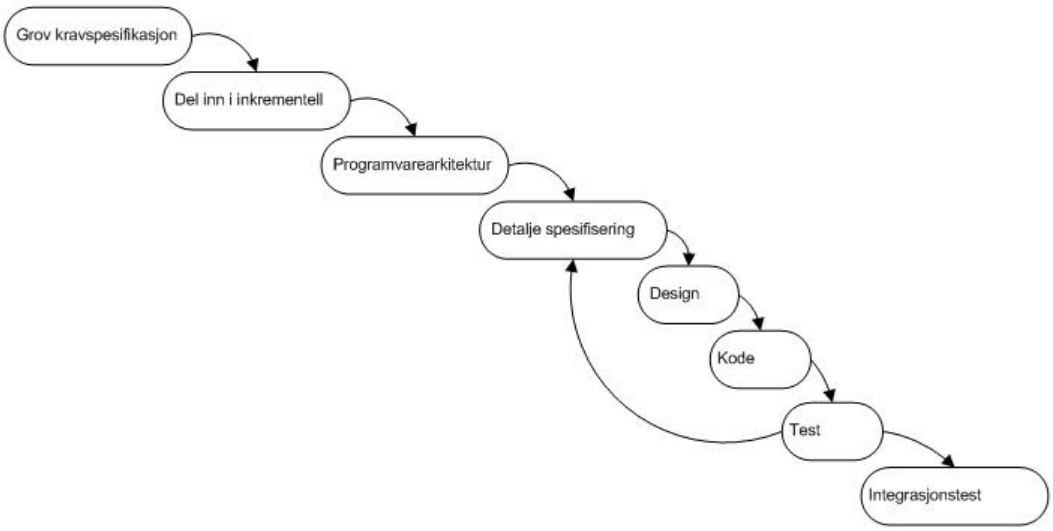
## *4.1 Hovedinndeling av prosjektet*

Oppdragsgiver har kommet med en grov kravspesifikasjon med forslag på hva som er ønskelig at produktet skal inneholde. Vi har tidligere fått opplyst at endel funksjoner på produktet er allerede eksisterende og tilgjengelig på forespørsel fra oppdragsgiver. Det er med dette naturlig for oss å benytte oss av inkrementmodellen i kombinasjon av en gjenbruksorientert utviklingsmodell.

Vi har valgt å styre unna evolusjonær-modellen i dette prosjektet. Dette fordi den fort blir uoversiktlig på større prosjekter, og ikke er godt egnet på modul-basert programvare. Evolusjonærmodellen produserer alle moduler på en gang, noe som blir uoversiktlig og øker sjansen for større feil. Vi går derfor ikke mer inn på denne, eller andre modeller som bygger på evolusjonær siden dette ikke er den arbeidsmetoden som passer for vårt prosjekt.

Fossefallsmodellen passer godt å bruke når man har fastsatte tidsfrister, og må ha god planstyring på det man driver med. Den tar for seg trinn for trinn i utviklingen, og går ikke videre før trinnet er ferdig. Til vårt produkt passet ikke denne utviklingsmodellen, da vårt system baserer seg på moduler, og endringer underveis vil forekomme. Fossefallsmodellen er meget svak på å håndtere endringer, derfor har vi besluttet å ikke velge denne modellen. I tillegg kan den være noe treg, og lite ressursutnyttende.

Inkrementmodellen passer godt til utvikling av modul-basert programvare. Modellen tar opp endringer etter hvert og ser viktige punkter som må endres tidligere enn andre modeller. Man starter med å finne kravene til programvaren, og prioriterer de forskjellige funksjonene som skal produseres. Deretter blir modulen(e) med størst prioritet produsert først. Når modulen(e) er ferdig kan den lanseres til oppdragsgiver, slik at han/hun kan sette produktet i produksjon med engang. Dette sørger for at de mest kritiske funksjonene blir testet nøye fra starten av, noe som sørger for lavere risiko med tanke på eventuelle feil som dukker opp i programvaren.




*Figur E-4-1 - Inkrementmodellen*

Systemet er tenkt at skal benyttes av studenter og ansatte på hele skolen. Vi vil derfor legge vekt på brukervennlighet, men valg av modell vil ikke bli påvirket av dette ettersom hovedvekten av arbeidet ligger i selve kodingen.

Vi har besluttet at vi skal benytte oss av inkrementmodellen (se fig. 4.1.1). Dette fordi den håndterer endringer underveis godt, og passer veldig bra til vårt prosjekt. Ved å velge denne modellen får vi god planlegging og fleksibilitet under utviklingen. Vi kan enkelt dra inn elementer fra andre modeller dersom dette trengs og bruke en kombinasjon av det beste fra flere modeller.

## *4.2 Plan for statusmøter og beslutningspunkter*

På statusmøtene vil vi gjennomgå hvordan vi ligger ann i forhold til arbeidsplanen og hva som skal gjøres fremover. Det vil bli skrevet en formell statusrapport månedlig.

Statusmøtene vil bli holdt ukentlig med veileder. Før hvert statusmøte vil vi ha en grov statusrapport som vi går igjennom med veileder. Oppdragsgiver vil også delta på noen av disse ukentlige møtene.

# 5. ORGANISERING AV KVALITETSSIKRING

## *5.1. Dokumentasjon, standardbruk og kildekode*

All kildekode som blir skrevet skal kommenteres på en forståelig måte slik at prosjektet kan fortsettes av andre etter utviklingsperioden, dersom dette er ønskelig.

Dokumentasjon som blir generert i prosjektperioden skal leses av begge gruppemedlemmene for å avdekke eventuelle skrivefeil og komme med forslag til forbedringer. Det er også avtalt at dokumentasjonen skal gjennomgås av en annen gruppe, og vi skal gjennomgå deres dokumentasjon. Dette gjøres for utvidet kvalitetssikring.

Nettsidene vi produserer i prosjektet skal være gyldig XHTML 1.0 Transitional.

Rapportene skal følge formatering til fastsatte maler (se punkt 3.2).

## *5.2. Versjonsstyring*

Det skal brukes versjonsstyringsverktøy på alle dokumenter og kildekode som er en del av prosjektet.

Gruppen vil benytte Subversion som versjonsstyringsverktøy i dette prosjektet. IT-tjenesten ved HiG drifter og tilbyr denne tjenesten.

## *5.3. Risikoanalyse*

**Ikke få ferdigstilt prosjektet til rett tid**
Risiko: LAV. Med god planlegging og ved bruk av riktig systemutviklingsmodell har vi tro på at dette går bra. Det er alltid en risiko for at noe uforutsett kan skje i prosjektperioden, slik at vi ser at risikoen er tilstede.

**Ikke få ferdigstilt prosjektet i sin helhet**
Risiko: MIDDELS. Det er en risiko for at noe dukker opp underveis, slik at vi ikke rekker alt vi skal.
Ved å dokumentere kode underveis, vil det være enklere å eventuelt videreutvikle applikasjonen senere. Vi vil også gå for en modulbasert applikasjon, slik at det vil være mulig å benytte den delen av prosjektet som er ferdigstilt.

**Miste arbeid som er gjort**
Risiko: LAV. Versjonsstyringsverktøy tar seg av automatisk backup i tilegg til at vi sprer alle dokumenter og filer utover maskinene som blir brukt i prosjektarbeidet. Dette minimerer risikoen for tap av data. Vi har ikke tatt høyde for ekstraordinære hendelser som «force majeur».

# 6. PLAN FOR GJENNOMFØRING

## *6.1 Hovedaktiviteter og milepæler*

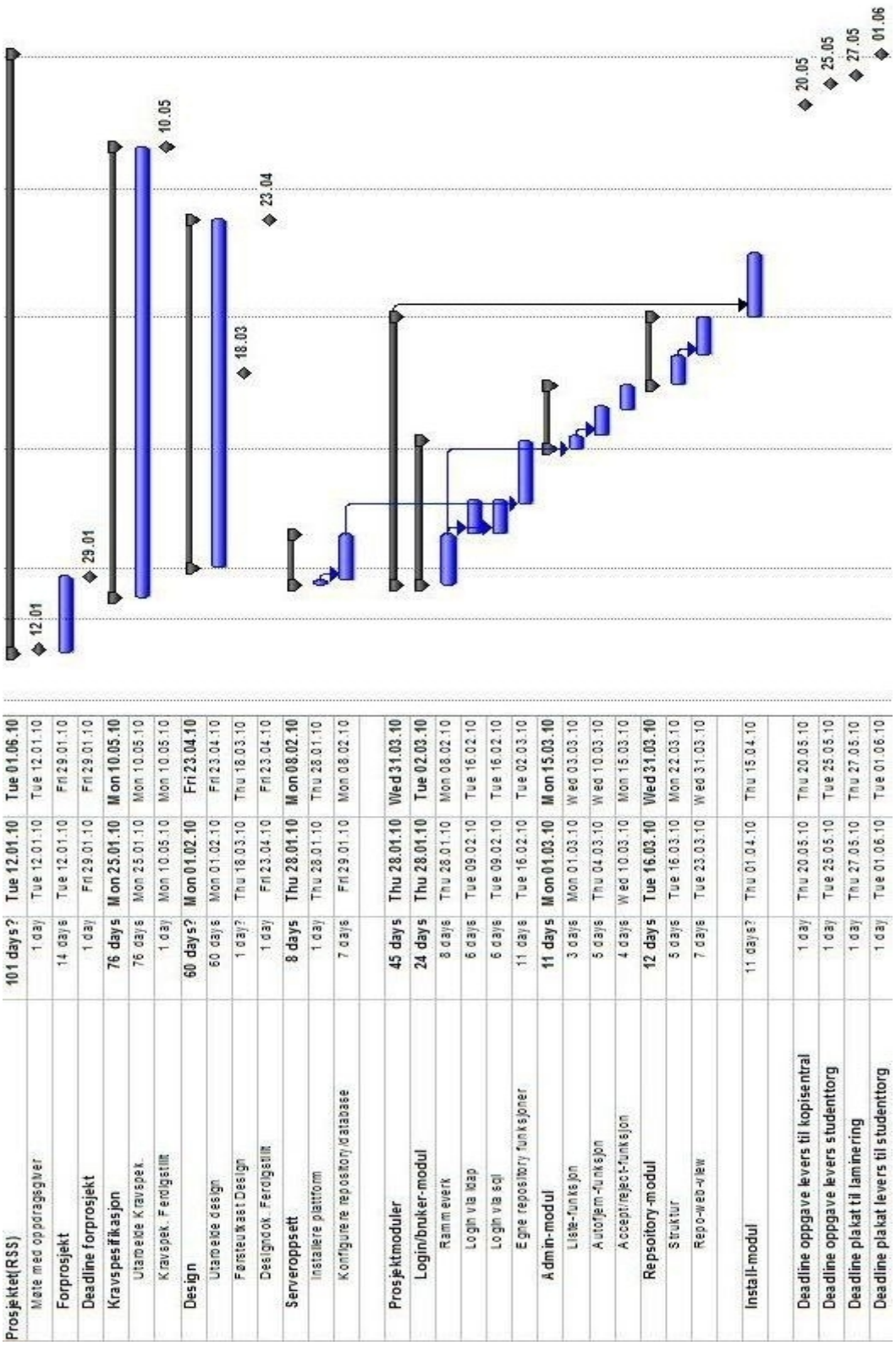
De viktigste datoene for prosjektet er

- 29/01/2010 – Innlevering av forprosjektsrapport og prosjektavtale med oppdragsgiver
- 20/05/2010 – Bacheloroppgaven leveres til kopisentralen
- 25/05/2010 – Bacheloroppgaven leveres til studenttorget
- 27/05/2010 – Plakaten til prosjektet leveres til laminering
- 01/06/2010 – Laminert plakat leveres til studenttorget
- 03/06/2010 – Presentasjon av oppgaven

Utenom disse datoene har vi ukentlige statusmøter med veileder og eventuelle tilleggsmøter etter behov.

Alle hovedaktiviteter er listet opp i vedlagt Gannt-skjema.

## *6.2 Gantt-skjema*



| Prosjektet(RSS) | 101 days? | Tue 12.01.10 | Tue 01.06.10 |
|---|---|---|---|
| Møte med oppdragsgiver | 1 day | Tue 12.01.10 | Tue 12.01.10 |
| Forprosjekt | 14 days | Tue 12.01.10 | Fri 29.01.10 |
| Deadline forprosjekt | 1 day | Fri 29.01.10 | Fri 29.01.10 |
| Kravspesifikasjon | 76 days | Mon 25.01.10 | Mon 10.05.10 |
| Utarbeide Kravspek. | 76 days | Mon 25.01.10 | Mon 10.05.10 |
| Kravspek. Ferdigstilt | 1 day | Mon 10.05.10 | Mon 10.05.10 |
| Design | 60 days? | Mon 01.02.10 | Fri 23.04.10 |
| Utarbeide design | 60 days | Mon 01.02.10 | Fri 23.04.10 |
| Førsteutkast Design | 1 day? | Thu 18.03.10 | Thu 18.03.10 |
| Design ok. Ferdigstilt | 1 day | Fri 23.04.10 | Fri 23.04.10 |
| Serveroppsett | 8 days | Thu 28.01.10 | Mon 08.02.10 |
| Installere plattform | 1 day | Thu 28.01.10 | Thu 28.01.10 |
| Konfigurere repository/database | 7 days | Fri 29.01.10 | Mon 08.02.10 |
| | | | |
| Prosjektmoduler | 45 days | Thu 28.01.10 | Wed 31.03.10 |
| Login/bruker-modul | 24 days | Thu 28.01.10 | Tue 02.03.10 |
| Rammeverk | 8 days | Thu 28.01.10 | Mon 08.02.10 |
| Login via ldap | 6 days | Tue 09.02.10 | Tue 16.02.10 |
| Login via sql | 6 days | Tue 09.02.10 | Tue 16.02.10 |
| Egne repository funksjoner | 11 days | Tue 16.02.10 | Tue 02.03.10 |
| Admin-modul | 11 days | Mon 01.03.10 | Mon 15.03.10 |
| Liste-funksjon | 3 days | Mon 01.03.10 | Wed 03.03.10 |
| Autofjern.-funksjon | 5 days | Thu 04.03.10 | Wed 10.03.10 |
| Accept/reject-funksjon | 4 days | Wed 10.03.10 | Mon 15.03.10 |
| Repsoitory-modul | 12 days | Tue 16.03.10 | Wed 31.03.10 |
| Struktur | 5 days | Tue 16.03.10 | Mon 22.03.10 |
| Repo-web-view | 7 days | Tue 23.03.10 | Wed 31.03.10 |
| | | | |
| Install-modul | 11 days? | Thu 01.04.10 | Thu 15.04.10 |
| | | | |
| Deadline oppgave levers til kopisentral | 1 day | Thu 20.05.10 | Thu 20.05.10 |
| Deadline oppgave levers studenttorg | 1 day | Tue 25.05.10 | Tue 25.05.10 |
| Deadline plakat til laminering | 1 day | Thu 27.05.10 | Thu 27.05.10 |
| Deadline plakat levers til studenttorg | 1 day | Tue 01.06.10 | Tue 01.06.10 |