
BACHELOROPPGAVE:

# EasyUp

FORFATTERE:


- Ola Kjelsrud
- Ole Woldstad


Dato:

- 23.05.2012

# SAMMENDRAG


| | | |
|---|---|---|
| Tittel: | EasyUp | Dato: 23.05.2012 |
| Deltakere: | Ola Kjelsrud | |
| | Ole Woldstad | |
| Veileder: | Øyvind Kolloen | |
| Oppdragsgiver: | Høgskolen i Gjøvik | |
| | Kjell Are Refsvik | |
| Stikkord | Opplastingsgrensesnitt, DSpace, HTML5, JavaScript, PHP | |
| Antall sider: | Antall vedlegg: | Tilgjengelighet: Åpen |

Kort beskrivelse av bacheloroppgaven:

Deling av elektronisk materiale er alltid en utfordring.

Det skal være enkelt raskt, men samtidlig sikkert og pålitelig.

DSpace er et gratis digitalt arkiv som kan brukes til publisering av alle typer digitale filer.

Dessverre er dagens metode for å publisere filer veldig omfattende slik at vår oppdragsgiver ikke har lykkes i å få folk til å bruke systemet. Oppgaven var derfor å utvikle en løsning som gjør publisering av filer langt enklere.

Vi brukte mye ny HTML5 funksjonalitet for å utvikle et webgrensesnitt som blant annet har denne funksjonaliteten:

- Dra og slipp filer
- Fortsettelse av avbrutt opplasting av filer
- Automatisk henting av metadata fra filer i tillegg til manuell input.
- Forhåndsvisning av bilder i nettleser før opplasting
- Framdriftslinje på opplasting av filer

Etter at filer er lastet opp publiserer systemet automatisk filene in i DSpace.

Vår løsning er enkel og raskt, men samtidig tar den i mot nok informasjon slik at DSpace kan publisere filene på riktig måte.

# SUMMARY


| | | |
|---|---|---|
| Title: | EasyUp | Date: 23.05.2012 |
| Participants: | Ola Kjelsrud | |
| | Ole Woldstad | |
| Supervisor: | Øyvind Kolloen | |
| Employer: | Høgskolen i Gjøvik | |
| | Kjell Are Refsvik | |
| Keywords: | File Upload, DSpace, HTML5, JavaScript, PHP | |
| Number of pages: | Number of appendix: | Availability: Open |

Short description of the bachelor thesis:

Sharing digital files will always be a challenge. You want it to be quick and easy, but at the same time you want security and reliability. DSpace, an open source digital repository enables easy and open access to all types of digital content.

Publishing files in DSpace however is a complicated and time consuming process.
Our goal was to simplify this process. We chose to do so with new HTML5 functionality.

We developed a web interface which supports some of the following functionality:

- Drag and Drop
- Pause/Resume on uploads
- Automatic gathering of metadata from files and manually added metadata
- Previews in browser depending on file type. (Thumbnails for pictures)
- Progress bar on uploads

When files are uploaded to the web server, the files are automatically published to DSpace.
The goal was to make this process quick and easy, while still maintaining the complexity DSpace requires.

# EasyUp

Ola Kjelsrud & Ole Woldstad

# Forord

Denne bacheloroppgaven har som mål å forenkle publisering av elektronisk materiell for ansatte og studenter på Høgskolen i Gjøvik. Vi har laget og presentert en webløsning for dette. Vår tilnærming ble godt mottatt av oppdragsgivers kontaktperson Kjell Are Refsvik. Vi ønsker at leseren av denne rapporten skal få et innblikk i hvordan vi har jobbet, og hva vi har fått til i løpet av semesteret.

Rapporten vil også gi et innblikk i hvordan det er å jobbe med ikke satte standarder, ny funksjonalitet og hvordan man forholder seg til dette.

Med HTML5 standarden fortsatt under utvikling visste vi at problemer ville oppstå, men også mulighet til å løse utfordringer på nye måter. Vi så frem til å følge med på utviklingen av den nye standarden og prøve ting underveis.

Vi vil takke vår oppdragsgiver høgskolelektor Kjell Are Refsvik for god hjelp under prosjektet. Vi har stått veldig fritt til å utvikle produktet på vår måte. Takk for at du stolte på vår tolkning av oppgaven og håper du er fornøyd med resultatet.

Vi vil også takke veileder Øivind Kolloen, en god støttespiller som har gitt oss veiledning og gode tips underveis. Takk også for tilbakemelding av rapportene før levering.

Til slutt vil vi takke Høgskolen i Gjøvik for informasjon og gode kurs for bachelorprosjekter.

Høgskolen i Gjøvik 23/05-2012

…… ……

Ola Kjelsrud Ole Woldstad

# Innhold

1.0 Introduksjon ..... 1
1.1 Innledning ..... 1
1.2 Oppgavebeskrivelse ..... 1
1.3 Avgrensninger ..... 2
1.4 Målgrupper ..... 3
1.5 Prosjektmål ..... 3
1.5.1 Effektmål ..... 4
1.5.2 Resultatmål ..... 4
1.5.3 Læringsmål ..... 5
1.6 Faglig bakgrunn ..... 5
1.7 Rammer ..... 6
1.7.1 Organisering ..... 6
1.7.2 Utviklingsmodell ..... 7
1.8 Roller i prosjektet ..... 7
1.9 Organisering av rapporten ..... 8
1.9.1 Definisjoner ..... 9
2.0 Kravspesifikasjon ..... 10
2.1 Krav til systemet ..... 10
2.2 Omgivelser ..... 11
2.3 Funksjon ..... 12
2.4 Operasjon ..... 20
3.0 Design ..... 22
3.1 Arkitekturbeskrivelse ..... 23
3.2 Moduler ..... 24
3.3 Handlingsflyt for bruker ..... 25

3.4 Grensesnittdesign ..... 30
4.0 Bakgrunn ..... 33
4.1 DSpace ..... 33
4.1.1 Historie ..... 33
4.1.2 Funksjonalitet ..... 34
4.1.3 Importering av filer ..... 36
4.1.4 Installasjon og konfigurasjon ..... 38
4.1.5 Kjøring av DSpace ..... 39
4.2 HTML5 ..... 41
4.3 CSS ..... 41
4.4 JavaScript ..... 42
4.5 AJAX ..... 42
4.6 PHP ..... 43
4.7 Oppsummering ..... 43
5.0 Implementering ..... 44
5.1 Tankeprosess ..... 44
5.2 Innlogging ..... 45
5.3 Hovedsiden ..... 47
5.3.1 Hjelpefiler for publisering ..... 49
5.3.2 SWORD ..... 53
5.4 Filhåndtering ..... 55
5.4.1 Dra og slipp ..... 55
5.4.2 Input ..... 56
5.4.3Viser valgte filer ..... 57
5.4.4 Fortsettelse av avbrutt opplasting ..... 58
5.5 Opplasting ..... 59
5.6 Publisering ..... 62

5.7 Sikkerhet ..... 65
5.7.1 Kommunikasjon ..... 66
5.7.2 System ..... 67
5.8 Testing ..... 69
6.0 Avslutning ..... 70
6.1 Plan vs virkelighet ..... 70
6.1.1 Avvik i produktet ..... 70
6.1.2 Avvik under uvikling. ..... 72
6.2 Selvevaluering ..... 75
6.2.1 Innledning ..... 75
6.2.2 Forbedringespotensiale og videre arbeid ..... 76
6.2.3 Gruppeorganisering og arbeidsfordeling ..... 76
6.2.4 Personlig tilbakeblikk ..... 77
7.0 Konklusjon ..... 79
8.0 Litteraturliste ..... 80
9.0 Vedlegg ..... 82
Vedlegg A - Forprosjekt ..... 82
Vedlegg B - Prosjektavtale ..... 97
Vedlegg C - Grupperegler ..... 100
Vedlegg D - Statusrapporter ..... 102
Vedlegg E - Møtereferater ..... 107
Vedlegg F - Logg ..... 114
Vedlegg G - Dublin Core metadata ..... 135

# Figurliste

Figur 1 - Use Case ..... 12
Figur 2 - Logisk arkitektur ..... 23
Figur 3 - Handlingsflyt ..... 25
Figur 4 - EasyUp oversikt over metadata ..... 28
Figur 5 - DSpace oppbygning ..... 34
Figur 6 - Dublin_core.xml eksempel ..... 37
Figur 7 - Dataflyt for DSpace tilkobling ..... 40
Figur 8 - Apache virtuell host konfigurasjon ..... 40
Figur 9 - cURL tilkobling til DSpace for autentifisering ..... 45
Figur 10 - EventListener eksempel ..... 47
Figur 11 - Liste over mulige metadata ..... 47
Figur 12 - Ekstra metadata eksempel ..... 48
Figur 13 - Dublin_core.xml oppbygging ..... 49
Figur 14 - Sending av metadata fra JavaScript til PHP ..... 50
Figur 15 - Bygging av Dublin_core.xml ..... 51
Figur 16 - Håndtering av språkvalg: 1 ..... 52
Figur 17 - Håndtering av språkvalg: 2 ..... 53
Figur 18 - Håndtering av språkvalg: 3 ..... 53
Figur 19 - Sword Servicedokument ..... 54
Figur 20 - Dra og slipp dropsone ..... 55
Figur 21 - Kode for dropsone ..... 55
Figur 22 - Dropsone skifter farge ..... 56
Figur 23 - Kode for fargeskift ..... 56
Figur 24 - Kode for input element ..... 56
Figur 25 - Input elementets funksjonalitet ..... 56
Figur 26 - EventListener for dropp og input av filer ..... 57
Figur 27 - Legger til nye filer i fillisten ..... 57
Figur 28 - Viser valgte filer ..... 57
Figur 29 - Kode for miniatyrbilder ..... 58
Figur 30 - Viser ikke publiserte filer for bruker ..... 59
Figur 31 - Kode for å dele opp filen som skal opplastes ..... 60
Figur 32 - Kode for å sende fil til server ..... 60

Figur 33 - Kode for å lagre fil på server ..... 60
Figur 34 - Fremgangslinje funksjonalitet ..... 61
Figur 35 - Kode for fremgangslinje ..... 61
Figur 36 – Bash skript for importering av filer til DSpace ..... 63
Figur 37 - Kode for å kjøre bash skript fra PHP ..... 63
Figur 38 - Skjermbilde under publisering ..... 64
Figur 39 - Skjermbilde for fullført publisering ..... 65
Figur 40 - Kommunikasjonsflyt for data ..... 66
Figur 41 - Kode for å sikre brukerinput ..... 68
Figur 42 - Grov tidsplan ..... 74

# 1.0 Introduksjon

## 1.1 Innledning

Deling av elektronisk materiale er alltid en utfordring. Det bør være enkelt, sikkert, og raskt. Høgskolen i Gjøvik har i dag ingen god løsning på dette problemet. Ansatte og studenter bruker mange forskjellige tjenester for deling og innleveringer av filer.
Eksempler på slike tjenester er:

- Google Documents
- Imageshack
- Pastebin

Dette fører til at materiell ligger spredt utover mange tjenester. Skolen opplever dermed ofte problemer da flere brukere skal ha tilgang til samme informasjon samtidlig. Et annet problem er når studenter skal levere oppgaver. Ansatte må ofte laste ned materiell fra flere forskjellige kilder. Dette er både tidkrevende og tungvindt. I tillegg hender det at man ikke får tilgang på opplastet materiell og må derfor anse innleveringen som ikke levert.

I 2010 satte Høgskolen i Gjøvik opp en DSpace server. En publiseringsserver for elektronisk materiell. Dessverre er grensesnittet for å publisere filer alt for omfattende ”ut av boksen” slik at ingen har tatt i bruk serveren.

## 1.2 Oppgavebeskrivelse

Høgskolen i Gjøvik ønsker å benytte sin DSpace server til lagring av filer for ansatte og studenter. Serveren skal kunne lagre alle typer elektronisk materiell over lengre tid slik at studenter kan referere til innhold også etter studietiden.

Vår oppgave blir dermed å sette oss inn i hvordan DSpace fungerer og utvikle et opplastingsgrensesnitt som overtar opplasting og publisering av nytt materiell inn i DSpace.

Oppdragsgiver var veldig fleksibel på hva og hvordan dette skulle gjennomføres. Vi kom derfor sammen til følgende konklusjon:

Vi skulle utvikle et webgrensesnitt som tar seg av opplasting av filer til server. Løsningen skulle utvikles ved hjelp av HTML5 og skulle støtte mye av den nye funksjonaliteten som kommer med standarden. Følgende funksjonalitet skulle implementeres:

- Dra og slipp filer (I tillegg til vanlig filvalg)
- Legge til metadata, i tillegg til de data som finnes i filen fra før (Data om data)
- Lisensvalg (Publiseringslisens)
- Forhåndsvisning av valgte filer
- Mulighet for fortsettelse av avbrutt opplasting

Etter at bruker har valgt filer og lagt til metadata skal filene lastes opp til server. Når opplasting er ferdig skal løsningen automatisk publisere filene, slik at DSpace kan vise de til bruker.

For å få til ønsket funksjonalitet kommer vi til å bruke HTML5, PHP, JavaScript, bash, cURL og deler av DSpace.

## 1.3 Avgrensninger

DSpace driftes av IT-Tjensten. Eventuelle nye tjenester kan ikke direkte settes inn i et produksjonsmiljø uten omfattende testing. Vi har derfor satt opp vår egen DSpace server og webserver for utvikling og testing under bacheloroppgaven. Vi leverer en løsning som fungerer, men pga tid og sikkerhetsaspekter har vi ikke fått testet løsningen i et produksjonsmiljø. Det blir derfor opp til oppdragsgiver i samarbeid med IT-Tjensten å sette løsningen i drift etter levering.

På grunn av ikke satte standarder og stadig ny funksjonalitet i blant annet HTML5, har vi valgt og kun støtte følgende nettlesere:

- Google Chrome
- Mozilla Firefox
- Apple Safari

Nettlesere som Internet Explorer og Opera har ikke implementert støtte for mye av funksjonaliteten HTML5 tilbyr. Vi har derfor valgt og ikke støtte disse per dags dato. Når HTML5 standarden er satt og disse nettleserne legger til støtte for den nye funksjonaliteten, vil det enkelt være mulig og støtte disse nettleserne også.

## 1.4 Målgrupper

### Prosjektet

Det er i hovedsak oppdragsgiver, Høgskolen i Gjøvik, IMT som er målgruppen for bruken av systemet. De skal dra nytte av produktet som lages videre i undervisningssammenheng. Sensor og veileder er også i målgruppen da de skal stå for vurdering av bacheloroppgaven.

I tillegg håper vi at andre aktører kan ta utgangspunkt i vårt prosjekt for bruk i flere mulige sammenhenger. EasyUp kan med enkle grep splittes fra DSpace og brukes som et opplastingsgrensesnitt for filer til andre formål.

### Rapporten

Rapporten er skrevet for at oppdragsgiver og sensor skal få innblikk i hvordan vi har jobbet med prosjektet. Her vil vi dokumentere hva vi har vurdert og tatt hensyn til underveis. I tillegg vil vi vise hvordan vi har løst de forskjellige problemstillingene vi har møtt.

Rapporten er i hovedsak skrevet for folk med teknisk bakgrunn innenfor webutvikling og programmering. Men alle som leser rapporten vil få ett godt innblikk i hva som er gjort og hvordan ting kan løses.

## 1.5 Prosjektmål

Prosjektmål deler vi opp i flere underkategorier. Effektmål sier hva oppdragsgiver forventer. Resultatmål sier hva gruppen skal levere og oppnå. Vi har også valgt å ta med læringsmål siden dette er en bacheloroppgave. Læringsmål sier hva gruppen ønsker å oppnå med prosjektet.

### 1.5.1 Effektmål

Oppdragsgiver ønsker et enkelt opplastingsgrensesnitt der ansatte og studenter kan laste opp filer i forbindelse med undervisning. Løsningen skal automatisk publisere de opplastede filene i DSpace. DSpace tar seg av lagring og visning av filene.

Dette skal føre til at alt materiell lagres på samme plass og vi unngår dagens spredning over flere nettbaserte løsninger. Løsningen skal tilby sikker langtidslagring for alle parter.

### 1.5.2 Resultatmål

Gruppen skal utvikle et opplastingsgrensesnitt som automatisk publiserer filer i DSpace. Løsningen skal være enkel i bruk, men samtidlig gi nok informasjon om publiserte filer slik at DSpace kan håndtere de på en korrekt måte.

Løsningen skal inneholde mye ny funksjonalitet fra HTML5. Dette innebærer grensesnittene for filhåndtering: File-API og kommunikasjon mellom klient og server: XMLHttpRequest litteraturliste.

Siden mye funksjonalitet fortsatt er under utvikling har vi som mål å støtte nettleserne nevnt under avgrensninger.

Løsningen skal primært kunne brukes under utdanning, men innholdet skal også være tilgjengelig for studenter etter endt utdanning, slik at man kan referere til innhold ved f. eks jobbsøknader.

### 1.5.3 Læringsmål

Vi ser frem til å jobbe med et større prosjekt over en lengre periode. Vi har i løpet av studietiden lært mye om planlegging, arkitektur og strukturering av arbeid og kode. Nå ser vi frem til å bruke det vi har lært i praksis.
Hovedtemaer vi kommer innom blir:

- HTML5
- Diverse grensesnitt
- PHP
- JavaScript
- DSpace

Vi håper også at vi kommer til å få god innsikt i hvordan det er å jobbe med ikke satte standarder som HTML5. Annen nyttig erfaring vil bli å tilpasse løsningen til å fungere på flere nettlesere som støtter ulik funksjonalitet, få til en god og effektiv serverkommunikasjon og forenkle funksjonaliteten til DSpace med et oversiktlig grensesnitt.

## 1.6 Faglig bakgrunn

Begge gruppens medlemmer har samme faglige bakgrunn. Vi har begge utdannelsen serviceelektroniker og studerer nå ingeniørfag data på Høgskolen i Gjøvik.

Vi har programmeringserfaring på c, c++, og script språk som bash og PowerShell fra HIG. Fra jobb og fritidsprosjekter har vi også vært innom litt enkel webdesign. Deriblant HTML, PHP og JavaScript.

Via HiG har vi også hatt systemutvikling som kommer godt med på et prosjekt som dette.

Vi var klare på at vi langt fra var noen eksperter på disse emnene, men vi så for oss en god mulighet for å bruke mye av teorien vi har lært i praksis. Samtidlig så vi muligheten til å lære oss nye ferdigheter. Vi så også frem til å følge med på utviklingen av den nye HTML5 standarden og tilhørende grensesnitt.

## 1.7 Rammer

Underveis i prosjektet vil det være en del tidsfrister.

- 27 jan. Prosjektavtale og forprosjekt
- 3 mar. Hjemmeside for prosjektet
- 23 mai. Prosjektrapport
- 30 mai. Plakat for presentasjon.

I tillegg vil det være muntlig presentasjon av prosjektet 07/06-2012.

Vi har i tillegg til dette valgt å lage en statusrapport i starten på hver måned som vi legger til på prosjektets hjemmeside. Disse er laget for at veileder og oppdragsgiver skal få en oversikt over fremgangen på prosjektet. Statusrapporter finnes i vedlegg D.

### 1.7.1 Organisering

Siden gruppen bare består av 2 medlemmer som i tillegg bor sammen har vi valgt å jobbe hjemmefra under prosjektperioden. Ola Kjelsrud er valgt som prosjektleder. Lederen har dobbeltstemme i alle beslutninger slik at det i teorien blir et diktatur.
NB: lederen er demokratisk valgt.

Vi har satt oss som mål å jobbe med prosjektet minst 16 timer hver i uka. Siden vi bor sammen blir det også naturlig å jobbe sammen og samtidlig. Vi har derfor innredet et eget prosjektrom der vi kommer til å jobbe med oppgaven. Fordelen med dette er at vi unngår distraksjoner og at problemstillinger som dukker opp underveis kan enkelt og raskt diskuteres og løses.

Grupperegler finnes i vedlegg C.

### 1.7.2 Utviklingsmodell

Når det kommer til valg av utviklingsmodell har vi valgt eXtreme Programming (XP) som rammeverk, men med visse modifikasjoner. XP er en fleksibel modell som takler endringer underveis godt. Vi vil ikke benytte oss av ”On Site Customer” fult ut slik som XP, men vi vil hele tiden ha kontakt med oppdragsgiver for tilbakemelding.

Vi ser for oss at vi benytter disse hovedpunktene i vår utvikling:

- Inkrementell og iterativ utvikling. Legger hele tiden til ny funksjonalitet.
- Kontinuerlig testing. Tester hver ny funksjon, både alene og sammen med andre.
- Parprogrammering. På enkelte moduler vil vi benytte parprogrammering.
- Brukerinteraksjon under utvikling. Tilbakemelding fra oppdragsgiver og veileder.
- Daglige møter (Stand up meetings). Mellom gruppemedlemmer.
- Refactoring. Renskriving av kode underveis selv om produktet fungerer.
- Gjenbruk. Modifisere kode andre har laget.

## 1.8 Roller i prosjektet

Prosjektgruppen består av følgende medlemmer:

- Ola Kjelsrud (Prosjektleder)
- Ole Woldstad

Oppdragsgiver er Høgskolen i Gjøvik. Vår kontaktperson er Kjell Are Refsvik. Vi skal ha flere møter underveis i prosjektet der vi skal diskutere arbeid som er gjort og eventuelt ny ønsket funksjonalitet.

Veileder ved Høgskolen i Gjøvik er Øivind Kolloen. Hans rolle vil være veiledning og tips, samt tilbakemelding på rapporter underveis i prosjektet.

Møtereferat fra alle møter finnes i vedlegg E.

## 1.9 Organisering av rapporten

Etter en introduksjon av oppgaven og hva vi hadde planlagt i kapittel 1, fortsetter vi i kapittel 2 med kravspesifikasjon. Her bruker vi use case for å beskrive funksjonalitet som oppdragsgiver ønsket. Dette er krav vi har utviklet sammen med oppdragsgiver.

I kapittel 3 går vi gjennom systemets design. Her beskriver vi hvordan systemet er delt inn i moduler og hva slags løsninger som er valgt for hver av dem.

Kapittel 4 tar for seg hva vi har brukt for å utvikle systemet. Her går vi gjennom DSpace, HTML, CSS, JavaScript, AJAX og PHP. Litt enkelt informasjon som forklarer hva og hvordan disse teknologiene fungerer.

I kapittel 5 tar vi for oss implementeringen av systemet. Her forklarer vi hvordan vi har brukt teknologiene fra kapittel 4 for å få til ønsket funksjonalitet fra kapittel 2, kravspesifikasjon. Vi går gjennom vår tankeprosess for hvordan vi har tenkt under utviklingen. I tillegg bruker vi både illustrasjoner fra systemet og kode for å forklare funksjonalitet. Til slutt går vi kort gjennom sikkerhet. Her forklarer vi eventuelle sikkerhetshull, og hvordan vi har løst de største av dem.

Kapittel 6 starter med å ta for seg mangler og avvik i systemet. Her nevner vi hva vi ikke har implementert og hvorfor. Til slutt har vi tatt med litt selvevaluering. Her reflekterer vi over prosjektperioden. Til slutt har vi også tatt med personlige refleksjoner av perioden.

Etter konklusjonen i kapittel 7 der vi kort oppsummerer bacheloroppgaven ligger alle vedleggene.

### 1.9.1 Definisjoner

| Terminologi | Definisjon |
|---|---|
| HTML | HyperText Markup Language. |
| CSS | Cascading Style Sheets. |
| XML | Extensive Markup Language. |
| CNRI | Corporation for National Research Initiatives. Ikke profitt organisasjon som har utviklet handle systemet DSpace kan bruke for permanente linker til innhold. |
| PHP | PHP: Hypertest Preprocessor. |
| API | Application Programming Interface. Bibliotek med metoder som kan brukes for å kommunisere med programvare. |
| LDAP(S) | Lightweight Directory Access Protocol. Protokoll brukt for å kommunisere med en katalogtjeneste på en server. Som f. eks Active Directory. |
| DOM | Document Object Model. Beskriver et HTML eller XML dokument som en trestruktur. |
| AJAX | Asynchronous JavaScript and XML. Flere teknologier som jobber sammen for å gjøre nettsider mer dynamisk og responsive nettsider. |
| CGI | Common Gateway Interface. Standard for web server programvare for å kjøre kjørbare filer. |
| Metadata | Data om data. Beskriver eller definere andre data. |
| XMLHttpRequest | API for å sende informasjon mellom klient og server. |
| FTP | File Transfer Protocol. |
| HTTP | HyperText Transfer Protocol. |
| HTTPS | Sikker utgave av HTTP. Bruker SSL/TSL for å kryptere kommunikasjon. |
| cURL(libcurl) | Komandolinjeverktøy og apacheplugin for dataoverføring støtter HTTP, FTP, LDAP, POP3 m. Fler. |
| SSL | Secure Socket Layer |
| TSL | Transport Layer Security |
| ATP | Advanced Packaging Tool. Brukes av Debian for å håndtere programvare. Installasjon og avinstallasjon av programmer. |

# 2.0 Kravspesifikasjon

Kravspesifikasjonen er utarbeidet fra den generelle oppgavebeskrivelsen beskrevet i punkt 1.2. I tillegg har vi fått informasjon og tilbakemeldinger muntlig, i våre møter med Kjell Are Refsvik. Se møtereferater vedlegg E.
Det meste av valgene vi har tatt i vår løsning er basert på kravene fra vår oppdragsgiver om plattformuavhengighet og brukervennlighet.

Kravspesifikasjonen vil inneholde vårt sammendrag av all informasjon vi har fått knyttet til krav om funksjonalitet og operasjonsrelaterte krav. Samt omgivelser og rammer systemet skal operere under.

## 2.1 Krav til systemet

På grunn av vår valgte fremgangsmåte og oppdragsgivers åpne tankegang rundt funksjonaliteten og systemdesignet er det svært få krav som var satt til å begynne med.
Kun følgende hovedkrav var til stede når vi startet planleggingen av prosjektet:

- Vi skal lage et opplastningsgrensesnitt til DSpace.
- Systemet skal kunne brukes av alle skolens brukere, studenter og ansatte.
- Det skal kunne brukes under ulike operasjonssystemer (Mac/Windows/Linux).
- Det skal benytte seg av skolens DSpace server for publisering.
- Systemet skal være brukervennlig.

Disse kravene setter standard for hva som forventes av løsningen. I tillegg kommer krav som blir satt av rammene vi jobber under, og detaljkrav gitt av oppdragsgiver underveis.

## 2.2 Omgivelser

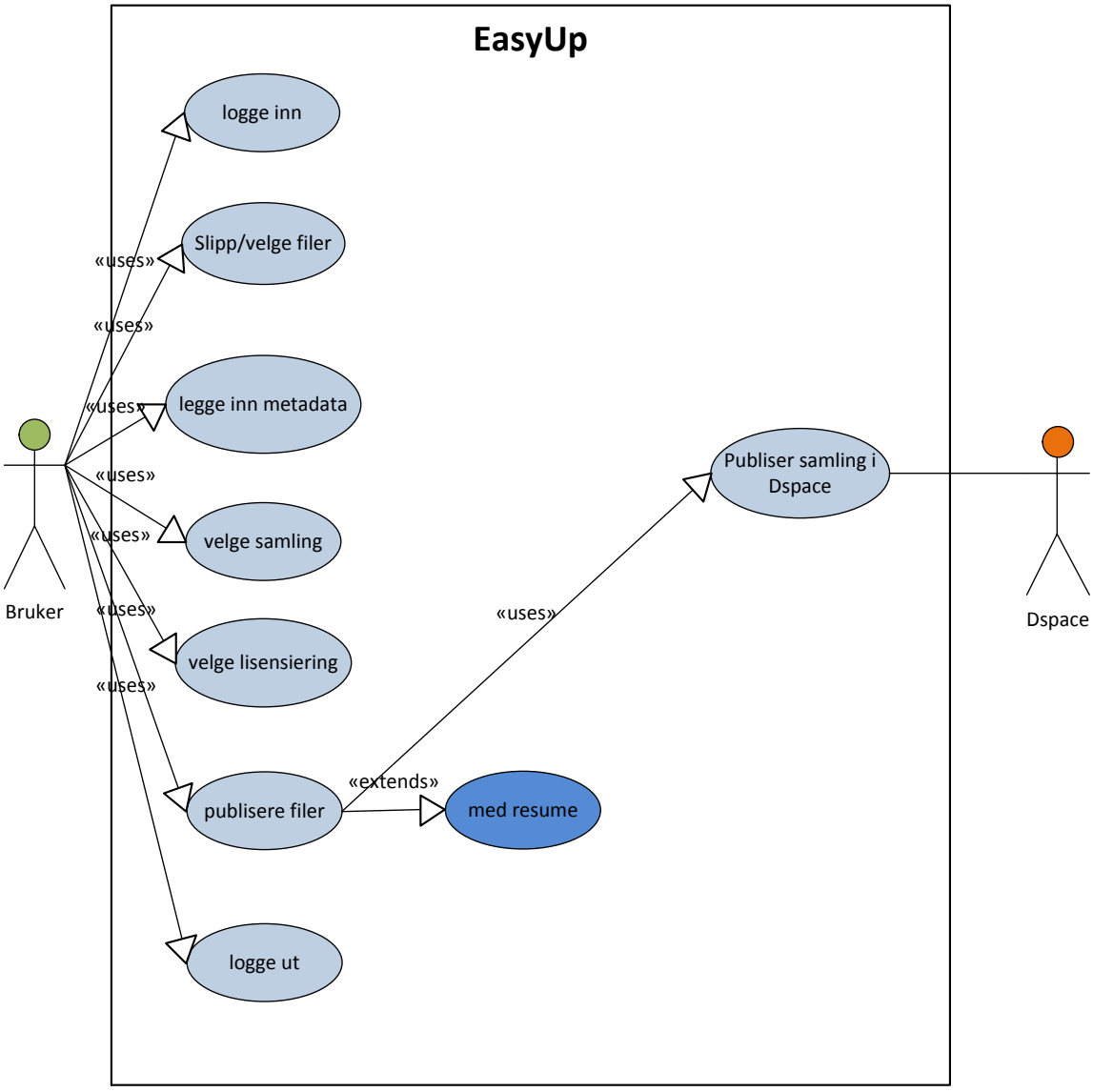
EasyUp blir en nettbasert løsning og alt som kreves på klientsiden skal være en kompatibel nettleser. Autentisering skal kreves før hver bruk av EasyUp.
Det vil være skolens IT-Tjeneste som har ansvar for å drifte løsningen etter levering.
Det vil derfor være IT-Tjenesten sitt ansvar at serveren er maskinvaremessig utrustet for å håndtere trafikken som bruken av vårt system medfører.

EasyUp serveren på skolen skal kjøre DSpace og Apache med PHP-støtte. Apacheserveren må ha modulene for simpleXML og cURL installert. For håndtering av XML-filer og tilkobling til DSpace for brukerautentifisering.

Løsningen vår skal være tilgjengelig både internt i skolens nettverk og eksternt.
Autentifisering skal foregå via feide og følge skolens retningslinjer for sikkerhet.
Planen til skolen er at filer som ligger på serveren skal være tilgjengelig også flere år etter at elevene som har publisert filene har sluttet som studenter på HIG.

## 2.3 Funksjon

I tillegg til de funksjonelle hovedkravene nevnt i innlendingen til dette avsnittet skal løsningen inneholde funksjonalitet vist og forklart her ved bruk av et use case diagram med tilhørende use case beskrivelser.




**Figur 1 - Use Case**

| Use Case: | **Logg inn** |
|---|---|
| Aktør: | Bruker |
| Mål: | Autentisere bruker og presentere bruker for webgrensesnittet. |
| Sammendrag: | Bruker logger inn med feide brukernavn og passord. Om autentiseringen er vellykket blir bruker logget inn i systemet og hovedsiden vises. |
| Forutsetninger: | Bruker må eksistere i DSpace eller DSpace må være satt opp slik at bruker blir opprettet ved innlogging(LDAP). |
| Ferdigtilstand: | Session er opprettet i nettleser med nødvendig informasjon. |

| **Handlingsforløp** | |
|---|---|
| Aktør | System |
| 1. Bruker går til nettsideadressen. | 2. System sjekker om bruker er innlogget og sender bruker til innlogging eller hovedsiden om brukeren er autentisert. |
| 3. Bruker logger inn med brukernavn og passord. | 4. System kobler seg mot DSpace og sender brukeren til hovedsiden hvis passord og brukernavn var riktig, hvis ikke skal brukeren informeres om feil bruker/passord. |

| Use Case: | **Slipp/Velge filer** |
|---|---|
| Aktør: | Bruker |
| Mål: | Velge filene som skal lastes opp. |
| Sammendrag: | Brukeren skal ha et felt hvor bruker kan slippe filer fra lokal pc. Bruker skal også ha en knapp som viser en vanlig fildialog slik at bruker kan velge filer om dra/slipp metoden ikke er støttet eller ønsket. Flere filer skal kunne velges/slippes samtidig. |
| Forutsetninger: | Bruker må være logget inn i EasyUp. |
| Ferdigtilstand: | Filer er tilgjengelige for EasyUp, og informasjon om filer vises til bruker. |

| **Handlingsforløp** | |
|---|---|
| Aktør | System |
| 1. Bruker slipper eller drar filer fra mappestruktur på klientmaskin. | 2. System tar i mot filer og viser disse til bruker i nettleser med relevant informasjon om filnavn/type og størrelse. |

| Use Case: | **Legg inn metadata** |
|---|---|
| Aktør: | Bruker |
| Mål: | Få samlet inn nødvendig tilleggsinformasjon om publiseringen. |
| Sammendrag: | Hvem hva hvor er stikkord. hvilke personer/fag/gruppe/rådgiver samlingen er knyttet til. Beskrivelse og tittel på samling kontaktinformasjon til bruker og språk på rfc1766 standard (vanlig skrivemåte på nett og DSpace sitt valgte språkformat) all informasjon blir søkbar i DSpace. |
| Forutsetninger: | Bruker må være Innlogget |
| Ferdigtilstand: | Informasjonen er tilgjengelig for EasyUp |

| **Handlingsforløp** | |
|---|---|
| Aktør | System |
| | 1. Systemet presenterer grensesnittet hvor bruker kan fylle inn relevant informasjon. |
| 2. Bruker fyller inn informasjon. | |

| Use Case: | **Velge samling** |
|---|---|
| Aktør: | Bruker |
| Mål: | Velge publiseringssted i DSpace. |
| Sammendrag: | Bruker må velge mellom tilgjengelige publiseringssteder i DSpace. |
| Forutsetninger: | Bruker må være Innlogget. |
| Ferdigtilstand: | Informasjonen er tilgjengelig for EasyUp. |
| Andre krav | Det skal være mulig å publisere per avdeling både åpent og lukket. |

| **Handlingsforløp** | |
|---|---|
| Aktør | System |
| | 1. Systemet presenterer bruker med en liste over samlingene som brukeren har rettigheter til å publisere i. |
| 2. Bruker velger ønsket samling fra liste. | |

| Use Case: | **Velge lisensiering** |
|---|---|
| Aktør: | Bruker |
| Mål: | Velge betingelser samlingen skal publiseres med. |
| Sammendrag: | Bruker skal velge mellom et forhåndsbestemt utvalg av publiseringsbetingelser. Informasjon om detaljer skal være lett tilgjengelig for bruker. |
| Forutsetninger: | Bruker må være Innlogget. |
| Ferdigtilstand: | Informasjonen er tilgjengelig for EasyUp. |

| **Handlingsforløp** | |
|---|---|
| Aktør | System |
| 1. Bruker velger ønsket lisensiering fra liste. | 2. Systemet assosierer valgt lisens med lisensfil tilgjengelig på server. |

| Use Case: | **Publisere filer** |
|---|---|
| Aktør: | Bruker |
| Mål: | Filene skal publiseres på DSpace. |
| Sammendrag: | Når nødvendig informasjon og filer er valgt av bruker skal bruker kunne publisere samlingen. |
| Forutsetninger: | Bruker må være Innlogget, minst en fil må være valgt og nødvendig tilleggsinformasjon må være fylt inn. |
| Ferdigtilstand: | Samlingen med tilleggsinformasjon er lastet opp på server. |
| Andre forutsetninger | |

| **Handlingsforløp** | |
|---|---|
| Aktør | System |
| 1. Bruker trykker på knapp for publisering. | 2. Filer lastes opp og opprettes på server. |
| | 3. Publiseringen startes. |

| Use Case: | **Publisere filer med mulighet for fortsettelse av avbrutt opplastning** |
|---|---|
| Aktør: | Bruker |
| Mål: | Filene skal publiseres på DSpace |
| Sammendrag: | Når nødvendig informasjon og filer er valgt av bruker skal bruker kunne publisere samlingen, hvis filene eller deler av dem er lastet opp på serveren skal de fortsettes på. |
| Forutsetninger: | Bruker må være Innlogget, minst en fil må være valgt og nødvendig tilleggsinformasjon må være fylt inn. |
| Ferdigtilstand: | Samlingen med tilleggsinformasjon er lastet opp på server. |
| Andre forutsetninger | Minst en fil må være lastet opp eller delvis lastet opp fra før uten at den/de har blitt publisert i DSpace. |

| **Handlingsforløp** | |
|---|---|
| Aktør | System |
| | 1. Systemet sjekker serveren for ikke-publiserte filer, og viser informasjon om dette til bruker. |
| 2. Bruker kan velge å fortsette opplastningen av den/disse filene. | 3. Systemet deler opp og fortsetter på filen samt overfører ny tilleggsinformasjon(metadata, samling, lisens). |
| | 4. Publiseringen startes. |

| Use Case: | **Logge ut** |
|---|---|
| Aktør: | Bruker |
| Mål: | Brukeren skal avautentiseres fra server. |
| Sammendrag: | Bruker kan trykke på en knapp med det resultat at bruker ikke lenger er autentisert. Ny autentisering skal kreves før videre bruk av løsningen skal tillates. |
| Forutsetninger: | Bruker må være Innlogget. |
| Ferdigtilstand: | Bruker er IKKE innlogget. |
| **Handlingsforløp** | |
| Aktør | System |
| 1. Bruker trykker på logg ut. | 2. Systemet gjør sitt for å fortrenge bruker. |
| | 3. Bruker blir presentert med innloggingsfelter. |

| Use Case: | **Publisere samling i DSpace** |
|---|---|
| Aktør: | DSpace |
| Mål: | DSpace skal ha samlingen publisert i sin database |
| Sammendrag: | Når nødvendige data er lastet opp på server skal samlingen publiseres i DSpace. |
| Forutsetninger: | Nødvendige data og filer må være lastet opp på server |
| Ferdigtilstand: | Samlingen er publisert |
| **Handlingsforløp** | |
| Aktør | System |
| | 1. Systemet starter publiseringen |
| 2. DSpace importerer/publiserer samling | |
| 3. DSpace informerer systemet om at samlingen er publisert. | |

## 2.4 Operasjon

En av de viktigste operasjonelle kravene er at systemet skal være responsivt og meget lett å bruke. Systemet skal håndtere flere samtidige brukere.

**Feilhåndtering**
Vi tenker at ikke kritiske feil skal skjules for brukere, men logges. Kritiske feil skal rapporteres til bruker og logges på server. DSpace logger sine egne feil.
Ikke kritiske feil defineres som ”feil som ikke hindrer handlingsflyten i publiseringsprosessen.” Kritiske feil defineres som ”feil som hindrer eller avbryter publiseringsprosessen.”

Oppetiden er i stor grad avhengig av serverdriften, noe som er utenfor vår kontroll. Når det er sagt så prioriteres god kode høyt, og det som kan kjøres på klienten, skal kjøres på klienten for å avlaste server.

**Livssyklus**
Løsningen skal ikke vedlikeholdes av oss. På grunn av at vi benytter oss av HTML5, må det påregnes noe etterutvikling og tilpasning. Dette fordi det er en standard under utvikling.

Løsningen skal kunne flyttes delvis eller helt. Delvis flytting kan skje om ønskelig, ettersom EasyUp webgrensesnitt ikke trenger å være på samme fysiske eller logiske server som DSpace.

**Ytelse**
Det at løsningen skal være brukervennlig innebærer blant annet rask respons og kjapp lasting av sider og menyer. Selve publiseringsprosessen i DSpace kan ta litt tid da store filer skal flyttes, dette blir en kontrast til den responsiviteten som EasyUp skal ha ellers, og brukere må derfor informeres tilstrekkelig om at publisering kan ta tid.

**Grensesnittdesign**

Nåværende publiseringsløsning har syv ulike steg for opplasting. Vi skal få disse ned til maksimalt tre steg i tråd med kravet om brukervennlighet.
Siden vi skal bruke ny nettleserteknologi, har vi muligheten til å gjøre flere av prosessene som tidligere var delt opp i steg, asynkront. Det vil si at dataoverføring mellom server og klient foregår i bakgrunnen og sparer brukeren for tiden det tar å navigere mellom ulike sider.
I tillegg vil brukeren slippe å lære seg plasseringen av funksjonalitet på flere ulike undersider.

Grensesnittet til EasyUp skal være modulært oppbygget og utformet etter handlingsflyten for opplastningsprosessen. Brukeren skal bevege seg fra venstre topp til høyre bunn, fylle ut og velge alternativer. Viktige eller påbudte valg skal merkes visuelt med en farge, for lett å tiltrekke seg brukerens oppmerksomhet.

Designet skal gjøres i CSS og skal være lett å endre på. Vi er ikke kvalifiserte designere og oppdragsgiver skal derfor ha mulighet til å utforme den visuelle stilen han selv ønsker.

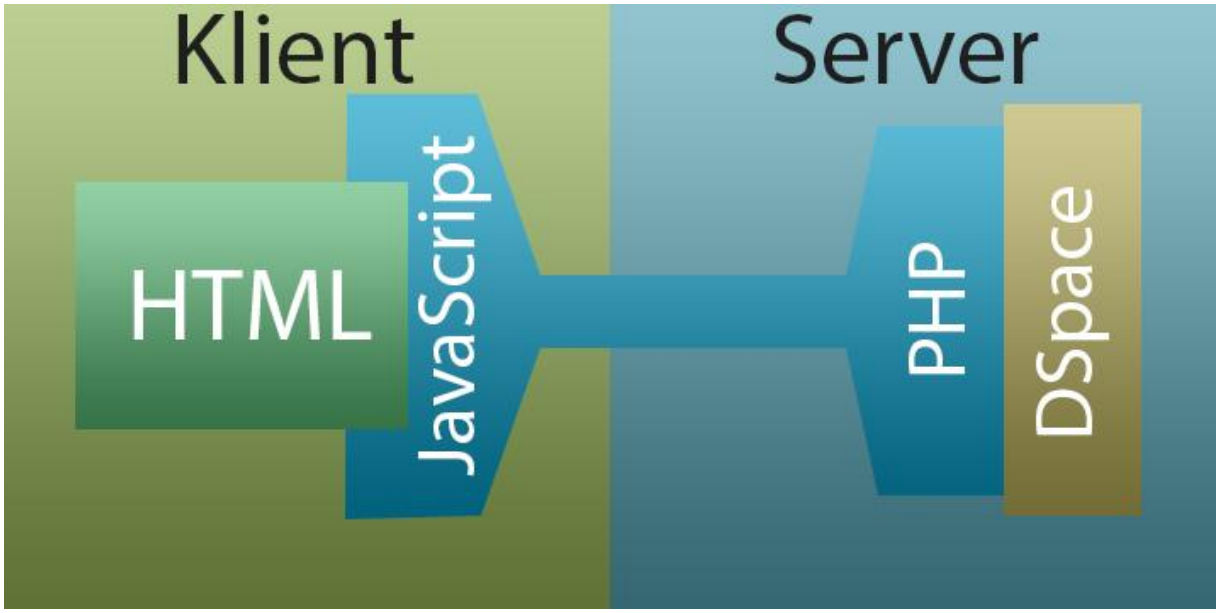
# 3.0 Design

Vi designet EasyUp I to ”faser”. Første fase bestod i å kartlegge hovedfunksjonalitet og bestemme oss for hvordan vi skulle dele opp, og angripe modulene. Denne fasen kaller vi grovdesign. Det meste av de store beslutninger angående design ble tatt i denne planleggingsfasen. Detaljdesign av moduler og medfølgende avvik fra grovdesign ble gjort pr. modul, underveis i implementasjonsprosessen.

Vi valgte denne inkrementelle utviklingsmåten fordi, vi på godt norsk ikke visste helt hva vi skulle lage når vi startet utviklingen av EasyUp. Den andre fasen kaller vi for moduldesign. Designdelen i rapporten inneholder informasjon om begge disse designfasene og beslutningene som ble tatt i dem.

## 3.1 Arkitekturbeskrivelse

Siden vi har en webløsning er løsningen vår en klient - server løsning. Denne består av en DSpace server, en webserver og klienter med nettlesere. Vår første designbeslutning bestod i å velge oss en arkitektur, først logisk, så fysisk.




**Figur 2 - Logisk arkitektur**

Den logiske arkitekturdelen er vist ovenfor. Kommunikasjon foregår med HTML og JavaScript til server. PHP knytter vår løsning sammen med DSpace. I tillegg blir PHP brukt til diverse oppgaver som må utføres på server. Valg av programmeringsspråk diskuteres i detalj i del 5 om implementeringen.

Fysisk valgte vi å bruke en og samme server for web og DSpace for å spare intern båndbredde, og øke hastighet. Denne beslutningen er reversibel og kan endres om ønskelig. Det kan for eksempel være praktisk å dele web og DSpace om publiseringene er mange, men små i størrelse. En slik trend vil øke belastningen for datakraft, men minske kravene for overføringshastighet i forhold til få publiseringer av stor filstørrelse.

Det er først og fremst medielinjen på skolen som har behov for en slik server pr. dags dato og de vil bruke den til hovedsakelig video, lyd og bildesamlinger (les store filstørrelser).

Det er ønskelig å dytte flest mulige arbeidsoppgaver over på klienten slik at løsningen blir så skalerbar som mulig.

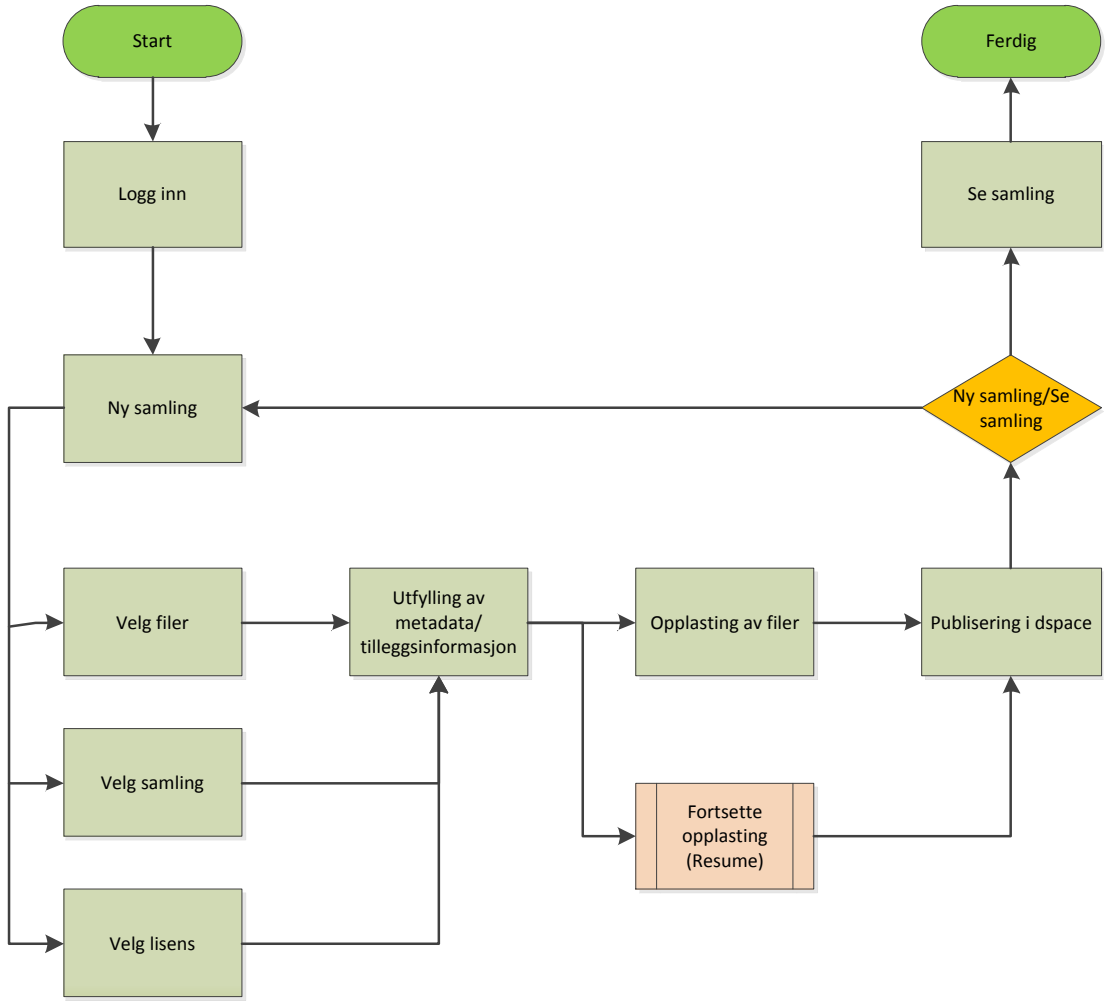
## 3.2 Moduler

Under grovdesignet delte vi den essensielle funksjonaliteten inn i moduler. De første modulene vi valgte oss var innlogging, velg/slipp filer, velg samling, og last opp til server. (sistnevnte var en tidlig versjon av publiseringsmodulen.)

Som vist i handlingsflytdiagrammet figur 3 er dette starten på hovedfunksjonaliteten i løsningen. Med hovedfunksjonalitet menes korteste vei fra klient til DSpace. Denne funksjonaliteten var viktig å få på plass tidlig på grunn av usikkerhet rundt støtte og fremgangsmåte på ulike nettlesere. Her beveger desigdelen seg over på implementeringsbiten av utviklingen, dette viser godt utfordringene med ny teknologi.

Samling, lisens og metadata ble lagt til senere da disse er DSpace-spesifikke og ikke er avhengig av nettleserteknologi direkte.
"Velg filer" i form av dra og slipp funksjonalitet, "Publisering i DSpace", og "Fortsette opplasting" er de modulene det var størst usikkerhet rundt implementeringen av.

## 3.3 Handlingsflyt for bruker




**Figur 3 - Handlingsflyt**

### Logg inn

Beskrivelse:

Brukeren logger seg på med brukernavn, passord og autentiseres i systemet.

Deretter blir bruker videresendt til hovedside.

Databehandling:

Innloggingsinformasjon tas i mot, renses og brukes til å autentisere bruker mot DSpace. Hvis brukeren blir autentisert opprettes brukermapper og alt av verdier på server. Dokumentet med brukers rettigheter i DSpace blir så mottatt og informasjon om hvor bruker har lov å publisere blir hentet ut. Deretter videresendes bruker til hovedsiden.

**Ny samling**

Beskrivelse:

Webgrensesnittet klargjøres for ny publisering.

Databehandling:

Ny samling modulen kjøres da hovedsiden lastes inn. Metadata om bruker blir hentet ut av LDAP server. Metadataformen blir laget, og fylt inn med forhåndshentede verdier. Samlingene brukeren har lov til å publisere i, blir vist i en liste. Påbudte felter blir sjekket og merket med rødt om de ikke er fylt inn, eller tilfredsstiller gitte krav. I tillegg sjekkes brukerens mappe på serveren for gjenværende filer. Filstørrelse og navn på gjenværende filer blir vist til bruker og "lagret" i nettleseren med informasjon om fortsettelses funksjonalitet.

**Velg filer**

Beskrivelse:

Filer kan velges eller slippes inn i nettleser og blir etter det vist i nettleseren. Etter at filer er valgt, vises boksen for metadata til brukeren.

Databehandling:

Filer tas i mot enten ved at de blir sluppet inn i nettleseren eller ved at bruker velger filer. Om filene er bildefiler, blir det generert en liten forhåndsvisning av bildene. Denne blir vist ved siden av navn, størrelse og filtype i fillisten. Hvis filene er av et annet format blir et lite ikon for filtypen vist i stedet.  Når fillisten er ferdig vises metadataformen. Denne har vært skjult hittil for å styre brukerens oppmerksomhet mot filvelgeren.

**Velg samling**

Beskrivelse:

Samling velges fra listen

Databehandling:

I det samlingen velges, legges denne automatisk inn som en verdi på serveren. Denne verdien blir brukt senere under publiseringen.

**Velg lisens**

Beskrivelse:

Tilgjengelige lisenstyper blir vist og bruker kan lese mer om, eller velge lisensen de ønsker å publisere med.

Databehandling:

Valget som er tatt blir hentet ut under publiseringen og sendt til server. På serveren blir den valgte lisensen lagt med samlingen når den publiseres.

Utfylling av metadata

Beskrivelse:

Metadata kan endres (om allerede utfylt) eller skrives inn. Flere felter med metadata kan legges inn hvis dette er ønskelig.

Databehandling:

Automatisk innhentet metadata er fylt inn i formen. Resterende metadata kan fylles inn. Hver gang en endring skjer i metadataformen blir den validert på nytt og ulovlige felter blir merket. Språkfeltet er ett av metadatafeltene og består av et søkefelt og en liste, slik at språk kan søkes opp etter navn eller forkortelse. Forkortelsen blir brukt videre under publisering.

Det er knapper ved siden av som kan brukes til å legge til tilleggsfelt hvis mer informasjon er ønskelig ved publisering. Knappen blir da usynlig og et nytt felt blir lagt til i metadataformen. Validering av samtlige felt i formen blir utført etter hvert tastetrykk. E-post valideres etter oppbygning.

**Opplasting av filer**

Beskrivelse:

Hvis filer er valgt, nødvendig metadata fylt ut og samling er blitt valgt, er det mulig å starte opplastingen. Filer, metadata og samling blir da lastet opp til server. Etter at opplastingen er fullført starter publiseringen i DSpace.

Databehandling:

Når alle kriterier for opplasting er møtt, aktiveres "last opp" knappen. Denne kan brukeren trykke på. Når knappen trykkes sendes alle feltene i metadata pluss dagens dato, informasjon om ulike filtyper og valgt lisens inn til serveren.

Serveren lagrer denne informasjonen på filer i brukermappen. Dette gjøres i et format som er påkrevd av DSpace sin importeringsfunksjon. Parallelt med dette lastes filene opp. Når filene er lastet opp starter publiseringsprosessen.

**Fortsette opplasting**

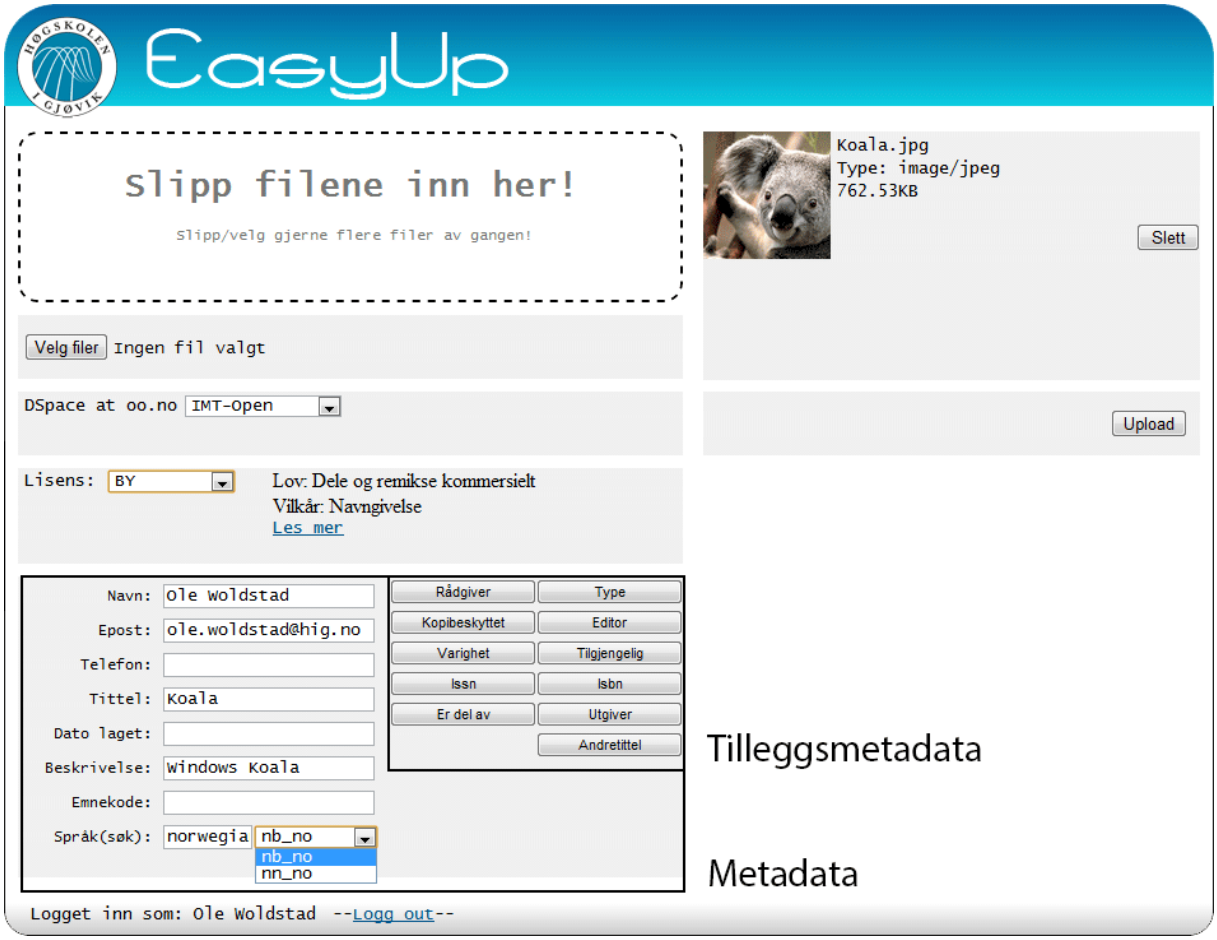
Beskrivelse:

Hvis filene finnes fra før på serveren skal filene deles på klientmaskinen og fortsette der de slutter på serveren, slik at ingen informasjon blir overført på nytt. Ikke-fullførte filer kan oppstå når tidligere opplastinger har blitt avbrutt.

Databehandling:

Når filene skal lastes opp, sjekkes hver fil mot oversikten som ble laget under "Ny samling" modulen. Hvis den aktuelle filen eksisterer blir den delt og skjøtet med filen som allerede befinner seg på serveren. Hvis hele filen ligger der og ikke bare en del, betyr bare dette at filen starter på 100 %, noe ny data blir da ikke overført.

Hvis filen ikke finnes i listen over filer på server lastes den bare opp som normalt. Denne prosessen skjer per fil slik at all metadata og publisering skjer på samme måte som i "Opplasting av filer", som dette er en undermodul av.




**Figur 4 - EasyUp oversikt over metadata**

**Publisering i DSpace**

Beskrivelse:

Samlingen med tilhørende informasjon publiseres i valgt samling på DSpace serveren.

Databehandling:

Når filene er lastet opp på webserveren, og hjelpefilene er laget i tråd med DSpace sine krav startes filimporteringen.

Her blir lisensfil i tillegg til alle brukeropplastede filer publisert til DSpace ved hjelp av DSpace egen importfunksjon. Når denne er ferdig slettes filene i brukermappen i EasyUp.

**Ny samling/Se samling.**

Når publiseringen er gjort uavhengig av om den lykkes eller ikke, får bruker valg mellom å se de siste publiseringene som er gjort på DSpace, eller å lage en ny publisering. Svakhetene rundt dette diskuteres senere i oppgaven.

## 3.4 Grensesnittdesign

Den røde tråden i denne oppgaven har vært brukervennlighet. Dette ordet befinner seg til og med i oppgavebeskrivelsen. Mangel på brukervennlighet er grunnen til at det finnes et behov for EasyUp. I dette underpunktet skal vi skrive litt om hva slags valg vi har gjort i forbindelse med brukervennlighet og grensesnittdesign.

Oppdragsgiver ønsket et brukergrensesnitt som var både intuitivt, lett å bli kjent med og hurtig å bruke. Et naturlig resultat av disse målene var å kutte ut unødvendige steg.
Vi startet med kun det mest nødvendige for publisering og bygget på dette.

Den opprinnelige publiseringsløsningen til DSpace er modulært oppbygget og består av 7 ulike trinn. Hvert av trinnene har sin nettside og det er nødvendig å navigere mellom disse for å gjøre endringer i publiseringen.
Vi puttet all utfyllbar informasjon på én side, og minimerte eller skjulte det av informasjon som vi vurderte som mindre relevant.

Når det gjelder handlingsflyten i siden, benyttet vi oss av den allerede innøvde normen hos de fleste lesedyktige. Informasjonsflyt fra øverst til venstre, til nederst mot høyre i to kolonner.

**Filvelger**
Filvelgeren består av to bokser, en hvor filer kan slippes inn, og en med en knapp som starter en fildialog. Disse kan brukes om hverandre, til å velge filene som skal publiseres.

Fordelen med "slippboksen" er at filer kan velges fra mappen du allerede jobber i og har åpen på din arbeidsmaskin. Du slipper derfor å bruke tid på å navigere på nytt til riktig lokasjon. En dra og slipp metode for velging av objekter er også mer intuitiv. Dette fordi den ligger nærmere måten vi manipulerer gjenstander på i den virkelige verden.

**Valg av lisens og samling**

Her har vi valgt en nedtrekksliste. Dette sparer oss for plass og er fleksibelt fordi det ser likt ut uavhengig av hvor mange elementer listen inneholder. Under samlinger vil dette variere avhengig av brukerens rettigheter.

Vi er også bare interessert i å velge ett element fra hver liste, så antall musetrykk blir kun to pr. boks. Andre ting vi har gjort for hurtigere utfylling og orientering er å merke feltet for samling med rødt hvis bruker ikke har valgt en samling. Vi har også standardisert lisensvalg til "copyright".

**Metadata**

Å få med fornuftige metadata på forholdsvis liten plass var en stor utfordring. Her sparer vi tid ved å forhåndsutfylle mest mulig informasjon, navn e-post og telefonnummer om tilgjengelig. I tillegg går informasjon om filformater og dato publisert, i bakgrunnen for å spare plass.

Felter som er påbudte merkes med rødt for å kommunisere at disse må fylles ut, opplastningsknappen er også deaktivert til nødvendige felter er valgt eller fylt inn. En administrator kan lett endre påbudte felt i koden.

Under språkvalg vil DSpace kun ta i mot et standardisert forma på formen "en_US" (Engelsk USA). For å unngå å vise en liste med 122 språkvarianter delte vi språkvalget opp i to felt, et for søk og en nedtrekksliste.

Hver gang en ny bokstav tastes inn i søkefeltet begrenses nedtrekkslisten til kun å vise søketreff. Dette skal sørge for god oversikt over valgmuligheter og gjøre det raskt og enkelt å velge språk. Språk er søkbare både på språkkode og engelsk språknavn eksempelvis "german" og ikke "deutsch" eller "tysk". Dette gjør at alle språk lett kan finnes på maksimalt to tastetrykk og tre musetrykk. Norsk bokmål for eksempler kan velges på ett musetrykk og to tastetrykk.

Tilleggsmetadata inneholder felter som kan legges til i metadataformen. Disse dukker opp som et nytt felt under metadata når de blir valgt.

Tilleggsmetadata-knappene blir usynlige når de blir trykket på. De beholder posisjonen sin og etterlater hulrom når de blir valgt.

Dette er gjort på denne måten slik at man skal slippe å orientere seg på nytt for hvert ekstra felt som blir valgt (knappene trekker seg ikke sammen og bytter posisjon). I tillegg blir det færre knapper å se på, for hvert trykk. Å legge til nye felt skal være nesten like tilfredsstillende som å trykke bobleplast.

Feltene er representer som knapper og ikke nedtrekksliste fordi det med en nedtrekksliste ville medført to trykk pr. valgte tilleggsfelt. En nedtrekksliste begynner erfaringsmessig å bli uoversiktlig når den inneholder mer enn 5 elementer.

Hele metadataformen skjules inntil minst en fil er valgt. Dette er gjort for å bedre oversiktlighet og rette fokus mot dra og slipp boksen.

**Vis filer og last opp knapp.**

Filene blir vist med et forholdsvis stort ikon med tilhørende informasjon om filnavn, filstørrelse og filtype/format plassert til høyre. Ikonet brukes til rask visuell identifisering av filtype eller aktuelt bilde, filinformasjon til mer detaljert informasjon. Det er også plassert en slettknapp for hver fil i listen slik at filer lett kan fjernes, om de ikke er ønsket i publiseringen. Plassen nederst bak ikonet brukes til å vise en fremdriftslinje for opplasting pr. fil. ved gjenoppretting, starter denne på punktet hvor filen ble avbrutt.

Opplastningsknappen er plassert nederst til høyre i høyre kolonne og markerer enden på handlingsflyten. En fremdriftslinje for total fremdrift på formatet "x-fil/totalt antall filer" vises til venstre for last opp knappen.

# 4.0 Bakgrunn

## 4.1 DSpace

DSpace er et digitalt arkiv for lagring og publisering av digitale filer. DSpace er et kryssplattformprodukt lansert i åpen kildekode. DSpace brukes av mange institusjoner over hele verden. Alt ifra store universiteter til mindre organisasjoner som trenger en plattform for sikkert lagring og enkel deling av digitalt arbeid.

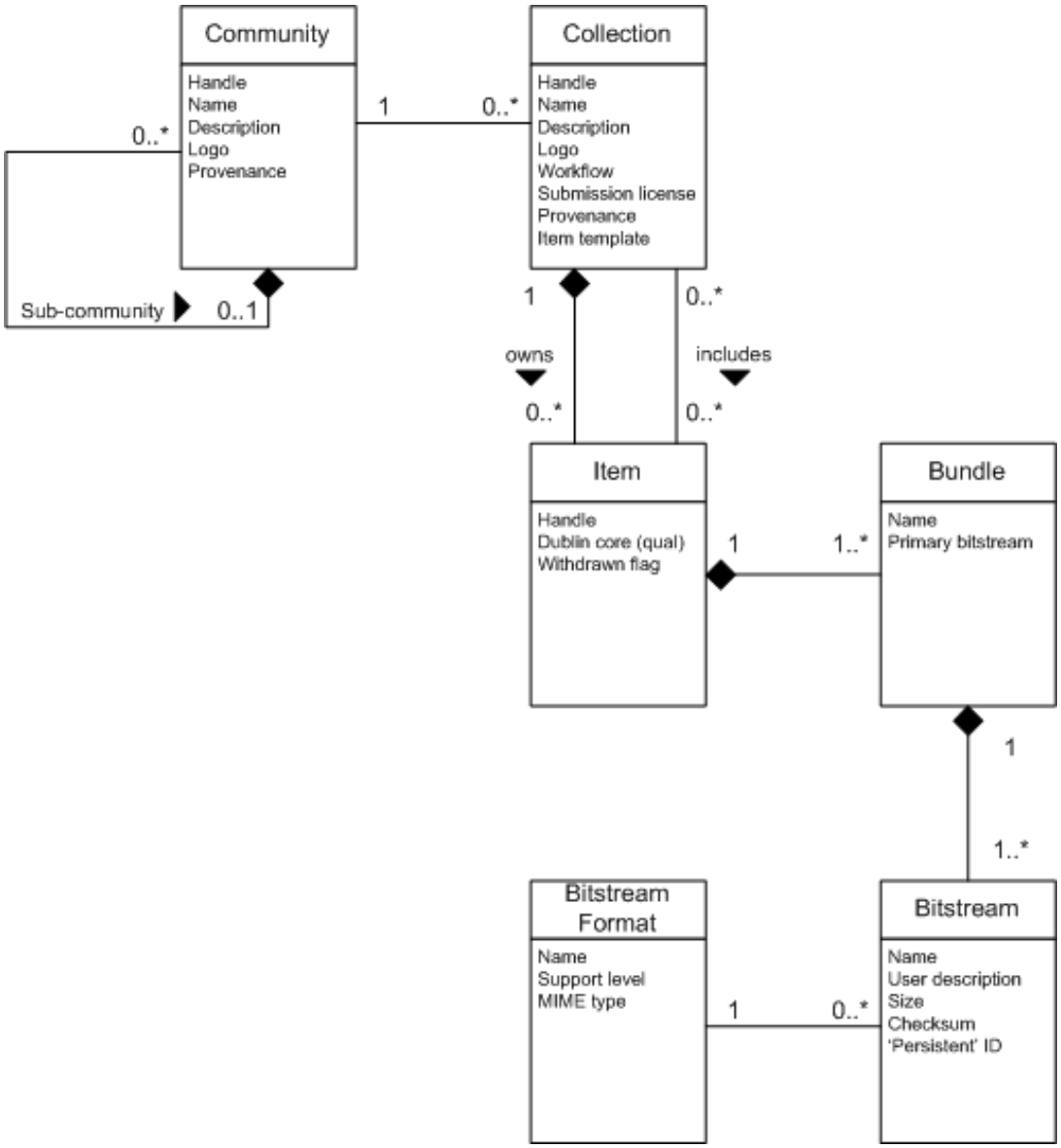
### 4.1.1 Historie

Den første versjonen av DSpace ble lansert i 2002 som et samarbeid mellom utviklere i HP og fra MIT. I de neste årene var styring gjort via diverse interesserte parter. I 2007 ble det opprettet en ikke-profitt organisasjon kalt ”DSpace Foundation” for å forbedre styringen av utviklingen og support.
12 mai 2009 gikk ”DSpace Foundation” og ”Fedora Commons” sammen og dannet DuraSpace. En ny ikke-profitt organisasjon som i dag står for utvikling og support av DSpace.

### 4.1.2 Funksjonalitet

DSpace er skrevet i Java, og krever en relasjonsdatabase for lagring av data. For å serve DSpace til brukere brukes Apache Tomcat som er en Java applikasjonstjener. Det kreves også verktøy for å installere DSpace som: Apache Ant, Apache Maven og Java JDK

Data lagret i DSpace er organisert på følgende måte:




**Figur 5 - DSpace oppbygning**

Det kan være ett eller flere communities som f. eks HiG, UiO eller NTNU.
Disse inneholder igjen en eller flere collections, gjerne avdelinger eller andre grupperinger.
Her vil det være naturlig å ha flere collections med ulike rettigheter for lesing og skriving.

I hver collection kan man publisere mange items. Hvert item kan inneholder en eller flere filer, her referert som bitstreams. Hvis flere filer utgjør en større fil sammen, f. eks html kode og bilder tilhørende en nettside, samles alle bitstreams sammen til en bundle.
Bitstream-format angir hvilken filtype det er snakk om.
Hvert item kan inneholde mye metadata som blir søkbare identifikatorer i DSpace.

**Sikkerhet**

DSpace bruker E-People og Grupper av E-People for å håndtere rettigheter. Rettigheter kan altså settes på enkeltpersoner eller grupper.

DSpace støtter flere ulike måter for å autentifisere brukere:

- Lokalt brukernavn og passord (Manuell registrering)
- Shibboleth
- LDAP
- IP adresser (Kun IP fra nevnte adresser kan autentifisere)
- X.509

Alle metodene støtter automatisk gruppemedlemskap og opprettelse av bruker ved første innlogging.

I tillegg støtter DSpace såkalt ”Stackable Authentification”. Det vil si at man kan kjøre flere typer autentifisering samtidlig og dermed ha flere kilder brukere kan autentifisere seg mot.

**Handles**

For å gi brukere en fast link som aldri endres bruker DSpace ett handle system laget av ”Corporation for National Research Initiatives"[11]. Denne linken vil alltid linke til samme fil selv om DSpace skulle endre dens plassering.
For å kunne bruke dette må hver instans av DSpace ha en egen prefiks som identifiserer serveren. Dette oppnås ved å søke CNRI om en unik prefiks for den spesifikke instansen.

### 4.1.3 Importering av filer

I tillegg til sin vanlige publiseringsmetode har DSpace kommandolinjeverktøy for importering av filer. Vårt system bruker nettopp denne metoden for publisering av filer.

Dette verktøyet krever at filene som skal importeres er lagt opp i ”DSpace Simple Archive Format”. DSAF har følgende struktur av filer med følgende innhold:

*Arkiv/*

*Item1/*

*Dublin_core.xml*

*Contents*

*License*

*Fil1.pdf*

*Fil2.jpg*

*Item2/*

*Dublin_core.xml*

*Contents*

*License*

*Fil1.doc*

*Fil2.rar*

*Fil3.avi*

*…*

Hvert item har sin egen mappe med følgende filer:

### Dublin_core.xml

Denne XML filen består av metadata i dublin core formatet. Denne informasjonen brukes for å beskrive publiseringen. For en liste over alle mulige metadata se vedlegg G.

Her ett eksempel på Dublin_core.xml:

```
<dublin_core>
   <dcvalue element="contributor" qualifier="author">Ola Kjelsrud</dcvalue>
   <dcvalue element="identifier" qualifier="other">ola.kjelsrud@HiG.no</dcvalue>
   <dcvalue element="identifier" qualifier="other">12345678</dcvalue>
   <dcvalue element="title" qualifier="none">Bacheloroppgave</dcvalue>
   <dcvalue element="date" qualifier="created">23/05-2012</dcvalue>
   <dcvalue element="description" qualifier="abstract">Bacheloroppgave</dcvalue>
   <dcvalue element="subject" qualifier="other">IMT3912</dcvalue>
   <dcvalue element="language" qualifier="iso">nb_no</dcvalue>
```

**Figur 6 - Dublin_core.xml eksempel**

Her kan bruker velge hvor mye informasjon som skal legges med i publiseringen. Alle metadata er søkbare i DSpace så lenge bruker har rettigheter til dette.

### Contents

Denne filen inneholder filnavn som skal inn i publiseringen. Altså: Fil1, Fil2, Fil3, osv.

Her er ett eksempel på Contents:
*License*
*Hydrangeas.jpg*
*Jellyfish.doc*
*Chrysanthemum.jpg*
*Desert.pdf*

**License**

License er en valgfri fil. Vi har likevel valgt å bruke denne filen for lisensinformasjon. Bruker velger hvilken lisens han/hun vil publisere med i EasyUp og lisensinfo blir lagt inn i License filen som tekst.

**Fil1, Fil2, Fil3, …**

Dette er selve filene som skal publiseres (Bitstream). Det er ingen grense på antall filer, og DSpace støtter de alle fleste formater. Se vedlegg G.

### 4.1.4 Installasjon og konfigurasjon

Som nevnt over krever DSpace flere verktøy for å kunne installeres og kjøres.
Her forklarer vi hvilke valg vi har tatt under installasjonen.

Vi kjører serveren på Linux plattform. I vårt testmiljø har vi brukt Debian 6.
Debian ble valg pga stabilitet og et meget bra pakkebehandlingssystem.
Vi valgte PostgreSQL som relasjonsdatabase over Oracle pga tilgjengelighet. Vi trenger ikke noe spesielle funksjoner fra Oracle slik at en gratis relasjonsdatabase ble naturlig.

For å kunne bygge og kjøre DSpace har vi brukt følgende:

- Java JDK 6
- Apache Maven 2.2
- Apache Ant 1.8
- PostgreSQL 8.4
- Apache Tomcat 6

Vi har brukt APT, ett pakkebehandlingssystem for installasjon av disse verktøyene. For detaljerte installasjonsinstrukser anbefales verktøyenes hjemmesider.

For detaljerte instrukser om DSpace installasjon anbefales:
DuraSpace dokumentasjon[ 6]

Her er noe av de viktigste konfigurasjonspunktene vi har valgt.

**LDAP autentifisering:**

*plugin.sequence.org.DSpace.authenticate.AuthenticationMethod = *

*org.DSpace.authenticate.LDAPHierarchicalAuthentication*

*provider_url = ldaps://hig1.HiG.no:636/*

Vi bruker LDAPS for sikker kommunikasjon mellom DSpace og LDAP server for å autentifisere brukere.

**Automatisk brukeropprettelse og gruppemedlemskap:**

*autoregister = true*

*login.specialgroup = Submit-Open*

Hvis bruker ikke finnes fra før opprettes bruker automatisk med informasjon hentet fra LDAP. Brukeren blir også automatisk medlem i en gruppe som har rettigheter til å publisere i diverse collections.

**Handle**

*handle.prefix = 1337*

Som forklart tidligere må hver DSpace instans søke om handle prefiks fra CNRI. Vi har derfor valgt en midlertidig manuell handle, 1337.

**E-post**

*mail.server = smtp.HiG.no*

*mail.from.address = DSpace-noreply@HiG.no*

*feedback.recipient = DSpace-help@HiG.no*

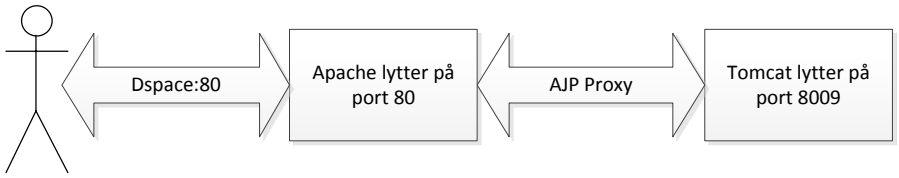
*mail.admin = DSpace-help@HiG.no*

E-post kan sendes fra DSpace på mange kriterier. Vi bruker HiG sin E-post server til dette.

### 4.1.5 Kjøring av DSpace

Tomcat brukes som webserver for å serve DSpace til brukere. Tomcat kjøres derimot av en ikke privilegert bruker slik som Apache. Dette medfører at Tomcat ikke kan lytte på såkalte kjente porter (0-1023). For å komme rundt dette problemet bruker vi ”Apache JServ Protocol”. En protokoll som fungerer som en proxy. Apache og Tomcat blir dermed knyttet sammen og brukere kan bruke port 80 som normalt.

Vi får derfor følgende dataflyt:




**Figur 7 - Dataflyt for DSpace tilkobling**

### Tomcat konfigurasjon:

*ln -s DSpace/webapps/xmlui tomcat6/webapps/ROOT*

Vi setter altså DSpace sin xmlui til Tomcats default servlet. Dette gjøres slik at vi kan bruke http://DSpace.HiG.no i stedet for http://DSpace.HiG.no/xmlui

### Apache konfigurasjon:

Vi må sørge for at AJP modulen er tilgjengelig

*a2enmod proxy_ajp*

I tillegg lager vi er virtuell host i Apache:

```
NameVirtualHost *:80
<VirtualHost *:80>
        ServerAdmin DSpace@HiG.no
        ServerName DSpace.HiG.no
        ServerAlias DSpace
   <Proxy *>
     AddDefaultCharset Off
     Order deny,allow
     Allow from all
```

**Figur 8 - Apache virtuell host konfigurasjon**

## 4.2 HTML5

“HyperText Markup Language” er ikke et programmeringsspråk men et markeringsspråk for å beskrive innhold på nettsider. HTML bruker såkalte ”HTML Tags” som <h1>,  og <p> osv for å beskrive innhold. Disse taggene kan så brukes av CSS for å beskrive hvordan nettleseren skal presentere innholdet. Nettleseren viser altså bare innholdet i taggene til bruker. For mer informasjon og historie rundt HTML anbefales W3C [10].

HTML5 blir den nyeste standarden i rekken. Standarden er ikke ferdig utviklet. Dette medfører hyppige endringer i funksjonalitet. Det er viktig å ikke tenke på HTML5 som bare HTML språket. I realiteten er HTML5 en samling av mange forskjellige teknologier, funksjoner og API-er som jobber sammen for å bedre opplevelsen til både utviklere og brukere.
For mer informasjon om HTML5 anbefales HTML5 Rocks [12]

## 4.3 CSS

”Cascading Style Sheets” er et språk som brukes for å definere utseende på HTML filer. Mens HTML merker innhold, sørger CSS for hvordan nettlesere viser innholdet.
CSS kan f. eks definere en HTML tagg som *background-color: red;*. Dette vil si at alt innhold i disse taggene vil ha en rød bakgrunn i nettleser.

CSS3 er den nyeste standarden i rekken. Denne er heller ikke ferdig utviklet, men lider i mindre grad av stadige endringer. I motsetning til CSS2 som består av ett sort spesifikasjonsdokument består CSS3 av mer enn 50 forskjellige moduler som dekker forskjellige funksjonalitet. Modulene har forskjellige stabilitet støtte i ulike nettlesere.
For mer informasjon om CSS anbefales W3C [10].

## 4.4 JavaScript

JavaScript er et scriptspråk som har mange bruksområder. Det meste kjente er for å gi nettsider mer dynamisk funksjonalitet. Det er viktig å ikke forveksle JavaScript med programmeringsspråket java. Selv om de har enkelte likheter er de i realiteten helt uavhengige av hverandre i funksjonalitet og bruksområdet.

JavaScript benyttes på klient siden, det vil si at JavaScript kjøres lokalt på brukerens system og ikke på server slik som f. eks PHP. Det er JavaScript som gir oss tilgang til de nye HTML5 API-ene vi bruker i systemet. I tillegg er det JavaScript sammen med HTML, CSS og XML som danner grunnlaget for AJAX.

## 4.5 AJAX

”Asynchronous JavaScript and XML” består av flere forskjellige teknikker som sammen danner grunnlaget for asynkron kommunikasjon mellom klient og server. Med dette menes kommunikasjon mellom klient og server i bakgrunnen uten at nettsiden nødvendigvis endrer seg for bruker. Tanken er at nettsider skal gjøres responsive ved å utveksle litt og litt data med server istedenfor alt på en gang for hver endring brukeren gjør.

Selv om XML inngår i navnet trengs det ikke for å bruke AJAX som teknologi.

For å endre nettsider dynamisk manipulerer JavaScript DOM. ”Document Object Model” er en beskrivelse av hvordan et HTML eller XML dokument er bygd opp. JavaScript bruker DOM -APIet som inneholder funksjoner for å legge til, endre eller fjerne objektene i et dokument. JavaScript kan på denne måten endre HTML og CSS slik at utseende på nettsiden endres dynamisk.

For kommunikasjon mellom klient og server brukes som regel XMLHttpRequest objekter. XMLHttpRequest sender http- eller https-forespørsler fra klient til server. Dette skjer ofte asynkront i bakgrunnen uten av bruker er klar over det. Nettsiden oppleves dermed som dynamisk og interaktiv.

## 4.6 PHP

PHP eller "PHP Hypertext Preprocessor" er et serverside programmeringsspråk som brukes for å lage dynamiske nettsider. PHP-koden kan skrives i HTML kode eller i ekstern fil og kjøres av webserveren. Resultatet av koden blir lest av nettleseren som HTML.
Koding i PHP kan også brukes til å hente ut, flytte eller editere informasjon på en server, eller til å koble til databaser.

Vi befinner oss nå på hovedversjon 5 og mye er gjort med strukturen på språket, men hovedfunksjonaliteten er den samme.
Vi valgte PHP5 som serverside skriptspråk både fordi vi begge har erfaring med språket og fordi det er fritt i motsetning til Microsoft sitt alternativ ASP.

## 4.7 Oppsummering

HTML er formateringen som bygger en nettside. Nettsiden blir deretter ”sminket” med CSS. Funksjonalitet og dynamikk kan bygges med JavaScript og ved hjelp av AJAX får man laget dynamiske og interaktive sider via kommunikasjon til server.  PHP utfører oppgaver på serveren og kan presentere resultatet i HTML, eller kommunisere med JavaScript ved hjelp av Ajax funksjonalitet.

# 5.0 Implementering

I dette kapittelet forklarer vi hvordan vi har implementert funksjoner og hvordan vi har løst ulike problemer i løpet av prosjektperioden. Vi har delt forklaringer inn i 5 hovedpunkt: Innlogging, hovedside, filhåndtering, opplasting og publisering. Disse punktene forklarer hva som er gjort og hvordan EasyUp fungerer. I tillegg har vi punkter med sikkerhet, tankeprosess og SWORD, en alternativ importeringsmetode av filer inn i DSpace.

Vi forklarer i korte trekk de viktigste kodesnuttene der vi spesielt trekker frem ny og spennende funksjonalitet som kommer med de nye standardene. Kodesnuttene som er lagt med vil ofte være noe forenklet for lettere forståelse, koden er også kommentert på norsk. Denne delen av rapporten krever allikevel mer innsikt i programmering og nettsideteknologi enn de andre delene. Begreper som PHP, AJAX, JavaScript og HTML er beskrevet i kapittel 4.

## 5.1 Tankeprosess

Som forklart tidligere jobber vi med forholdsvis nye standarder, mange er enda ikke satt. Dette medfører at stadig ny funksjonalitet blir lagt til/ tatt bort/endret og vi kan ikke garantere at vi ikke blir berørt av dette. Vi har også kommet borti funksjonalitet der både gammel og ny kode er aktiv og fungerer.

Et eksempel på dette er EventListeners. Altså håndtering av forskjellige hendelser på nettsider som f. eks load, click osv. I disse tilfellene har vi vært klare på å bruke de nyeste standardene fra ”World Wide Web Consortium” (W3C) der dette lar seg gjøre.

## 5.2 Innlogging

Innloggingsprosessen baserer seg på et hittil udefinert begrep SWORD. Vi forklarer hva SWORD er og hvorfor vi har brukt dette senere i implementeringsdelen. Det er to ting som er viktig å merke seg om SWORD nå. DSpace presenterer et SWORDdokument som er skreddersydd etter brukers rettigheter. Dette er et XML-dokument.

Hele innloggingsprosessen håndteres av PHP og dette gir oss mulighet til å benytte oss av biblioteker for håndtering av XML og cURL for innlogging til DSpace.
cURL er et bibliotek for URL overføring og støtter blant annet FTP, HTTP, HTTPS, SFTP samt ulike mailprotokoller.

**Innloggingsprosessen**

1. Brukernavn og passord blir sendt til checklogin.php når logg inn knappen trykkes på. Overføringen skjer over POST. Dette gjør at informasjonen sendes i klartekst om ikke SSL er aktivert på server.
2. checklogin.php tar i mot brukernavn og passord og renser disse, se punkt 5.7.2 under sikkerhet.
3. Kobler til "DSpace server"/sword/servicedocument med brukernavn og passord for å hente servicedokumentet til brukeren. Her brukes cURL for å gjøre dette i bakgrunnen. Informasjon om tilgjengelige samlinger hentes ut fra servicedokumentet

Koden under er limt sammen av ulike utdrag. den er kun ment for å vise hovedtrekkene i hvordan cURL brukes for å håndtere tilkoblingen til DSpace.

```
//Kodeblokken henter og behandler dokumentet og bygger en liste
      cURL_setopt($swdoc_cURL, CURLOPT_USERPWD, $myusername . ":"
      .$mypassword);                                    //Setter brukernavn
                                                        //og passord
      $swdoc_url = "http://DSpace.oo.no/sword/servicedocument";
      $swdoc_resp = cURL_exec($swdoc_cURL);             //Henter ned filen
                                                        //Linjen under henter
                                                        //overføringsstatusen
      $swdoc_status = cURL_getinfo($swdoc_cURL, CURLINFO_HTTP_CODE);
                                                        //Hvis alt er ok...
...
//Avsnittet bygger en liste med samlingene brukeren har tilgang til
            if (buildXMLArray($swdoc_resp)) {       //Hvis riktig
```

**Figur 9 - cURL tilkobling til DSpace for autentifisering**

4. Ved hjelp av simpleXML blir XML-filen gått gjennom. To verdier er relevante; navn på samling og link til samling. Disse lagres i liste med to tilhørende verdier ["IMT-Open"]["666/4"] (navn med tilhørende link pr. oppføring).
5. Samlingens URL en blir strippet av, ved hjelp av en egen funksjon, dette gjøres slik at den er i riktig format før den brukes til å kjøre scriptet for importering. Filen "servicedokumentXMLParser.php" er en egenlagd fil som brukes til gjennomgang og bygging av collectionarray/samlingsliste.
6. Hvis tilkoblingen er vellykket og en liste med samlinger er laget, registreres en session med både brukernavn, passord og brukermappe. I tillegg opprettes brukermappe på server og rettigheter settes på denne. Denne doble sjekken sørger for at serveren godtar brukernavn og passord og at et gyldig XML-dokument ble overført.

**Feilhåndtering**

Ved feil i tilkobling får bruker beskjed om at feil brukernavn og passord er tastet og bruker kan prøve på nytt. Denne beskjeden ser brukeren også hvis det er problemer med kommunikasjon til DSpace eller autentiseringsserveren.

Hvis en fil blir mottatt men ikke har riktig struktur, vises en annen feilmelding hvor det informeres om feil ved XML-filen. Scriptet dør etter begge feiltilfellene.

## 5.3 Hovedsiden

Hovedsiden, index.php består hovedsakelig av et skall med rammer, tekst, knapper og bokser bygget med HTML. Det aller meste av utseende og funksjonalitet blir lagt til etter hvert ved hjelp av JavaScript.

JavaScript filen dnd.js inneholder to klasser. Klassen ”Upload” tar seg i hovedsak av filhåndtering og klassen ”Metadata” tar seg av input fra bruker. Det meste av funksjonalitet er ”handlingsstartet”. Dette betyr at kode blir kjørt enten når elementer er lastet inn, eller ved diverse brukervalg.

Eksempel på en hendelseslytter (Eventlistener):

```
collectionForm.addEventListener('change', checkButton, false);
      //Denne linjen lytter etter endringer I collectionForm og
      //kjører checkButton-funksjonen hver gang den oppdager noen.
      //Checkbutton sjekker alle feltene i formen og merker
      //obligatoriske felt som ikke er riktig fylt inn med rødt
```

**Figur 10 - EventListener eksempel**

To lister er hardkodet i dnd.js (se underliggende kodeboks) disse inneholder henholdsvis navn på felter som skal vises og felter det er mulig å legge til.

Når hovedsiden lastes inn, bygges innholdet i metadata med knapper knappene fra extradataArray, se fig 11. Dette slik at det skal bli enkelt å endre hva slags felter som skal være med, og hva slags felter som skal kunne legges til av bruker.

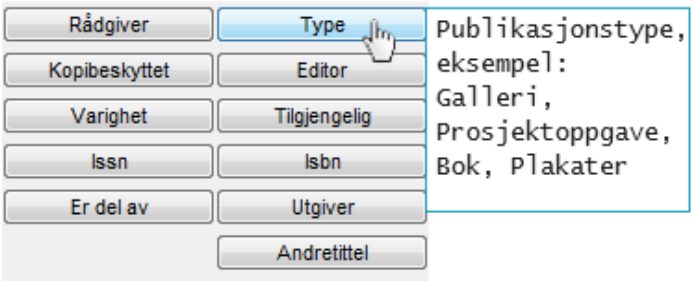
```
      //Ekstra metadata knapper(utdrag)
      extradataArray['Type']= "";
      extradataArray['Rådgiver']= "";
      extradataArray['Editor']= "";
      extradataArray['Kopibeskyttet']= "";

      //Metadata felter (utdrag)
      metadataArray['Navn']= "";
      metadataArray['Epost']= "";
      metadataArray['Telefon']= "";
      metadataArray['Tittel']= "";
```

**Figur 11 - Liste over mulige metadata**

Hvis felter skal legges til eller endres navn på, må de i tillegg endres i ”oversettingsfunksjonen” i uploadfiles.php. Denne funksjonen konverterer norske navn til lovlig syntaks for DSpace.

På ekstra metadataknappene er det kodet inn en hjelpeboks som viser ekstra informasjon om feltene. Disse vises når musen holdes over boksene ved hjelp av en "EventListener". Denne informasjonsboksen er bygget opp som en vanlig boks i HTML. En slik boks kalles en div. Div-en har tilhørende egenskaper satt i CSS. Egenskaper som bredde, høyde, posisjon, rammefarge og tykkelse er bestemt i CSS.

Ved hjelp av JavaScript kan egenskaper og CSS endres med DOM-API`et i HTML. Dette gjør det mulig å vise en slik informasjonsboks med ønsket tekst knyttet til hver enkelt knapp. Posisjonen til boksen kan da styres i forhold til knappen den dukker opp ved. Hjelpeteksten kan styres etter navn eller innhold i knappen. Eksempel i figur 12.




**Figur 12 - Ekstra metadata eksempel**

Når hovedsiden er lastes inn kjøres to parallelle AJAX handlinger. Disse bruker begge XMLHttpRequest og JSON array som overføringsformat. Den ene kjører fra Upload-klassen, den andre fra Metadata-klassen.

Fra klassen ansvarlig for filhåndtering startes filesremaining.php denne filens oppgave er å sjekke om det finnes filer igjen i brukermappen på serveren. Hvis det er filer igjen der, overføres filstørrelse og filnavn. verdiene blir tatt i mot av JavaScriptet og vist til brukeren i slipp-boksen.

Den andre AJAX handlingen som kjøres, er henting av informasjon via skolens åpne LDAP server. Her hentes brukerens navn, e-post og telefonnummer ut via PHPs LDAP-modul. Når brukerinformasjonen er mottatt av JavaScriptet legges den inn i formen, og navnet til brukeren vises nederst på siden ved logg ut knappen.

Tilkoblingen til den åpne LDAP serveren er ikke kritisk for EasyUp sin funksjonalitet. Hvis tilkoblingen feiler, fortsetter publiseringsprosessen som normalt uten advarsler. Brukeren blir da nødt til å fylle ut navn og e-post manuelt.

### 5.3.1 Hjelpefiler for publisering

Hjelpefilene som brukes under publiseringen lages under opplastingsdelen. Grunnen til at disse blir omtalt her, er at informasjonen som brukes til å lage disse filene blir innhentet fra hovedsiden.

#### dublin_core.xml

Denne filen kan for eksempel se slik ut:

```
<dublin_core>
    <dcvalue element="contributor" qualifier="author">Ole Woldstad</dcvalue>
    <dcvalue element="identifier" qualifier="other">ole.woldstad@HiG.no</dcvalue>
    <dcvalue element="title" qualifier="none">sd</dcvalue>
    <dcvalue element="description" qualifier="abstract">sd</dcvalue>
    <dcvalue element="language" qualifier="iso">nb_no</dcvalue>
    <dcvalue element="date" qualifier="issued">17/5/2012</dcvalue>
    <dcvalue element="format" qualifier="mimetype">image/jpeg</dcvalue>
</dublin_core>
```

**Figur 13 - Dublin_core.xml oppbygging**

Formatet på XML filen er bestemt av Dublincore standarden[13]. Denne benyttes av DSpace. ”Element” og ”qualifier” må være godkjente verdier for at importeringen til DSpace skal fungere.

Informasjonen som hentes til dublin_core.xml og lisensiering blir hentet ut av nettsiden ved hjelp av FormData. FormData er en del av XMLHttpRequest API`et og er et grensesnitt for lett håndtering og sending av data via JavaScript.

Et FormData-objekt kan opprettes, skjøtes på, eller brukes til å hente inn data direkte fra en utfylt form i HTML. Fra serversiden ser en overføring med formdata ut som det ville gjort om man poster en HTML-form med encoding satt til ”multipart/formdata”.

Koden under vise hvordan metadataene og valgt lisens hentes ut av hovedsiden og sendes til ”importfiles.php” for videre behandling.

```
var xhr2 = new XMLHttpRequest();            //Ny XMLHttpRequest
               var input = new FormData();  //Bruker Formdata for å sende xmlhttp
               var form = document.metad;   //Henter ut form fra nettside
                                            //Og åpner PHP-fil med post og asyncron
                                            //kommunikasjon…
       xhr2.open("POST", 'importfiles.php', true);
                                            //Legger til Lisenstype felt i formen…
       input.append('Licensetype', document.getElementById("licenseselect").value);
       for (var i=0;i<form.length;i++) {    //Går gjennom alle feltene I metadata
               if (form.elements[i].id != "languagesearch")
                                            //Og legger til alle feltene som ikke er
                                            //språksøk…
                       input.append(form.elements[i].id, form.elements[i].value);
               }
       }
```

**Figur 14 - Sending av metadata fra JavaScript til PHP**

Dublin_core.xml blir bygget av PHP og det brukes ”switch case” for å oversette norske navn til riktig kombinasjon av ”qualifier” og ”element”.

Under vises noen utdrag fra koden på hvordan XML-filen bygges av PHP.

```
//Fil opprettes i brukerens mappe og dublincore elementet startes…
$file = fopen($_SESSION['folder']."/1"."/"."dublin_core.xml", "w");
fwrite($file, "<dublin_core>\n");
…

//To eksempler på Cases fra oversetteren switch case kjøres på alle feltnavn
case Navn:            //Hvis casen treffes, kjøres funksjonen xmlField
                      //med to hardkodede parametere (element og
                      //qualifier) pluss formens verdi…
       xmlField("contributor", $value, "author");
       break;

case Epost:
       xmlField("identifier", $value, "other");
       break;
…
fwrite($file, "</dublin_core>");     //Avslutter xmlformatet
fclose($file);                       //Lukker filen
…
//Funksjonen som skriver til dublincore
function xmlField($element, $value, $qualifier){
       global $file;                 //Den globale filvariablen brukes
                                     //Syntaksen i xmlfilen settes i variabler
       $beforeElement = '<dcvalue element="';
```

**Figur 15 - Bygging av Dublin_core.xml**

### License

På hovedsiden kan bruker velge mellom ulike lisenser fra en nedtrekksliste. , og PHP blir brukt for å kopiere tilhørende lisensfil inn i brukermappen.

### Content

Denne filen inneholder en liste over filer som skal være med på publiseringen. Filnavn som ikke befinner seg inne i denne listen blir ikke publisert.
Filen bygges med PHP ut i fra listen med filer som blir overført når opplastingen starter.

### Samling

Valg av samling er laget annerledes. Valgt samling lagres på server med en gang den er valgt av bruker. Samlingen blir laget eller endret i en sesjonsvariabel med et XMLHttpRequest.

Samlingsboksen bruker AJAX for å vise/fjerne tekst om at samlingen ikke er valgt når dette er tilfelle.

Denne boksen er bygget av en PHP-fil som inkluderes i index.php for å spare plass.
En slik modulær metode kan benyttes på moduler om det er ønskelig at en administrator skal kunne legge til/fjerne funksjonalitet. Dette er gjort med PHP fordi listen over samlinger ligger som en sesjonsliste på server.

**Språkvalg**

Vi har valgt å utdype forklaringen rundt språkvalg i metadata. Dette gjør vi fordi det er et meget godt eksempel på funksjonalitet som kun er mulig med asynkron JavaScript og XML.(AJAX).

For å forenkle måten man velger språk på, har vi brukt et søkefelt kombinert med en nedtrekksliste. Listen endres dynamisk etter hva slags søkeord man skriver og oppdateres hver gang en ny bokstav trykkes inn.

Dette er løst ved hjelp av en søkefunksjon i PHP som kjøres med en parameter hver gang en bokstav trykkes inn i søkefeltet.

Kort forklart, sendes innholdet i søkefeltet inn til server. Serveren bruker PHP til å søke gjennom en tekstfil. Resultatene sendes tilbake til klienten, separert med semikolon. Klienten putter så svarene inn i nedtrekkslisten, slik at brukeren kan se og velge språk. Alt dette skjer i løpet av noen hundre millisekunder.

Hele innholdet i nedtrekkslisten bygges med JavaScript. Denne koden er tidkrevende å sette seg inn i, men triviell rent teknisk. Vi skal i stedet forsøke å forklare søkefunksjonen i PHP under.

XMLHttprequest settes opp og innholdet i søk sendes til language.php for hvert tastetrykk:

```
//Verdien som sendes til PHP-filen language.php
xmlhttp.open("POST", "language.php", true); //Asynkron overføring
xmlhttp.setRequestHeader("Content-type","application/x-www-form-urlencoded");
xmlhttp.send("q=" + input.value);           //Variabelen «q» sendes med søkeordet som
                                            //verdi
```

**Figur 16 - Håndtering av språkvalg: 1**

Søkeverdien sendes som POST, og blir kalt ”q”

Det første som skjer i language.php er at filen rfc1766.txt leses inn og puttes i en todimensjonalt liste kalt et array.

```
//Utdrag fra rfc1766.txt:
ar_qa;Arabic (Qatar)
eu;Basque
bg;Bulgarian
```

**Figur 17 - Håndtering av språkvalg: 2**

En oppføring i arrayet(listen) som blir bygget ser slik ut [bg] = ”Bulgarian”.

```
//Alle språkene blir gått igjennom, for hvert språk spørres det om
// søkeverdien i små bokstaver er lik starten på språknavnet eller språkkoden…?
if (strtolower($q)==strtolower(substr($language,0,strlen($q))) ||
    strtolower($q)==strtolower(substr($langcode,0,strlen($q))))
//I såfall skal listen over treff bli en lengre…
$codes= $codes . ";"  . $langcode ;                //Add another line
//Og når alle språkene er sjekket returneres alle de semikolonseparerte språkene
return $codes;
```

**Figur 18 - Håndtering av språkvalg: 3**

### 5.3.2 SWORD

SWORD står for Simple Web-service Offering Repository Deposit og er basert på Atom Publishing Protocol. Sword er et bibliotek for publisering i databasedepoter.
Kjente depoter som støtter SWORD er DSpace, EPrints, Fedora, IntraLibrary og DataBank.

Publiseringsprosessen med SWORD består av to steg. Steg en består av å autentisere brukeren for å skaffe et dokument som inneholder en oversikt over hvor brukeren har lov å publisere. Dette dokumentet kalles SWORD servicedokument. Trinn to består i å publisere samlingen i et databasedepot.

Vi bruker SWORD sitt servicedokument til å finne ut hva slags samlinger(collections) våre brukere har rettigheter til å publisere i. Med andre ord benytter vi oss av DSpace sitt SWORD-API i autentiseringsprosessen vår. Vi benytter oss derimot ikke av et SWORD-bibliotek.

```
<?xml version="1.0" encoding="UTF-8"?>
<app:service xmlns:atom="http://www.w3.org/2005/Atom"
xmlns:app="http://www.w3.org/2007/app" xmlns:sword="http://purl.org/net/sword/"
xmlns:dcterms="http://purl.org/dc/terms/">
   <sword:version>1.3</sword:version>
   <sword:verbose>true</sword:verbose>
   <sword:noOp>true</sword:noOp>
   <sword:maxUploadSize>-1</sword:maxUploadSize>
   <atom:generator uri="http://www.DSpace.org/ns/sword/1.3.1" version="1.3"/>
   <app:workspace>
      <atom:title type="text">DSpace at oo.no</atom:title>
      <app:collection href="http://DSpace.oo.no/sword/deposit/666/8">
         <atom:title type="text">TØL-Open</atom:title>
         <app:accept>application/zip</app:accept>
         <sword:acceptPackaging q="1.0">http://purl.org/net/sword-
       types/METSDSpaceSIP</sword:acceptPackaging>
         <sword:collectionPolicy>NOTE: PLACE YOUR OWN LICENSE
         </sword:collectionPolicy>
         <sword:mediation>true</sword:mediation>
      </app:collection>
   </app:workspace>
</app:service>
```

**Figur 19 - Sword Servicedokument**

Over har vi lagt med et eksempel på et servicedokument for å vise hva det kan inneholde av informasjon. Informasjonen som er relevant for oss er antall samlinger, navn på samlingene og publiserings-URL.

Det er laget et eget, godt dokumentert PHP-bibliotek for SWORD. Dette er noe en fremtidig Versjon av EasyUp kunne benyttet seg av til autentisering og publisering.

Det er minst to store fordeler med å bygge EasyUp på SWORD. Et slikt bibliotek ville økt robustheten til EasyUp og du kunne med enkle grep brukt EasyUp på andre databasedepoter.

Vi hadde SWORD i bakhodet når vi programmerte og kodet oppgaven. Dette sørger for at endringer som skal til før EasyUp blir en SWORD-app, hovedsakelig begrenses til to filer: checklogin.php og uploadfiles.php.
Vårt estimat er at et team på to kunne gjort dette på en, til to arbeidsuker.

## 5.4 Filhåndtering

For håndtering av filer bruker vi File-API, se litteraturliste [4].

API`et inneholder funksjonalitet for å håndtere filer som objekter i JavaScript.

Denne verktøykassen inneholder følgende grensesnitt:

- FileList - Tilgang til lokale filer bruker velger.
- Blob - Tilgang rådata av hele eller deler av en fil.
- File - Informasjon om filer. Som f. eks navn, størrelse og sist endret dato.
- FileReader - Leser lokale filer inn i minne for deretter å bruke denne informasjonen til og f. eks lage miniaturbilder.

EasyUp har to forskjellige måter en bruker kan velge filer på.

### 5.4.1 Dra og slipp

Vi lager en droppsone hvor brukeren kan dra en eller flere filer fra sitt lokale system og slippe dem inn i nettleseren.


**Figur 20 - Dra og slipp dropsone**

```
<div id="dropzone">
      <h1 class="drop">Slipp filene inn her!</h1>
      <p class="drop">Slipp/velg gjerne flere filer av gangen!</p>
```

**Figur 21 - Kode for dropsone**

Droppsonen skifter farge når brukeren har valgt en fil og har musepekeren over seg.


**Figur 22 - Dropsone skifter farge**

Vi gjør dette med følgende JavaScript kode:

```
dropzone.style["backgroundColor"] = "#6dcff6";
```

**Figur 23 - Kode for fargeskift**

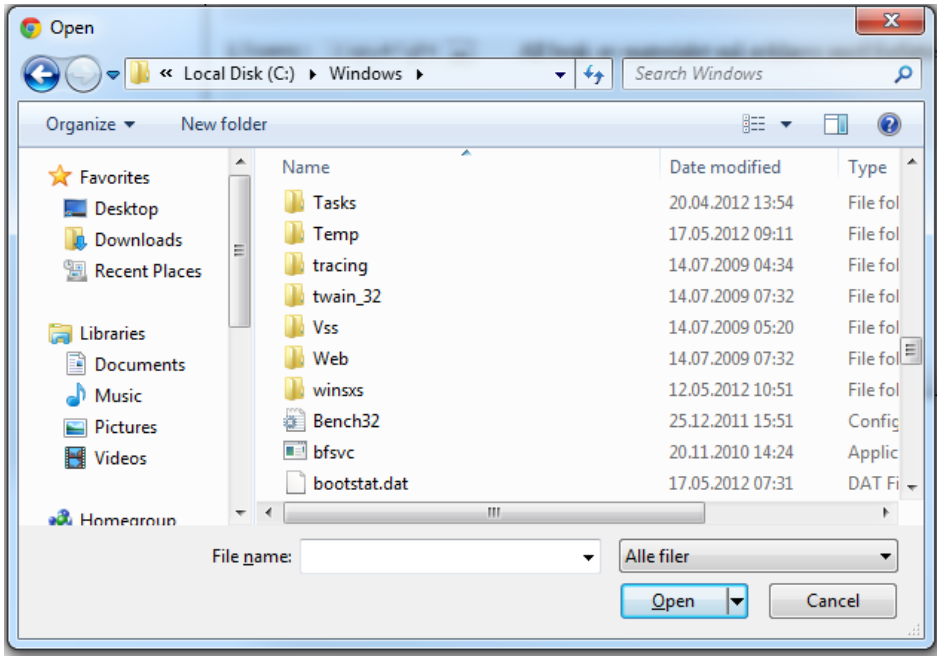
## 5.4.2 Input

Her har vi brukt input elementet med valget ”multiple” for å støtte flervalg av filer.

```
<input type="file" value="Legg til" id="inputfiles"
 name="files[]" multiple/>
```

**Figur 24 - Kode for input element**

Her kan bruker velge en eller flere filer for opplasting.




**Figur 25 - Input elementets funksjonalitet**

```
dropzone.addEventListener('drop', Drop, false);
inputfiles.addEventListener('change', Input, false);
```

**Figur 26 - EventListener for dropp og input av filer**

For å registrere når bruker har sluppet og/eller valgt filer bruker vi ”EventListeners” i JavaScript for å registrere handlinger.

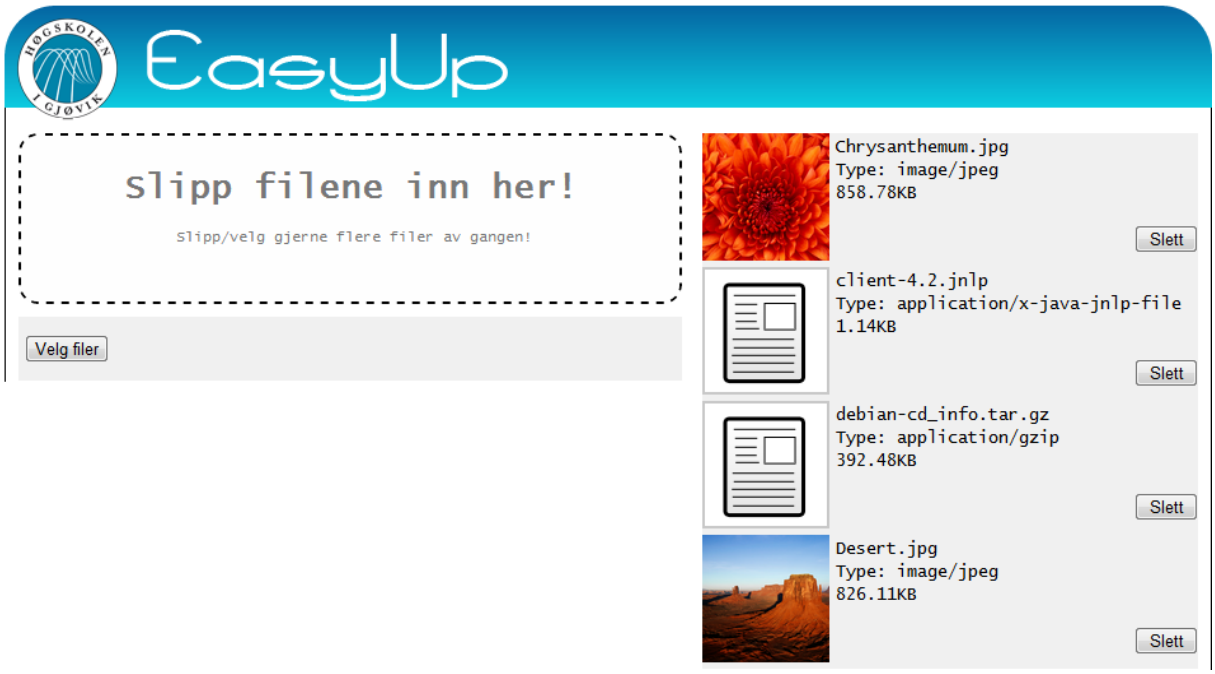
```
var tmpfiles = event.dataTransfer.files;      //Henter filer fra event
for (var i=0;i<tmpfiles.length;i++){          //Går filer fra event
      files.push(tmpfiles[i]);                //Legger filene i files[]
```

**Figur 27 - Legger til nye filer i fillisten**

For hver gang bruker slipper og/eller velger filer kjøres de respektive funksjonene: Dropp eller Input. Begge funksjonene gir samme resultat. De legger inn filene brukeren har valgt inn i et array kalt files[].

### 5.4.3Viser valgte filer

For å vise filene brukeren har valgt, lager vi en liste på høyre side. Hver fil får sitt eget ikon i tillegg til navn, type og størrelse. I tillegg får bildefiler som jpg, png, bmp osv sitt eget miniatyrbilde for og lettere kunne identifiseres.




**Figur 28 - Viser valgte filer**

```
var reader = new FileReader();          //Ny FileReader
reader.file = files[k];                 //Filen som skal leses
reader.id = "filereader"+k;             //Setter id og num...
reader.num = k;
reader.onload = function(event){        //Når filen er lest inn i minne
      var thumbnail = new Image();      //Nytt bilde element
      thumbnail.src = this.result;      //FileReader output = Nytt bilde
      thumbnail.setAttribute('class', "thumbnail");       //Setter class
      document.getElementById("filediv"+this.num).appendChild(thumbnail);
```

**Figur 29 - Kode for miniatyrbilder**

For denne funksjonaliteten bruker vi FileReader grensesnittet nevnt over. Vi bygger elementene vi trenger med JavaScript og fyller bildet med output fra FileReader.

### 5.4.4 Fortsettelse av avbrutt opplasting

For opplastinger som blir avbrutt før de er ferdige har vi implementert funksjonalitet som fortsetter med opplastingen der den slapp. Dette er spesielt hendig ved avbrytelse av større filer.
Bruker sparer dermed både tid og båndbredde på å laste opp kun den deler som mangler.
For å få til dette bruker vi AJAX.

Når bruker entrer siden sender JavaScript en forespørsel ved hjelp av XMLHttpRequest til server om det finnes uferdige filer i brukeren opplastingsmappe.

PHP skanner mappen og bygger ett JSON-array som inneholder antall filer samt navn og størrelse for hver fil. Arrayet sendes tilbake til JavaScript som skriver ut informasjonen til bruker. Brukeren kan da velge disse filene på nytt og dermed fortsette opplastingen. Hvis brukeren ikke velger disse filene, forsetter ikke opplastingen. Filene blir da slettet ved første publisering og må lastes opp på nytt om de skal publiseres.
For å få til dette bruker vi Blob grensesnittet som nevnt over. Vi bruker metoden slice(). Metoden tar 2 parametre, startbyte og stoppbyte. Altså velger vi fra hvilke byte og til hvilke byte av filen vi vil bruke.

Dessverre støttes slice() metoden kun av Chrome og Firefox til dags dato og dermed også fortsettelse av avbrutt opplasting. Andre nettlesere som Safari støtter dermed ikke vår funksjonalitet på dette punktet.

```
//Setter opp ett XMLHttpRequest objekt
var filecheck = new XMLHttpRequest();
filecheck.open("POST", 'filesremaining.php', true);

//Når JavaScript mottar svar fra PHP om avbrutte filer kjøres denne
//funksjonen. Den viser brukeren hvilke filer som kan fortsettes
filecheck.addEventListener("load", function(event){
	responseArray = eval ("(" + filecheck.responseText + ")");
	if(responseArray.files.length != 0){
	dropzone.innerHTML += "<p class='warning'>" +
		"Følgende filer er ikke publisert. <br>" +
		"Vil du fortsette, slipp/velg disse på nytt." +
		"(Filer som ikke er valgt, blir slettet ved neste " +
				"publisering.) </p><ul>";

	//Går gjennom svaret fra serveren og skriver ut navn og
	//størrelse på avbrutte filer slik at bruker kan fortsette
	for (var i=0;i<responseArray.files.length;i++){
		dropzone.innerHTML += "<p> - " +
		shortenString(responseArray.files[i].name, 33) +
					" - Lastet opp: " +
```

**Figur 30 - Viser ikke publiserte filer for bruker**

JavaScript kode for fortsettelsesfunksjonalitet.
Her for kommunikasjon mellom klient og server.

## 5.5 Opplasting

For opplasting av filer bruker vi også XMLHttpRequest. Kort forklart går vi gjennom arrayet med filer kalt files[] og sender hver enkelt fil til server. Her tar PHP seg av lagringen.
Som nevnt har vi fortsettelsesfunksjonalitet av avbrutte opplastinger. Fordi vi ville støtte nettlesere som Safari uten støtte for metoden slice() må vi her sjekke om funksjonalitet finnes og deretter kjøre kode tilpasset nettleser. I tillegg har Chrome og Firefox implementert slice() metoden forskjellig slik at også her må vi sjekke hvilke nettleser som kjøres.
Hvis slice() støttes, kjøres Chrome eller Firefox sin implementering:

```
if(file.mozSlice || file.webkitSlice    //Hvis slice() støttes

      /Går gjennom alle avbrutte filer...
      for(var e in filesize){
            if( e == file.name){

            //Hvis filen fines settes foreløpig status for progressbar
            status = ((filesize[file.name]/file.size)*(100));
            fileexists = filesize[file.name];
            }
      }
```

**Figur 31 - Kode for å dele opp filen som skal opplastes**

Deretter oppretter vi et XMLHttpRequest objekt for å sende enten filen enten som blob eller hele filen hvis ikke den var avbrutt. Her er blob eksemplet.

```
var xhr = new XMLHttpRequest();         //Nytt objekt

//Filen sendes til upload.php. I tillegg setter vi headere, blant
//annet bruker vi header til å overføre filnavn
xhr.open("POST", 'upload.php', true);
xhr.setRequestHeader("Cache-Control", "no-cache");
xhr.setRequestHeader("X-Requested-With", "XMLHttpRequest");
xhr.setRequestHeader("X-File-Name", file.name);
xhr.setRequestHeader("Content-Type", "multipart/form-data");
```

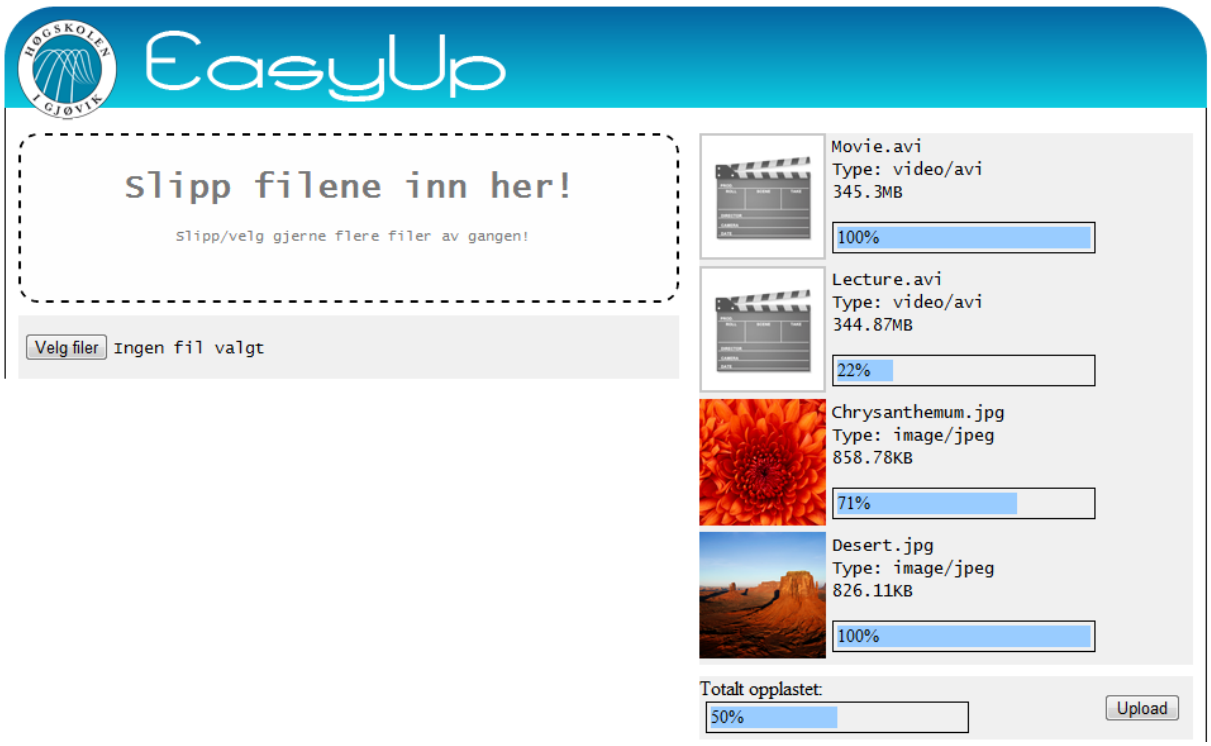
**Figur 32 - Kode for å sende fil til server**

På serversiden tar upload.php imot filen og skriver den til disk. For å kunne støtte fortsettelsesfunksjonalitet har vi valgt å bruke php://input stream for å lagre filer. Her vil vi kunne ta imot filen som en bitstream. Dette er ikke mulig i den mer vanlige måten å lagre filer på i PHP nemlig $_FILES arrayet som inneholder filer sendt via http post metoden.

```
//Vi oppretter en file handle. Dette er bitstreamen
$in = fopen('php://input', 'a');

// Deretter skriver vi bitstreamen til fil. Navnet henter vi ut fra
// http headeren ved navn ”HTTP_X_FILE_NAME” vi satte i JavaScript
// Vi henter informasjon om hvor filen skal skrives fra session
// variabelen $_SESSION[’folder’]
while(!feof($in)){
```

**Figur 33 - Kode for å lagre fil på server**

For å indikere til bruker hvor mye som er igjen av opplastingen har vi valgt å legge til en fremgangslinje for hver fil. I tillegg har vi en linje for fremgangen for hele opplastingen.




**Figur 34 - Fremgangslinje funksjonalitet**

```
var xhr = new XMLHttpRequest();

//Vi legger til en EvenListener på "progress" handlingen på upload
//funksjonen til XMLHttpRequest obkjektet.
upload = xhr.upload;
upload.addEventListener("progress", function(event) {

      //Vi henter en referanse til fremgangslinjen og en variabel
     /til nåværende opplastingsstatus. (Dette skjer flere ganger i
       sekundet)
      var progress = document.getElementById(this.id);
      var percentLoaded = Math.round(((event.loaded / event.total) *
      100)+ this.num);
      //Her settes fremgangslinjen til opplastingsstatus.
      //Vi setter linja til 100% manuelt for å sikre riktig visning
      if (percentLoaded < 100) {
            progress.style.width = percentLoaded + '%';
```

**Figur 35 - Kode for fremgangslinje**

For dette har vi bruk funksjonalitet på XMLHttpRequest objektet. Vi har satt på en EventListener på ”progress” handlingen. Denne slår inn mange ganger i sekundet under opplasting og rapporterer fremgangen i opplastingen. Dette bruker vi dermed til å oppdatere fremgangslinjen. Her er koden for hvordan vi har implementert dette.

## 5.6 Publisering

Når alle filene er opplastet og hjelpefiler er laget er alt klart for å publiseres i DSpace. Filstrukturen må være av typen ”DSpace Simple Archive Format” som forklart i punkt 4.1. DSpace bruker som forklart tidligere et bash skript for å ta seg av importeringen.
Skriptet kan startes med følgende kommando:

```
DSpace-installasjonsmappe/bin/DSpace import
```

Følgende argumenter kan brukes:

| | |
|---|---|
| -a | Ny publisering |
| -r | Erstatt publisering |
| -d | Slett publisering |
| -s | Mappen filene som skal importers ligger |
| -c | Hvilken collection skal filene publiseres i |
| -m | Hvor mapfile skal lages/hentes |
| -e | E-post til E-person som gjør publiseringen |
| -w | Skal importen kreve en godkjenning fra administrator |
| -n | Skal publiseringen gi bruker en E-post hvis vellykket |
| -t | Brukes for å teste importering. Ingenting importeres hvis dette argumentet brukes |
| -R | Brukes for å fortsette en import som feilet. |
| -h | Hjelp |

Vi har laget et bash script som tar seg av kjøringen av dette DSpace import skriptet.

```
#Import.bash
#!/bin/bash
#Setter hjelpevariabler
BIN="/usr/share/DSpace/bin
WEB="/home/DSpace/easyup.oo.no/ola

#Kjører DSpace sitt import script med variable fra PHP. Sover i 2
#sekunder etter import for å være 100% sikker på at importen er
#ferdig. I teorien kjøres aldri sleep før importen er ferdig uansett.
$BIN/DSpace import -a -e $1 -c $2 -s $WEB/$3 -m $WEB/$3/1/$4 ; sleep 2

#Sletter filene etter publisering
rm -rf $3/1/*
```

**Figur 36 – Bash skript for importering av filer til DSpace**

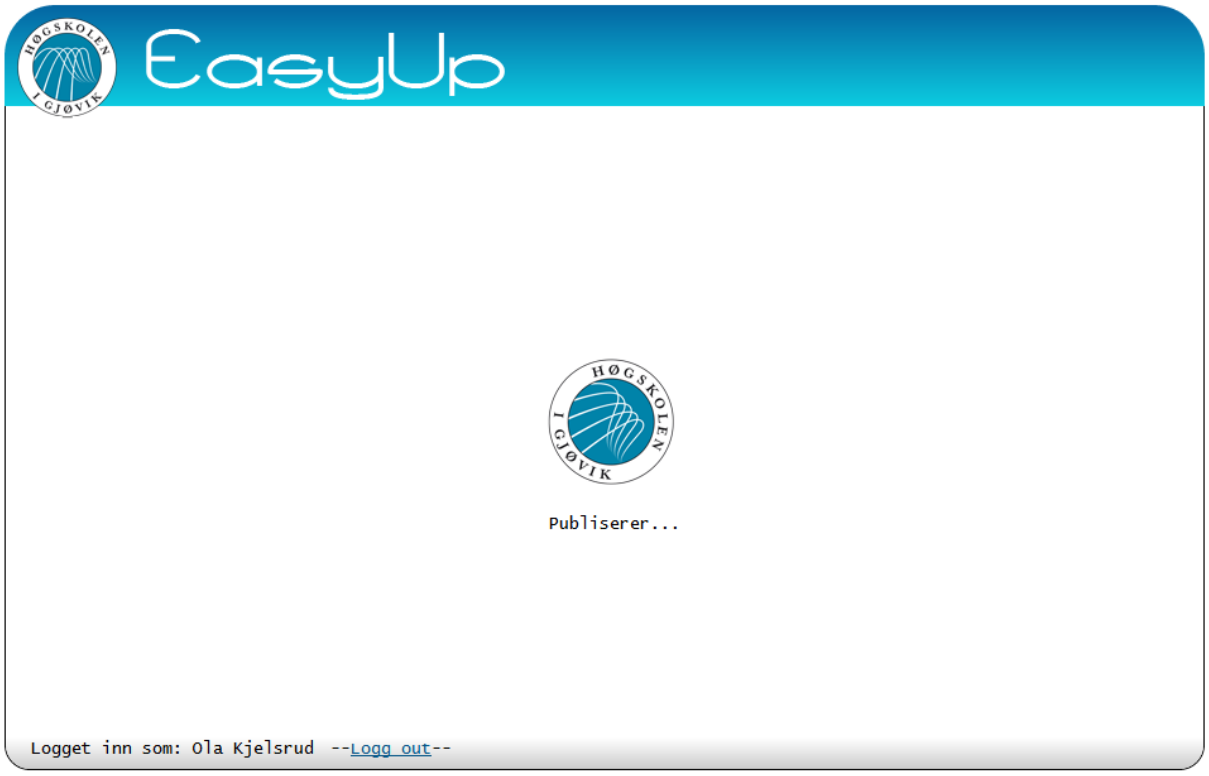
For å kjøre dette skriptet benytter vi PHP funksjonen:

```
$brukernavn = $_SESSION['u']."@HiG.no";
system("./import.bash $brukernavn $_SESSION[collection] $_SESSION[folder]
mapfile");
```

**Figur 37 - Kode for å kjøre bash skript fra PHP**

Alle argumenter hentes fra session i PHP og renskes med tanke på sikkerhet.

Mens importeringen er underveis viser JavaScript en liten HiG animasjon:




**Figur 38 - Skjermbilde under publisering**

Når JavaScript får beskjed om at importeringen er ferdig vil en side med diverse informasjon om publiseringen og annen informasjon vises. Denne siden kan enkelt redigeres. Foreløpig har vi valgt å linke til publiseringen som ble gjort samt tilbake til start for å publisere på nytt.


**Figur 39 - Skjermbilde for fullført publisering**

## 5.7 Sikkerhet

I alle IT-systemer blir sikkerhet mer og mer viktig. Systemer skal være sikret mot alle typer angrep. Vårt system skal driftes av IT-Tjenesten og det er viktig av deres krav blir møtt.

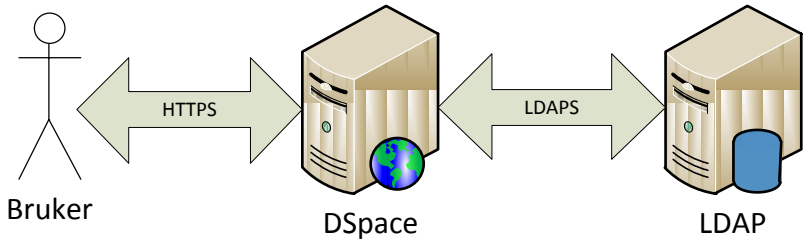
Vårt system består av flere typer kommunikasjon og brukerinput som potensielt kan utgjøre sikkerhetshull. Vi har tenkt sikkerhet under implementeringen, men med begrenset erfaring på dette området kan vi ikke garantere noe før en sikkerhetsgjennomgang av kyndig personell.

Informasjon om hva vi har tenkt og hvordan vi har løst noen av de største punktene innefor sikkerhet, følger.

### 5.7.1 Kommunikasjon

Vårt system består av flere kommunikasjonskanaler som i teorien kan blir angrepet. I hovedsak har vi her 3 aktører. Bruker, DSpace og LDAP server. Serveren som kjører DSpace kjører også Apache for EasyUp.

Vi får dermed følgende kommunikasjonsflyt:




**Figur 40 - Kommunikasjonsflyt for data**

Vi har valgt å bruke HTTPS og LDAPS for kommunikasjon mellom aktørene. Dette gjør at all kommunikasjon fra og til systemet er kryptert. På denne måten vil et potensielt avlyttingsangrep ikke medføre at informasjon som brukernavn og passord kommer på avveie. Protokollene gir også beskyttelse mot endring av data under transport.

**HTTPS**

"Hypertext Transfer Protocol Secure" er en kombinasjon av HTTP og SSL/TSL protokollen. HTTPS gir brukeren en sikker kanal som tilbyr autentifisering og kryptering av data for beskyttelse mot avlytting og endring av data under transport.

For å bruke HTTPS må serveren ha ett eget sertifikat som må signeres av et sertifikat autoritet som f. eks Microsoft, VeriSign eller lignende. Brukeren må deretter koble til via https://nettside.domene i stedet for http://nettside.domene. HTTPS bruker port 443 i stedet for port 80 som HTTP.

**LDAPS**

For autentifisering av brukere har vi konfigurert DSpace til å autentifisere mot HiG's LDAP server. LDAP protokollen sender til vanlig all kommunikasjon i klartekst. For å hindre avlytting og endring av data under transport har vi valgt å bruke LDAPS.

LDAPS er LDAP over SSL. Her vil altså all kommunikasjon krypteres og sikres mot endringer under transport. På samme måte som HTTPS krever dette ett sertifikat.
Denne kommunikasjonen er i vårt system skjult for bruker. Brukeren autentifiserer mot EasyUp som igjen autentifiserer seg mot DSpace som igjen autentifiserer seg mot HiGs LDAP server. LDAPS bruker port 686 i stedet for 389 som LDAP.

I tillegg henter EasyUp brukerinformasjon fra HiGs åpne eksterne LDAP server. Denne kommunikasjonen har vi valgt å ikke kryptere da all informasjon allerede er åpen for alle. Vi henter her ut navn, e-post og telefonnummer for at brukeren skal slippe å fylle ut disse punktene ved publisering.

### 5.7.2 System

Systemet tar imot både filer og input fra brukere. Det er derfor viktig å sikre seg mot eventuelle angrep som benytter seg av dette.

**Filer**

Vi har ikke valgt å implementere noe form for viruskontroll eller lignende for å sjekke opplastede filer. Dette vil være noe som bør evalueres av oppdragsgiver før bruk av systemet.

Det er heller ikke satt noen restriksjoner på hvilke filtyper som kan lastes opp. Dette er noe som enkelt kan implementeres skulle oppdragsgiver ønske slik funksjonalitet.

**Input**

Systemet tar imot brukerinput flere steder. Her er kode injeksjon et mulig angrep. Vi har derfor implementert funksjoner som stripper brukerinput for potensielle tegn som brukes til dette. Her er funksjonen vi bruker I PHP for å hindre tegn som: ”<”, ”;”, ”&” osv.

```
function secure($string) {    //Stripper string for farlige tegn
      $string = strip_tags($string);
      $string = htmlspecialchars($string);
      $string = trim($string);
      $string = stripslashes($string);
      return $string;
}
```

**Figur 41 - Kode for å sikre brukerinput**

### Session

Vi bruker i systemet session variabler for å overføre data mellom PHP filer. Dette er også en potensiell sikkerhetsrisiko. For eksempel finnes det angrep som session kapring der angriper først bruker avlytting for å kapre informasjon som deretter brukes for adgang til server.

### Bash

Etter at filene er lastet opp til server kaller vi bash skriptet import.bash som tar seg av importeringen til DSpace. Problemet er at importeringen krever høyere rettigheter enn brukeren som kaller skriptet har. Vi har løst dette ved å gi apachebrukeren tilgang til å kjøre DSpace sitt import skript.

Den potensielle faren her blir dermed at apachebrukeren har tilgang til ikke bare å importere filer, men også erstatte og slette publiseringer. Dette krever selvsagt at en angriper klarer å modifisere funksjonaliteten på bash skriptet.

## 5.8 Testing

Under utvikling og testing av systemet har vi brukt følgende utstyr:

**Hardware:**

- Dell PowerEdge SC 1425. 1U server
- Intel Xeon 3.2 GHz cpu
- 8GB mine
- 2x 160GB HDD. Raid 0

**Software:**

- Debian 6
- Apache 2
- DSpace

I tillegg har vi brukt programvare som nevnt i punkt 4.1 i rapporten for å installere og kjøre DSpace.

Vi har ikke hatt tid eller mulighet til å teste systemet i et produksjonsmiljø. Den testingen vi har gjort har derfor vært på vår lokale server innad i gruppa og andre interesserte studenter. I tillegg har vi gitt veileder og oppdragsgiver mulighet til å se funksjonalitet for tilbakemelding.

Vi vil derfor på det sterkeste anbefale en beta test av systemet før det tas i bruk. Her bør ytelse og skalering av systemet overvåkes.

Vi har derimot gjort noen observasjoner under vår testperiode.

- Systemet takler flere samtidlige brukere og håndterer både store og små filer på riktig måte.
- Under publisering krever java mye CPU tid. Dette er naturlig siden DSpace er skrevet i java. Vi har ikke klart å fremtvinge feil ved mange samtidlige publiseringer.
- Alle filtyper kan publiseres uten grense på filstørrelse.

# 6.0 Avslutning

## 6.1 Plan vs virkelighet

Dette underkapittelet inneholder informasjon og diskusjon rundt hva som ble gjort annerledes enn det var planlagt. Først tar vi for oss hva som ble annerledes i selve produktet, og hvorfor det ble gjort slik det ble gjort. Så tar vi for oss hva vi gjorde annerledes under selve utviklingen. Her er tidsaspektet i hovedfokus, men også hva vi gjorde og ikke gjorde i forhold til rutiner. Vi avslutter med å diskutere hva vi kunne gjort annerledes og generell kritikk av oppgaven.

### 6.1.1 Avvik i produktet

**Logging**

Vi starter med det kanskje største og groveste avviket mot planleggingen på sluttproduktet, logging. Her var planen at kritiske og ikke-kritiske feil skulle logges på en fil. Dette er ikke på plass.

Det vil ikke være mye jobb for å implementere kode for å sende feilmeldinger fil. Et slikt loggsystem ville kunne gitt oss informasjon om ustabile tilkoblinger eller eventuelle kodefeil. Ideelt sett skulle vi hatt et loggsystem som logget til en database, med feilkoder og ulike grader av feil. For hver feil skulle vi også tatt vare på fil, filtype, og brukernavn knyttet til den aktuelle feilsituasjonen.

Loggvisning skulle vært tilgjengelig gjennom et administratorgrensesnitt med mulighet for å sortere oppføringer basert på tidspunkt, alvorlighetsgrad, bruker, feilkode og hyppighet på feil. Et slikt system var aldri en del av oppgaven, men vil være noe oppdragsgiver kan vurdere i fremtiden.

Grunnen til at et loggsystem, selv i en enklere form ikke er på plass er nedprioritering fra planleggingsfasen og helt til slutten. Dette var en av de få tingene vi ikke fikk tid til.

**Metadata**

Neste punkt som mangler i sluttproduktet er en funksjon for tilbakeføring av endret metadata til fil på klientmaskin. Dette ble lagt fram som en funksjonalitet på det første møte med Kjell Are. Det er to gode grunner til at dette ikke ble implementert.

Filer lastes ofte opp i flertall, og metadata på DSpace blir lagt inn per publisering og ikke per fil. Det blir da problematisk å bestemme hvilke av metadataene som er relevante for hvilke filer.

Dette ble foreslått på et punkt hvor EasyUp var tenkt å være en klientapplikasjon og ikke et webgrensesnitt. Dette er derfor en funksjonalitet som sjelden vil være nyttig siden all informasjon allerede finnes i DSpace og er søkbar der.

Filer inneholder mye informasjon. Planen var å kunne hente ut denne informasjonen for å gjøre publiseringene søkbare på disse kriteriene. Dette har vi gjort, men bare delvis.
Vi henter ikke ut mer informasjon enn filstørrelse filtype og selvfølgelig filnavn. Resten har vi latt være fordi filer ofte publiseres som samlinger av flere filer.

Fordi samlinger ofte inneholder filer laget av andre brukere kan inneholde irrelevant informasjon som etter hvert kan resultere i håpløst mange treff på vanlige søkeord i DSpace.

**Sword**

SWORD til publisering av samlinger var en av beslutningene vi tok underveis. Dette gikk vi bort fra i første omgang fordi SWORD-biblioteket var så tungt og gjorde så mye mer enn det vi trengte av funksjonalitet. Vi valgte å fokusere på det vi hadde av krav i stedet. På den måten skulle vi sikre ønsket funksjonalitet først og eventuelt bygge om prosjektet på SWORD om vi fikk tid.

Det endelige sluttproduktet inneholder mye funksjonalitet og detaljer som ikke var planlagt til å begynne med. Noen av disse tingene er

- Miniatyrbilder og ikoner for filformater
- Framdriftsbar for opplasting
- Syv ulike valg for lisensiering
- Dynamisk språksøking
- Dynamisk tillegging av metadatafelter med informasjonsbokser

Alle elementene på denne listen ble implementert fordi de gjorde publisering med EasyUp lettere eller bedre.

### 6.1.2 Avvik under uvikling.

Vi skrev under forprosjektet om utviklingsmodell og hva slags verktøy vi planla å benytte oss av. Det meste av dette passet veldig godt inn i måten vi jobbet på men det ble allikevel noen avvik. Gjenbruksorientert utvikling og ukentlige møter med veileder er i hovedsak verktøyene vi lot være å benytte oss av. Jevnt over bunner det i mangel på behov.
Vi jobbet mye med ny funksjonalitet, så det var ikke så mye kode å gjenbruke. Vi har lært oss nye metoder ved hjelp av eksempler og er selvfølgelig inspirert av disse. Direkte avskrift har ikke forekommet.

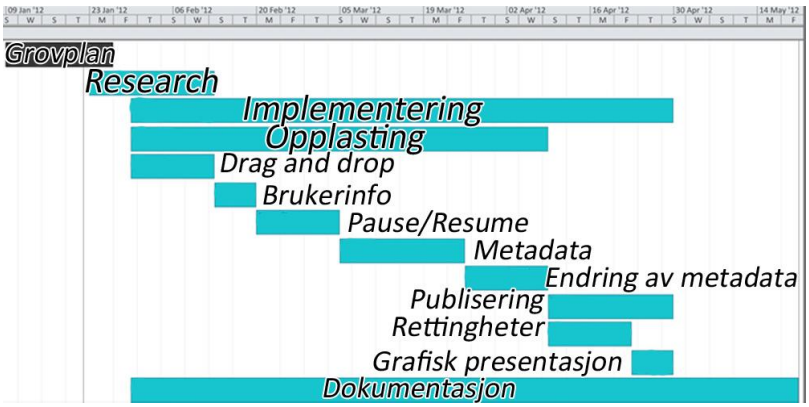
Vi startet ut med ukentlige veiledermøter, men vi så fort at det var litt i overkant. Etter det har vi avtalt møter kun etter behov. Se vedlegg E for møtereferater.

**Tidsavvik**

Når det kommer til tidsplanleggingen var det en god del som ble gjort annerledes. Mye på grunn av at det krevdes mer informasjonsinnhenting enn planlagt.
En annen ting som endret rekkefølgen på prioriteringene var at vi fikk bedre oversikt over hvordan vi skulle gjøre ting. Det gjorde at oppgaver vi tidligere så på som vanskelige, ble overkommelige og derfor flyttet fremover i tid. Fokuset på å implementere de usikre modulene først beholdt vi.

Under har vi lagt ved en kopi av den opprinelige tidsplanen. Vi skal nå kort gå gjennom hovedpunktene fra utviklingen av EasyUp. Punkter fra gantt-diagrammet er merket med "bold".

- **Researchfasen** startet som planlagt og fortsatte helt til starten på april. Implementeringen ble påbegynt etter planen med **drag and drop**, men fikk en lang pause til ca 15. mars på grunn av behov for mere informasjonsinnhenting.
- I midten av mars hadde vi funnet SWORD og begynte umiddelbart å lage innloggingsfunksjonalitet basert på SWORD-API til DSpace. I den forbindelse hadde vi også berørt **brukerinformasjon**.
- Vi hadde lest oss opp på resume og file slice, så vi trodde vi så en mulig løsning på dette. Vi la **resume** til side fordi dette ikke var kritisk for publiseringsprosessen.
- Nå var opplasting til server på plass (ikke publisering i DSpace) I den sammenheng skrev vi om **Drag and drop** på en bedre måte.
- Arbeidet med å lage framdriftslinje var nå påbegynt. Vi kunne også hente ut brukerinformasjon fra LDAP og med det var **metadata** modulen startet på.
- Parallelt med de fire siste punktene satt vi opp **DSpace** i vårt eget testmiljø. Dette brukte mer resurser enn planlagt.
- 30. Mars har vi en fungerende løsning med publisering til DSpace og små bildeikon for valgte bilder og **metadatafelter** er lagt til.
- Mye opprydding i kode og jobbing med å få løsningen til å fungere på våre valgte nettlesere, grafisk som funksjonsmessig.
- **Grafikk/ layout** er nå på plass. vi er nå i andre uka av april




Punktet med **endring av metadata** er fjernet som diskutert under "avvik i sluttprodukt". **Rettigheter** er noe som settes av DSpace administrator pr. samling.

**Figur 42 - Grov tidsplan**

- På grunn av måten vi implementerte metadata lagde vi knapper for å legge til **ekstra metadata** og tilhørende informasjonsbokser, metadata-modulen er nå fullført.
- Midt i april hadde vi en fungerende løsning. Vi satt nå på mer kunnskap og rekodet og restrukturerte derfor mye av funksjonaliteten.
- 17. og 18. april Lagde vi **resume** funksjonalitet.
- Laget bekreftelsesdialog til bruker på opplasting og kodet om framdriftslinje så den støttet resume og slik at den kan styles.
- De siste 10 dagene i april jobbet vi med nettleserstøtte, total fremdriftslinje, generell finpuss og vi avtalte dato for portering til skolens server.
- 2. mai startet vi med dynamisk språkvalg.
- 4. mai implementerte vi **lisensstøtte.**
- 7. mai lisensvalg ferdig og **rapport** påbegynt. Ingen funksjonalitetsendringer er gjort etter denne datoen.

Noe funksjonalitet ble med andre ord fjernet og noe ble lagt til. Mye kode ble endret underveis og en del ble rekodet.

Funksjonalitet som **resume** er et godt eksempel på funksjonalitet som tok mye kortere tid å implementere enn først planlagt. Dette skyldes hovedsakelig mye god informasjonsinnhenting. Vi hadde denne modulen i bakhodet når vi tok beslutninger hele veien, slik at støtten ble enkel å implementere.

Slik var det også med flere andre ting under implementeringen. Til tider satt vi i flere dager kun med teori før ting plutselig løsnet og kode vi trodde skulle ta en uke var på plass på en dag.

Et eksempel på ting som tok lengre tid enn planlagt var vårt arbeid med DSpace. Installasjon og konfigurering var mer avansert en antatt. Vi brukte flere uker på å sette oss inn i hvordan DSpace fungerte og hva vi skulle bruke av funksjonalitet.

## 6.2 Selvevaluering

### 6.2.1 Innledning

Når vi nå er ferdig med implementeringen er vi fornøyde med resultatet. Vi har innfridd oppdragsgivers krav til både ytelse og funksjonalitet. Brukere kan velge filer og publisere disse i DSpace. I tillegg til kravene fra oppdragsgiver har vi utviklet mye ny og spennende funksjonalitet.

Dette har vi kunne gjort på grunn av relativt frie tøyler fra oppdragsgiver. Vi har benyttet de nye mulighetene HTML5 tilbyr for å lage systemet både raskt og dynamisk for bruker. Disse nye standardene minsker gapet mellom standard klient og webapplikasjoner når det kommer til brukervennlighet og ytelse.

Vi er stolte av å kunne tilby blant annet fortsettelse av avbrutt opplasting, noe veldig få aktører på markedet hittil har kunnet tilby. Miniatyrbilder er også en funksjonalitet som gjør EasyUp til en attraktiv løsning. Vi håper også at andre kan dra nytte av kode og funksjonalitet fra vårt produkt til andre formål i fremtiden.

Fra bachelorprosjektet drar vi begge med oss nytting erfaring på flere punkter. Vi har fått et godt innblikk i utvikling av større systemer. Vi har også tilegnet oss erfaring på jobbing med ikke satte standarder der funksjonalitet stadig endres.

### 6.2.2 Forbedringespotensiale og videre arbeid

Selv om vi leverer et fungerende produkt som yter gitte krav fra oppdragsgiver er det visse punkter som har forbedringspotensiale.

Innenfor sikkerhet finnes flere punkter som må testes/sjekkes før systemet settes ut i et produksjonsmiljø. Vi har utviklet etter beste evne, men med begrenset erfaring på dette emne vil det være naturlig å sette fokus her før bruk.

Måten vi gjør DSpace importeringen gir oss ingen feilmeldinger. Det er dermed ingen måte å gjøre bruker oppmerksom på hva som feiler hvis noe skulle feile under importering. Dette vil kunne løses hvis vi hadde gått over fra bash skript til SWORD.

Design er også et punkt som sikkert kan forbedres. Dette er vanskelig da det er forskjellig fra person til person. Skulle oppdragsgiver ønske å endre designet er det i alle fall lagt opp til dette via CSS.

### 6.2.3 Gruppeorganisering og arbeidsfordeling

Med en gruppe på 2 personer der en ble valgt som gruppeleder med dobbeltstemme har vi ikke hatt problemer med å organisere oss. Vi har til tider vært uenige om punkter i oppgaven, men vi har alltid klart å komme til enighet etter en kort diskusjon.

Vi har dratt fordeler ved at vi bor og jobber sammen. Opptil flere ganger hver dag har vi tatt såkalte stående møter der vi diskuterer funksjonalitet og jobbing videre.

Vi har klart å fordele arbeide likt i mellom oss, og ingen av oss føler de har gjort mer enn den andre.  I tillegg er vi fortsatt venner, da de få konfliktene som har kommet opp har blitt tatt hånd om raskt.

### 6.2.4 Personlig tilbakeblikk

**Ola:**
Etter et vanskelig valg i november tok vi fatt på det som skulle bli ett lærerikt og minneverdig prosjekt. Jeg ble valgt som prosjektleder da jeg liker å ha oversikt over hva som foregår. Vi kom raskt frem til at et webgrensesnitt var å foretrekke og jeg hadde allerede her en god oversikt over hva som skulle inn av funksjonalitet.

Det å få lastet opp filer via et webgrensesnitt viste jeg ville gå greit. Jeg var derimot bekymret over hvordan DSpace fungerte og hvordan vi kunne på importere inn filer. Etter mye om og men for å installere og konfigurere DSpace kom vi over 2 forskjellige metoder.

Da jeg har erfaring med bash skript på Linux plattform fra før kjørte vi i gang med dette valget. Nå i etterkant angrer jeg litt på denne beslutningen da selv om systemet fungerer, ville nok SWORD med error og tilbakemeldinger vært ett bedre alternativ.

Etter 3 år som student har det vært moro å bruke mye av kunnskapen vi har lært oss på et større produkt. I starten av prosjektet hadde jeg bare så vidt vært borti HTML, PHP, CSS og JavaScript. Jeg har lært veldig mye innenfor webteknologi og kommer helt sikkert til å bruke dette for å utvikle flere web relaterte løsninger.

Selve systemet har mye funksjonalitet jeg er stolt over selv om jeg kanskje ikke er 100 % fornøyd med tanke på sikkerhet og bruk av DSpace bash import. Alt i alt har prosjektet gitt meg mye erfaring som jeg tar med meg videre til neste prosjekt, enten det blir i jobb eller studentsammenheng.

**Ole:**
Når vi startet på oppgaven virket den for meg både stor og uoversiktlig, men allikevel overkommelig. Tidligere har jeg laget noen nettsider som bygger på PHP, HTML og var stylet med CSS. JavaScript er noe jeg aldri har vært borti. Jeg hadde derfor ikke en anelse om hvordan vi skulle løse halvparten av utfordringene og vi visste ikke helt hvor vi skulle begynne.

Takket være min naive optimisme kastet jeg meg ut i oppgavene. Sammen jobbet jeg og Ola febrilsk med innhenting av informasjon for å gjøre det usikre sikkert. Jeg leste meg opp på JavaScript og PHP og tok to webtester på dette i forbindelse med en jobbsøkeprosess jeg var i.

Vi begynte etter hvert, så smått å få oversikt over hvordan oppgaven kunne løses. En plan etter vår felles beste evne, ble utformet. Arbeidsoppgaver ble fordelt og jobbingen startet.

Følelsen av å starte med et problem og avslutte med en ny funksjon, er noe jeg trives med. Dette var et mønster som gjentok seg gang på gang under utviklingen av EasyUp og det bygger selvtillit. I etterkant er det lett å glemme alle timene med frustrasjon som ligger mellom problem og løsning.
Etter at oppgaven er løst sitter jeg igjen med en "klart jeg kan"- følelse.
En følelse jeg håper jeg klarer å ta med meg inn i jobbverdenen. Webutvikling har jeg alltid hatt sansen for og med ny tilegnet kunnskap kommer det til å bli enda mer utvikling på hobbybasis.

# 7.0 Konklusjon

Målet for denne bacheloroppgaven har vært å forenkle opplasting av filer til DSpace, et digitalt filarkiv. Vi har utviklet et enkelt og responsivt opplastingsgrensesnitt ved hjelp av den nye HTML5 standarden.

Funksjonalitet som skiller seg ut fra lignende løsninger:

- Drag & Drop. For valg av filer.
- Pause/Resume. For avbrutte opplastinger.
- Enkel og dynamisk brukerinteraksjon.

Den nye HTML5 standarden gjør gapet mellom klient og webapplikasjoner mindre. Funksjonalitet som filhåndtering og øyeblikkelig tilbakemeldinger var før forbeholdt rene klientapplikasjoner.

Løsningen fyller kravene fra oppdragsgiver, men krever litt etterarbeid i form av testing og sikring da vi ikke har hatt mulighet til å kjøre en beta test i et produksjonsmiljø.

I løpet av utdanningen vår har vi bygget opp en verktøykasse bestående av blant annet systemutvikling, programmering, filhåndtering i operativsystemer og nettverkskommunikasjon. Det har vært interessant å se hvordan alle disse ferdighetene kan benyttes sammen for å lage et nytt og helhetlig produkt.

# 8.0 Litteraturliste


1. W3Schools online. PHP-tutorials [Internetside]. Sandnes; Refsnes Data AS;[oppdatert 2012]. Tilgjengelig fra: http://www.w3schools.com/php/default.asp

2. W3Schools online. CSS-tutorials [Internetside]. Sandnes; Refsnes Data AS;[oppdatert 2012]. Tilgjengelig fra: http://www.w3schools.com/css/default.asp

3. W3Schools online. JavaScript-tutorials [Internetside]. Sandnes; Refsnes Data AS;[oppdatert 2012]. Tilgjengelig fra: http://www.w3schools.com/js/default.asp

4. World Wide Web Consortium (W3C). File-API W3C Working Draft 20[Internetside]. W3C;[oppdatert Oct 2011]. Tilgjengelig fra: http://www.w3.org/TR/FileAPI/

5. World Wide Web Consortium (W3C). XMLHttpRequest Level 2 - Working Draft 17[Internetside]. W3C;[oppdatert Jan 2012]. Tilgjengelig fra: http://www.w3.org/TR/XMLHttpRequest/

6. Duraspace. DSpace V1.8 - Documentation. [Internetside].Duraspace; [oppdatert 24 Feb 2012]. Tilgjengelig fra: https://wiki.duraspace.org/display/DSDOC18/DSpace+1.8+Documentation

7. Stuart Lewis. Sword v2 Implementations. PHP-bibliotek [Internettside]. Joint Information Systems Committee; [oppdatert nov 2011] Tilgjengelig fra: https://github.com/swordapp/swordappv2-php-library/

8. The Internet Engineering Task Force, Koder for identifisering av språk, Internet RFC 1766[Internettside]. Tilgjengelig fra: http://www.ietf.org/rfc/rfc1766.txt

9. Sommerville I, Agile software development. I:Horton, M, redaktør. Software Engineering. 9. Boston, Massachusetts:Pearson Education Inc;2011.s.64-77

10. World Wide Web Consortium (W3C). HTML & CSS [Internettside]. USA: W3C; Tilgjengelig fra: http://www.w3.org/standards/webdesign/htmlcss

11. Corporation for National Research Initiatives. [Internettside]. Tilgjengelig fra: http://www.cnri.reston.va.us/

12. HTML5 ROCKS. [Internettside]. Tilgjengelig fra: http://www.html5rocks.com/en/

13. Dublin Core Metadata Initiative. [Internettside] Tilgjengelig fra: http://dublincore.org/specifications/

# 9.0 Vedlegg

## Vedlegg A - Forprosjekt

# EasyUp

Forprosjekt

Ola Kjelsrud og Ole Woldstad

# Innhold

1. MÅL OG RAMMER ..... 85
1.1 Bakgrunn ..... 85
1.2 Mål ..... 85
1.3 Rammer ..... 87
2. OMFANG ..... 88
2.1 Oppgavebeskrivelse ..... 88
2.2 Avgrensning ..... 88
3. Organisering ..... 89
3.1 Roller ..... 89
3.2 Regler ..... 89
3.3 Oppdragsgiver og veileder ..... 89
4. Planlegging ..... 90
4.1 Hovedinndeling av prosjektet ..... 90
4.2 Valg av utviklingsmodell ..... 91
4.3 Statusmøter og beslutningspunkter ..... 91
5. Organisering og kvalitetssikring ..... 92
5.1 Dokumentasjon og utvikling. ..... 92
5.2 Risikoanalyse ..... 92
6. Gjennomføring ..... 95
6.1 Plan ..... 95
6.2 Ganttskjema ..... 96

# 1. MÅL OG RAMMER

## 1.1 Bakgrunn

Høgskolen i Gjøvik har i dag ingen god løsning for deling av produsert undervisningsmateriell. Ansatte og studenter bruker derfor mange forskjellige kommersielle tjenester for opplasting som f. eks:

- https://docs.google.com
- http://imgur.com/
- http://imageshack.us/

Dette fører til at materiell ligger spredt utover mange tjenester. Skolen opplever dermed ofte problemer da flere brukere skal ha tilgang til samme informasjon samtidlig. Ett annet problem er når studenter skal levere oppgaver. Ansatte må ofte laste ned materiell fra flere forskjellige kilder. Dette er både tidkrevende og tungvindt. I tillegg hender det at man ikke får tilgang på opplastet materiell og må derfor anse innleveringen som ikke levert.
Skolen ønsker derfor en felles tjeneste driftet av skolen selv der alt av produsert undervisningsmateriell kan lastes opp og deles ut igjen på en enkel måte.
I 2010 satte Høgskolen i Gjøvik opp en DSpace filserver. Denne blir ikke brukt i dag da brukergrensesnittet er fryktelig tungvint. Ønsket er derfor en samlet tjeneste som kan laste opp innhold til DSpace serveren og publisere dette.

## 1.2 Mål

**Resultatmål:**
Målet med oppgaven er å utvikle en felles publiseringsløsning for ansatte og studenter. Løsningen skal være enkel i bruk, men samtidlig være komplisert nok til å være ett godt alternativ mot andre kommersielle løsninger
Brukere skal enkelt kunne laste opp materiell og dele det med andre. Løsningen skal automatisk hente ut bruker og filinformasjon slik at manuell input holdes til et minimum. Opplastet materiell skal så publiseres med en DSpace server slik at brukere enkelt kan dele materiell. Det skal være mulig å sette rettigheter på opplastet materiell for å begrense tilgang.

Løsningen skal primært kunne brukes under utdanning, men innholdet skal også være tilgjengelig for studenter etter endt utdanning slik at man kan referere til innhold ved f. eks jobbsøknader, med mer.

Løsningen skal fungere på følgende plattformer med følgende nettlesere:

- Windows
- Linux
- Mac OS X.
- Google Chrome
- Safari

**Effektmål:**

Visjonen med prosjektet er at all materiell samles på en enkelt løsning i stedet for å være spredt slik som i dag. Dette oppnås ved at løsningen skal være enkel i bruk slik at alle ansatte og studenter skal velge denne løsningen i stedet for andre kommersielle løsninger.
Løsningen vil spesielt gjøre innleveringer som inneholder større og flere filer enklere både for studenter og ansatte/sensorer. Det skal også bli enklere å dele materiell mellom brukere som f. eks forelesninger og annet undervisningsmateriell.
Løsningen skal være en trygg og stabil plass hvor brukere kan lagre arbeid uten frykt for å miste det.

**Læringsmål:**

I løpet av prosjektet håper vi å kunne utvide vår kunnskap på flere enkelt områder som:

- HTML5
- PHP
- Javascript
- Linux

I tillegg ser vi frem til å kunne bruke alt vi har lært i løpet av studietiden til lage ett større produkt. Det blir interessant å kunne bruke mye tid og energi på ett prosjekt der vi bruker mye av de mindre byggeklossene vi har tilegnet oss til å sette det sammen til ett stort produkt. Prosjektstyring og jobbing kommer til å stå i fokus. Dette vil forbrede oss godt på arbeidslivet.

## 1.3 Rammer

Høgskolen i Gjøvik har i dag en Linux server som kjører DSpace. Vi kommer til å bruke denne serveren til vårt prosjekt. Apache2 blir dermed ett naturlig valg av webserver.

Prosjektet skal være ferdig levert innen 23/05-2012. Dette betyr at prosjektet har en tidsramme på ca 4 måneder etter leveringsfrist for forprosjektet.

# 2. OMFANG

## 2.1 Oppgavebeskrivelse

Ansatte og studenter bruker i dag ingen felles opplastingstjeneste. Dokumenter og annet arbeid ligger spredt på mange forskjellige løsninger. I 2010 satte Høgskolen i Gjøvik opp en DSpace server, men denne blir ikke brukt i dag da brukergrensesnittet for registrering og opplastninger av filer er alt for omfattende.
Oppgaven for gruppen blir å utvikle en felles løsning som ansatte og studenter kan bruke til å publisere sitt arbeid enten det er video, lyd, bilder, dokumenter, eller andre type filer. DSpace står for publisering av filene, men mangler en god opplastningsmetode. Hovedfokuset på prosjektet blir derfor å få til et opplastningsgrensesnitt som blir enkelt nok slik at det blir brukt, men samtidlig gir nok informasjon slik at DSpace kan publisere dette på en god måte. Det skal være støtte for både Windows, Linux og Mac OS X slik at alle ansatte og studenter kan ta i bruk løsningen.
Vi ser for oss et webgrensesnitt basert på den kommende html5 standarden der vi implementerer ”Drag and Drop”. Filer skal enkelt kunne droppes inn, bruker og filinformasjon skal hentes ut via henholdsvis innloggingsinformasjon fra nettsiden og informasjon fra headeren på filen. Det er også ønskelig at man kan legge eller endre informasjon i headeren på filene i det man laster opp slik at brukere som laster ned filene kan dra nytte av denne informasjonen. En pause/resume funksjon må også implementeres da det ofte er snakk om større filer som f. eks video som skal legges ut.
Når opplastingsdelen av prosjektet er fungerende blir utfordringen å integrere dette med DSpace slik at vi kan publisere opplastet innhold. Her vil rettigheter på hvem som skal ha tilgang til filene og hvordan dette skal implementeres bli den største utfordringen.

## 2.2 Avgrensning

- Publiseringsløsningen består av opplastingsgrensesnittet og DSpace serveren. Det tas ikke hensyn til at disse skal fungere som separate løsninger.

- Filrettigheter begrenses til åpen for alle, eller privat innad skolen
- Det vil ikke være noen form for virussjekk og lignende på opplastet materiale.

# 3. Organisering

## 3.1 Roller

Ola Kjelsrud ble valgt som prosjektleder for gruppen. Prosjektleder har ansvaret for dokumentasjon og at tidsfrister blir opprettholdt.
Ole Woldstad har ansvaret for at nettsiden og at loggen holdes oppdatert.

## 3.2 Regler

- Prosjektleder har dobbeltstemme i alle avgjørelser
- Kostnader som ikke dekkes av arbeidsgiver deles mellom gruppemedlemmene.
- Alle gruppemedlemmer har rett til å signere på vegne av gruppen.
- Dersom ett av gruppemedlemmene ikke utfører arbeid som avtalt sendes en skriftlig advarsel
- Hvis dette ikke fører til forbedringer vil dette føre til ett møte med veileder der personen kan bli utelatt fra gruppa.
- Logg skal føres på slutten av hver dag hvor arbeid er utført.

Dokument med grupperegler ligger som Vedlegg A.

## 3.3 Oppdragsgiver og veileder

Oppdragsgiver er Høgskolen i Gjøvik avd. IMT. Kontaktperson er Kjell Are Refsvik.
Veileder ved Høgskolen i Gjøvik er Øivind Kolloen

# 4. Planlegging

## 4.1 Hovedinndeling av prosjektet

Løsningen skal brukes av ansatte og studenter til opplasting av undervisningsmateriell og innleveringer. Dette krever god stabilitet og oppetid. Løsningen skal også fungere på flere plattformer.
Vi baserer oss på html5. Dette er ny teknologi der standarden enda ikke er satt. I tillegg har vi flere nettlesere som jevnlig endrer støtten for html5. Endringer på veien til en universell standard fører til at vi må belage oss på endringer underveis både i kravspekk og implementering av løsningen.
Vi ser for oss 2 hovedinndelinger av prosjektet med flere mindre inndelinger under disse.

**Opplasting:**

Dette vil være hoveddelen i prosjektet. Her skal det utvikles ett opplastingsgrensesnitt med mye funksjonalitet. Vi ser for oss å starte enkelt og legge mer og mer funksjonalitet.
Viktige punkter inkluderer:

- Enkel opplasting via drag and drop
- Brukerinnlogging via feide-bruker
- Hente ut headerinformasjon fra filene og eventuelt å editere det før opplasting.
- All funksjonalitet fungerer sammen til et enkelt grensesnitt.

**Publisering:**

Her vil jobben være å få DSpace serveren til å håndtere opplastet materiell.
Viktige punkter inkluderer:

- Rettigheter på materiell som lastes opp
- Publisering/Visning av materiell

Oppdragsgiver hadde ideen om en opplastingsmappe da vi fikk presentert oppgaven. Vi så fort at med en slik type løsning ville bli problematisk å implementere på forskjellige plattformer. Det ville mest sannsynelig bli en selvstendig løsning for hver plattform. Vi så at en mer ideell løsning ville være ett webgrensesnitt da dette er mer kryssplattform vennelig. Ideen ble veldig godt mottatt av oppdragsgiver, og sammen videreutviklet vi spesifikasjoner og funksjonalitet.

## 4.2 Valg av utviklingsmodell

Ved valg av utviklingsmodell utelukket vi raskt tyngre planbaserte metoder som RUP og fossefallsmetoden da endringer underveis i prosjektet er ventet. Det er heller ikke spesielle krav om dokumentasjon underveis. Vi er i tillegg kun 2 personer på gruppa slik at å følge en modell slavisk vil være lite effektivt. Vi har også mulighet med hypping kontakt med oppdragsgiver. Vi så derfor at en utvikling basert på en iterativ og inkrementell metode passer oss best. Her kan vi dele opp prosjektet som nevnt over og hele tiden legge til ny funksjonalitet.
Vi har derfor valgt eXtreme Programming (XP) som rammeverk, men med visse modifikasjoner. XP er en fleksibel modell som takler endringer underveis godt. Vi vil ikke benytte oss av ”On Site Customer” fult ut slik som XP, men vi vil ha hyppige møter der vi for hver modul diskuterer tenkt funksjonalitet.
Vi kommer også til å benytte gjenbruksorientert utvikling. Det finnes mange eksempler på htlm5 og javascript funksjonalitet som vi kan tenke oss å bruke.

Vi ser for oss at vi benytter disse hovedpunktene i vår utvikling:

- Inkrementell og iterativ utvikling. Legger hele tiden til ny funksjonalitet.
- Kontinuerlig testing. Tester hver ny funksjon, både alene og sammen med andre.
- Parprogrammering. På enkelte moduler vil vi benytte parprogrammering.
- Brukerinteraksjon under utvikling. Tilbakemelding fra oppdragsgiver og brukere.
- Daglige møter (Stand up meetings). Mellom gruppemedlemmer.
- Refactoring. Renskriving av kode underveis selv om produktet fungerer.
- Gjenbruk. Modifisere kode andre har laget.

## 4.3 Statusmøter og beslutningspunkter

Vi kommer til å holde ett kort veiledermøte hver uke der vi orienterer om statusen på prosjektet. I tillegg brukes dette møte som veiledning.
I hvert månedsskifte kommer vi også til å oppdatere nettsiden med en post der vi:

- Tar ett tilbakeblikk på forrige måned.
- Kommenterer utfordringene som vi vet kommer neste måned.
- En totalvurdering av statusen til prosjektet.

Store beslutninger tas på møter med kontaktperson Kjell Are Refsvik. Mindre beslutninger tas med veileder eller i gruppen alene.

# 5. Organisering og kvalitetssikring

## 5.1 Dokumentasjon og utvikling.

All dokumentasjon vil opprettholde Høgskolen i Gjøvik sine rettingslinjer for bachelor/masteroppgaver. Hele prosjektet vil være tilgjengelig for allmennheten da verken oppdragsgiver eller gruppen har noe imot dette.
Utvikling av prosjektet vil foregå på Linux plattform. Kode kommer hovedsakelig til å bestå av HTML5, PHP, og Javascript. Vi kommer til å legge vekt på å dokumentere kode underveis i prosjektet, mens rapporten vil bli gjort etter implementeringsprosessen er ferdig.
Siden vi er bare 2 personer kommer vi til å benytte automatisk backup satt opp ved hjelp av en cron jobb som kjører hver dag kl 04:00. Vi ser ikke for oss noen problemer angående forskjellige versjoner da vi som regel jobber sammen og koordinerer hva som skal gjøres.

## 5.2 Risikoanalyse

**Under utvikling**
De største risikomomentene under utvikling ligger i usikkerheten rundt HTML5 standarden og de ulike nettleserne sin kompatibilitet med ”Drag and Drop” og pause/resume funksjonalitet. Prosessen rundt datakommunikasjon mellom nettleser og DSpace server er også et område med noe usikkerhet.
Et tredje risikomoment er hvorvidt tilbakeføring av endrede metadata til filer lokalt på maskinen lar seg gjøre før opplastning. Implementeringen av dette vil sannsynelig kreve ulike løsninger på de forskjellige plattformene.

**Under drift**

Det er tre utfall som kan betegnes som mer eller mindre kritisk når det gjelder bruk/drift av systemet:

1. **Tap av fil eller filinformasjon:**

Filer kan gå tapt under opplastning til server. Her er det snakk om å bygge inn en pause/resume funksjonalitet som skal ta seg av om tilkoblingsproblemer. En slik funksjon vil eliminere problemer der brukeren mister tilgang til nettverket eller tjenesten.
Filinformasjon kan tapes på server. Dette kan enten skje som et resultat av filbehandling i DSpace, eventuell kode vi produserer for behandling av data på server, eller maskinvarefeil. Backup og maskinvarefeil er HIGs IT-avdeling sitt ansvar, men er likevel en situasjon vi må ha i tankene.

2. **Uvedkommende får tilgang på privat materiale.**

Alvorligheten av dette er sterkt avhengig av filinnhold. De mest sensitive dataene som skal plasseres på serveren er i første omgang lærerens oppgavefasiter, og materiell beskyttet av opphavsrettsloven.
Konsekvensen av en lekkasje er ikke ønskelig, men heller ikke kritisk.
Det er en mulighet for at mere kritisk materiale også skulle lagres på serveren etter hvert, og konfidensialitet er derfor en spesifikasjon som skal med i planleggingen.

3. **Systemet er ustabilt eller utilgjengelig.**

Systemet kan oppfattes som ustabilt eller tregt ved ulike kodefeil.
Dette er kritisk for programvaren og bør ikke forekomme mer enn en gang annenhver uke. Vi har valgt å ta med elementer som par-programmering i utviklingen av kritiske funksjoner, og refactoring av kode for å sikre en mest mulig feilfri og stabil løsning. Stabiliteten til DSpace og eventuelle tilleggsmoduler er også avgjørende.  Nettverk er en annen viktig faktor for tilgjengelighet, også denne er utenfor vår kontroll.
Det er flere faktorer vi ikke har mulighet til å ha full kontroll over. Testing av system opp mot gitte krav underveis og etter hver modul er derfor eneste måte å sikre stabilitet på.

| Risikomoment | Sannsynlighet | Alvorlighet | Estimert | Prioritet |
|---|---|---|---|---|

| | | | Risiko | |
|---|---|---|---|---|
| HTML5 inkompatibilitet i nettlesere. | Middels | Høy | Middels+ | Høy |
| Kommunikasjonsproblemer mellom nettleser og DSpace server | Lav | Høy | Middels | Middels |
| Problemer med tilbakeføring av metadata til fil før opplastning | Høy | Lav | Middels | Middels |
| Tap av fil eller filinformasjon | Middels | Middels | Middels | Middels |
| Uvedkommende får tilgang på privat materiale.(avhengig av innhold) | Lav | Middels | Lav | Lav |
| Systemet er ustabilt eller utilgjengelig. | Middels | Høy | Middels+ | Høy |

**Tiltak**

Vi starter med implementering av opplastingsgrensesnittet for å adressere usikkerheten rundt inkompatibilitet og overføring av metadata. Vi har lite makt over nettleserfunksjonalitet og HTML5 standarden slik at vi må rette oss etter hva som fungerer per dags dato.

Tap av filer under opplasting adresseres ved å utvikle en pause/resume funksjon. Dette er også knyttet til nettleserkompatibilitet.

forhindring av uvedkommendes adgang til opplastet materiale via DSpace har lav prioritet i begynnelsen, grunnet lite sensitivt innhold. Dette må ikke forveksles med serverens sikkerhet i seg selv. Autentisering skal inn i form av brukerinnlogging via feide. Skolens og Feides sikkerhetskrav skal ivaretas.

Stabilitet er viktig for at oppdragsgiver og brukere skal bli fornøyd og for at løsningen skal bli tatt i bruk. I tillegg vil vi ta forhåndsregler som nevnt ovenfor for å adressere mulige problemer før de oppstår.

# 6. Gjennomføring

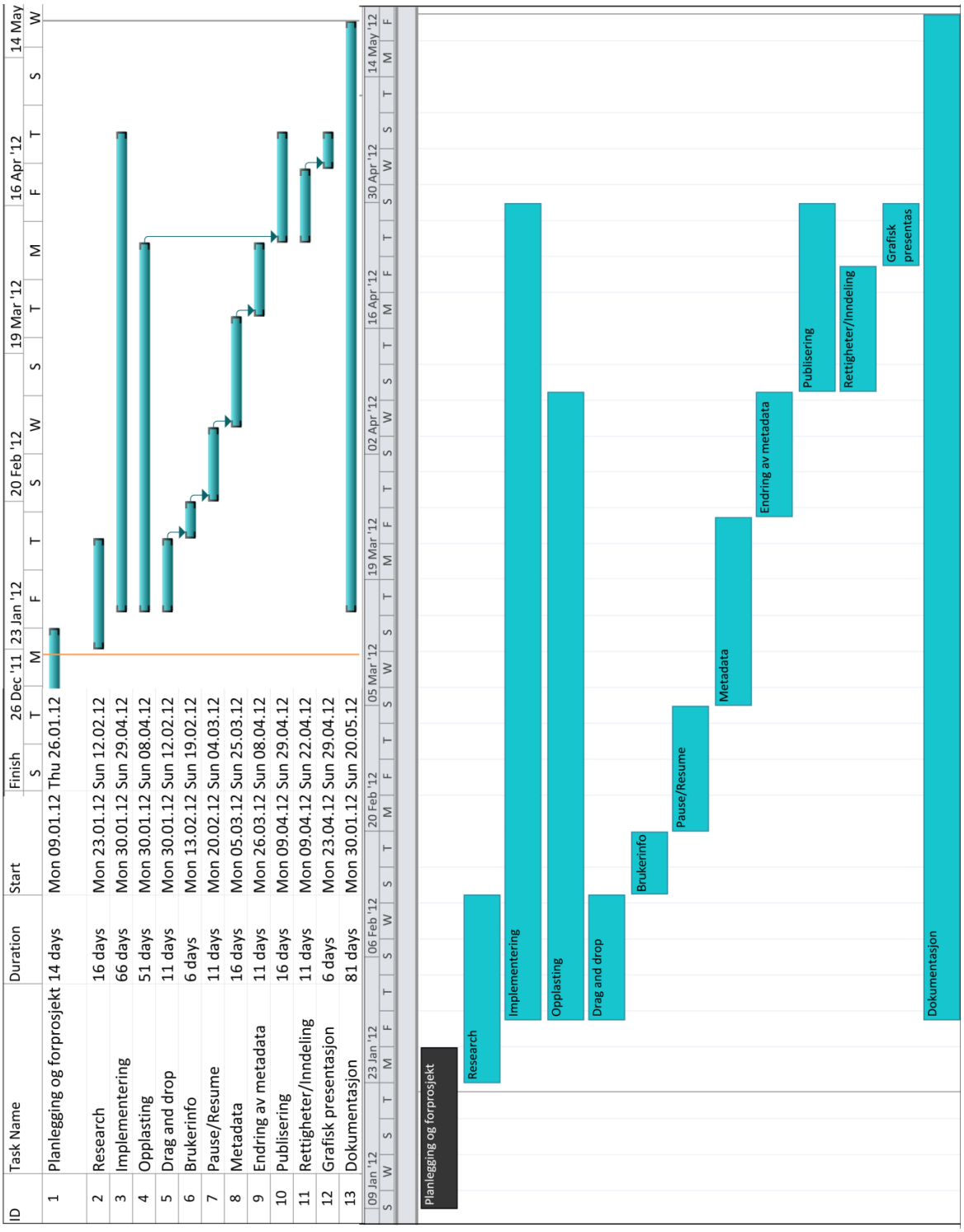
## 6.1 Plan

Gjennomføringen av prosjektet vil i hovedsakelig fire hoveddeler.

- Planlegging av prosjektet
- Research og utprøving
- Implementeringsfase.
- Dokumentasjonsfase

Disse 4 delene vil ha en del overlapping. Research og utprøving vil være forholdsvis kort alene, men vil foregår parallelt med implementeringen av løsningen da vi ikke kan teste ut alle deler før alt er på plass.
Implementeringen av løsingen vil være mest tidkrevende. Denne fasen vil vi dele opp i mange små inkrementer slik at vi kan legge til funksjonalitet litt etter litt.
Dokumentasjon vil ikke ha stor fokus underveis, men spesielt all kode og dens funksjonalitet skal dokumenteres. Hovedjobben vil her komme på slutten av prosjektet da rapporten skal utvikles.

## 6.2 Ganttskjema



## Vedlegg B - Prosjektavtale

HØGSKOLEN I GJØVIK

# PROSJEKTAVTALE

mellom Høgskolen i Gjøvik (HiG) (utdanningsinstitusjon),

_____

_____ (oppdragsgiver), og

Ola Kjelsrud, Ole Woldstad

_____

_____ (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 1/1 - 2012 til 23/05 - 2012

   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 oppgavebesvarelsen med vedlegg (pdf) i Fronter. I tillegg leveres et eksemplar til oppdragsgiver.

8. Denne avtalen utferdiges med et eksemplar til hver av partene. På vegne av HiG er det dekan/prodekan 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): Øivind Kolloen

Oppdragsgivers kontaktperson (navn): Kjell Are Refsvik

Student(er) (signatur): Ola Kjelsrud dato 18/1-2012

Ola Wahlstad dato 18/01-2012

_____ dato _____

_____ dato _____

Oppdragsgiver (signatur): Kjell Are Refsvik dato 14/1-2012

IMT Dekan/prodekan (signatur): _____ dato 15/2-2012

*Versjon Januar 2011hb*

## Vedlegg C - Grupperegler

# Grupperegler

- Prosjektleder valgt: Ola Kjelsrud
- Prosjektleder har ansvaret for gjennomføring av oppgaven.
- Prosjektleder har dobbeltstemme i alle avgjørelser
- Kostnader som ikke dekkes av arbeidsgiver deles mellom gruppemedlemmene.
- Alle gruppemedlemmer har rett til å signere på vegne av gruppen.
- Dersom en av gruppemedlemmene ikke utfører arbeid som avtalt sendes en skriftlig advarsel
- Advarsel kan sendes ut etter følgende brudd:
  -Utelatt arbeid.
  -Avtalebrudd
- Hvis dette ikke fører til forbedringer vil dette føre til ett møte med veileder der personen kan bli utelatt fra gruppa.

---------- ----------

Ola Kjelsrud Ole Woldstad

## Vedlegg D - Statusrapporter

# Statusrapport – 1. februar

Da var første måned over!

Vi startet tidlig med planlegging og kom godt i gang med forprosjektrapporten.
Rapport og prosjektavtale ble levert 27 januar i tide.

Møte med oppdragsgiver (Kjell Are Refsvik) ble også en suksess da han
var enig i våre prioriteringer og forslag for funksjonalitet.

Offisiell oppstart av implementerings fasen av prosjektet ble 27 januar.
Vi har kommet godt i gang med research og testing.

En "Proof of Consept" side er laget der vi demonstrerer "Drag and Drop" upload.
Her kan vi hente ut info om filen på klient siden, for og deretter
laste den opp til server.

Opplastingsdelen av prosjektet er på langt nær ferdig, men vi har nå stort sett
funnet  ut hvordan alt skal gjøres.

Vi har fortsatt ikke fått webområdet av IT@HIG så vi venter i spenning!

# Statusrapport - 1. mars

Ny måned, ny statusrapport!

Vi startet med å utvikle ett standalone opplastingsgrensesnitt ved hjelp av HTML5/JavaScript/PHP. Grenesnittet fungerer nå med alle funksjoner utenom pause/resume. Vi satte dette på vent da vi ikke viste hvordan vi skulle importere filer inn i DSpace.

Etter en del trøbbel har vi også fått nyeste versjon av DSpace til å fungere på vår testserver. I løpet av denne prosessen kom vi over noe som heter for Sword (EasyDeposit).
Dette er en lettvekts protokoll som vi kan bruke til å lage vår egen webapp til å laste opp filer til DSpace med. Dette endrer vår plan noe, ettersom vi her slipper manuell integrering av vår opplastingsgrensesnitt. Vi kommer i stedet til å bruke Sword slik det ser ut i dag. Vi har bare rukket så vidt å teste dette, og foreløpig ser det veldig lovende ut.

Dermed kommer vi til å utvikle vår egen swordklient med SWORD sitt PHP bibliotek, og sy denne sammen med vårt html5 opplastningsgrensesnitt. Planen er at denne skal ta i mot filene fra opplastningsgrensesnittet, pakke og legge de inn i DSpace. DSpace er en av mange repositorier som støtter sword for enkel opplasting.

Planen videre er å utforme use case og oppdatere tidsskjema, da denne ikke lenger stemmer med hva som er gjort/skal gjøres. Vi håper at vi før neste rapport har en fungerende opplastingsløsning til DSpace via Sword.

# Statusrapport - 1. april

Dette er ikke en aprilspøk.

Prosjektet har virkelig tatt seg opp. Vi har nå en fungerende løsning som vi har vist frem til både veileder og oppdragsgiver. Begge virket fornøyd selv om dette er en meget tidlig versjon.

Som nevnt forrige måned var Sword det store nye. Vi merket derimot fort at Sword var så mye mer enn det vi trengte. Ett alternativ var å bruke DSpace sin egen bash importer.

Vi trappet derfor opp på møte med veileder og presenterte 2 mulige løsninger og hadde som mål å komme ut igjen med en. "Gjør begge deler" var beskjeden vi fikk.

Vi kom frem til at vi skulle starte med DSpace bash importer da denne passet oppgavebeskrivelsen på en prikk. Når denne løsningen var ferdig skulle vi starte med Sword. (Les møterefferat for mer informasjon)

Idag sitter vi med en løsning som fungerer, men på langt nær er ferdig. Vi mangler litt funksjonalitet, pluss mye feilhåndtering. Vi regner med at iløpet av første halvdel av april vil denne løsningen være ferdig leveringsklar.  Planen er da og starte med Sword og få til en løsning også her før vi virkelig må fokusere på rapporten siste del av mai.

Alt i alt har vi virkelig fått fart i sakene denne siste måneden. Vi har gjort mye planlegging i starten som vi ser vi får nytte av nå når vi begynner å lage selve produktet. Selv om vi har måtte planlegge endel på nytt har dette hjulpet oss i å ikke starte på feil sted. Vi har fulgt den nye planen og unngått feilskjær. Dette sparer oss mye tid istedet for å kjøre igang å kode tidlig i prosjektet.

# Statusrapport – 1. mai

Prosjektet nærmer seg slutten og vi har i dag en løsning som fungerer ut i fra oppdragsgivers krav.

Dette er bash import delen av prosjektet. Dette tok litt lengre tid enn antatt.
Noen grunner til dette:
-Mye tilpassning til forskjellige plattformer/Nettlesere.
-Mye funksjonalitet som tok lengre tid en planlagt. F.eks “Pause/Resume”
-Kragebeinsbrudd (Ola)
-Kommentering og rydding i kode

Både oppdragsgiver og IT-Tjenesten har sett prosjektet og syntes det så spennende ut.
Det som er igjen på denne delen er sikkerhetstesting og eventuell utbedring. Her kreves det er testperiode, dette blir utenfor vår oppgave.

Uansett så er alle parter enige om at dette kommer til å bli en god byggeblokk og ett stort steg videre i å ta i bruk DSpace serveren til publisering av læringsobjekter.

Vi kom frem til sammen med veileder at vi skulle droppe sword delen da tiden blir knapp på slutten. Dette gjør vi for å fokusere mer på dokumentasjonen da det er dette som danner grunnlaget for vurderingen sensor gjør.

Vi kommer derimot til å skrive en del teori rundt sword i dokumentasjonen da vi har satt oss inn i også dette i løpet av prosjektperioden.

## Vedlegg E - Møtereferater

# 07/05-2012

Veileder:

Vi diskuterte status og tiden frem mot levering. Vi kom frem til at vi dropper å utføre sword delen av oppgaven da tiden begynner å bli knapp.

Dette er for å fokusere på rapporten som danner grunnlaget av sensors vurdering.

Vi kommer derimot til å nevne sword i rapporten. Hvordan vi ville gjort det, hva som må gjøres og hvor lang tid det ville tatt.

Øivind ville gjerne ha ett utkast av rapporten fredag 18 mai slik at han kan gi tilbakemeldinger på den før levering.

---

# 02/05-2012

Oppdragsgiver:

Vi møtte på en sykmeldt Kjell Are på skolen og fikk tatt ett kort møte med han angående feedbacken vi hadde fått før påske.

Han ville gjerne at bruker kunne velge mellom Copy Right, /Copy Left(Creativecommons) på publiseringen.

Påpekte at sikkerhet er viktig spesielt på enkelte publiseringen som inneholder personinformasjon.

Dette er noe som DSpace administrator må tenke på ved inndeling av grupper og tilgang.

Vår opplastingsløsning sørger for selve publiseringen, hvem som har rettigheter til denne er opp til administrator.

---

# 30/03-2012

Veileder:

Kort presentasjon av DSpace bash import løsningen vi har laget:

-Brukeren logger inn med sitt feide brukernavn og passord.

-Filer droppes inn (En eller flere)

-Brukeren får opp en liste over filene. I denne listen inkluderes en liten forhåndsvisning (Thumbnails) av bilder. For andre typer filer har vi ett ikon som indikerer filtype.

-Brukeren kan deretter velge hvor i DSpace han vil publisere filene, samt legge til: Tittel, beskrivelse og andre metadata.

-Når all nødvendig informasjon er fylt ut får brukeren tilgang til en opplastingsknapp som starter importen. (Progressbar osv indikerer fremgang på opplastingen)

Øivind var positiv til løsningen så langt. Han var enig at det var en god start, selv om det er endel jobb til vi kan si oss ferdig med denne løsningen.

Vi kom fram til noen punkter som er viktige videre:

-Kommentere i både kode og rapport hvor vi har hentet kode/inspirasjon.

-Se på sikkerthet i løsningen (Input fra bruker, eventuelle feil som kan oppstå, osv)

Oppdragsgiver:

Vi hadde også ett meget kort møte med oppdragsgiver over skype der vi demonstrerte løsningen nevnt over.

Kjell Are virker fornøyd med hva han så selv om det er en tidlig versjon og ingenting er satt i stein på verken design eller funksjonalitet.

Vi kom frem til noen punkter sammen som vi skal se videre på:

-Implementere ett valg der brukeren velger under hvilke lisens filene som importeres skal ligge under. Dette skal være forholdsvis enkelt å implementere.

-Enkel måte å endre designet på løsningen uten å endre funksjonalitet. Dette er alleredet gjort da vi gjør all design via css.

-Se litt på muligheten å bruker flere metadata fra filene som lastes opp slik at brukere kan søke på flere verdier i DSpace for innhold. Dette er også lett å implementere da vi bare kan lage flere felter i xml filen som inneholder data om importen.

---

# 12/03-2012

Veileder:

Vi presenterte 2 mulig løsninger på oppgaven:

1. Sword. Lage vår egen sword klient som bruker vårt opplastingsgrensesnitt til å få filene lastet opp på server for å deretter håndtere import inn i DSpace via sword API.

2.Bruke vårt opplastingsgrensesnitt til å laste opp filer til server. Deretter bruker vi DSpace egen bash import scripts til å importere filene inn i DSpace.

Metode 1 vil definitivt være mest jobb. Vi må først virkelig sette oss inn i sword, for å deretter utvikle vår egen klient.

Det positive med denne metoden er at løsningen vi utvikler enkelt kan tilpasses andre DSpace installasjoner/Repositorer.

Metode 2 passer meget godt til oppgavebeskrivelsen, og vil kunne utføre opplastingen akkurat slik som både vi og oppdragsgiver hadde planlagt.

Det negative med denne metoden er at løsningen vil kun fungere på vår DSpace server. Vi vil også ha færre muligheter for å utvide løsningen uten mye kodeendringer.

Øivind påpekte også at metode 1 vil kunne gi oss erfaring med å bruke andres kode. Vi ble etterhvert enige om å starte med metode 2, altså bash import. Vi kommer derfor til å utvikle metode 2 først, ettersom denne løsningen passer veldig godt med tanke på oppgaven. Vi regner med at vi blir ferdige med denne i god tid før levering slik at vi også kommer til å prøve oss på metode 1. Forhåpentlighvis kommer vi til å rekke å utvikle en god løsning også på metode 1, men dette blir altså ikke førsteprioritet.

---

# 20/02-2012

Veileder:

Kort statusrapport fra gruppa til veileder.

Ingen av partene hadde mye å komme med, da vi for tiden prøver å komme inn i hvordan DSpace virker og hvordan vi kan implementere vår opplastingsgrensesnitt.

---

# 05/02-2012

Veileder:

Vi presenterte vår ide om "Drag and Drop" opplastingsgrensesnitt.

Vi fikk bekreftet at vi kunne bruke HTML5 med File-API på klient siden for å hente ut metadata, JavaScript med XMLHttpRequest() for å håndtere opplasting, og PHP på serversiden for å lagre filene på server.

Vi diskuterte flere måter å lagre filer på:

PHP's php://input stream for å streame filene til disk. Dette virker som den beste måten sålangt.

PHP's FILES[] array. Denne metoden fungerer dårlig for større filer pga måten filene mellomlagres.

Grunnen til at vi valgte streaming metoden er for å kunne implmenetere en "Pause/Resume" funksjon.

---

# 30/01-2012

Veileder:

Rask gjennomgang av forprosjektrapport med tilbakemelding.

Øivind hadde ikke noe spessielt å kommentere på rapporten. Men arbeidet vi hadde gjort foreløpig var bra.

Vi fikk dermed tillatelse til å fortsette vår bacheloroppgave.

Vi presenterte vår fremdriftsplan og vår tolkning av oppgaven og fikk bekreftet at vi var på rett vei.

Veileder var enig i vår plan angående hvordan vi teknisk skal løse oppgaven.

---

# 10/01-2012

Veileder:

Oppstartsmøte med veileder.  Diskuterte oppgaven veldig generelt.

Fikk noen tips på følgende:

-Forprosjektet

-Hjemmeside

-Loggføring

---

# 15/11-2011

Oppdragsgiver:

Første møte med oppdragsgiver.

Møtet startet ved at Kjell Are Refsvik gav oss en utdypende forklaring av oppgaven.

Vi fikk presentert en problemstilling som møtte media elever på skolen. De trengte en god måte å dele og å laste opp større filer på, enn det som var tilgjengelig på nåværende tidspunkt. Det ideelle hadde vært en løsning som støttet resume funksjonalitet, da elevene til tider satt på dels ustabile trådløse tilkoplinger. Løsningen skulle selvfølgelig også kunne brukes av de andre avdelingene på skolen.

Vi fikk en litt dypere forklaring på hva DSpace var, og hva som var svakhetene med DSpace sin eksisterende løsning for opplasting.

Vi hadde sett litt på oppgaven og tenkt på mulighetene for å løse denne med et webgrensesnitt.

Dette virket som en god måte å løse det på på grunn av både "drag and drop" funktionaliteten som var ønsket. Og siden det skulle fungere både på mac og windows maskiner.

Kjell Are hadde forestilt seg dette løst anderledes; med kode som kjørte på klient (plattformavhengig) og med exiftool på serversiden for å hente ut metadata fra filer.

---

## Vedlegg F - Logg

08/05.2012 - 23/05.2012

Begge:

Vi velger å lage det siste punktet i loggen som et sammendrag.

Vi har begge jobbet sammen med å fullføre bacheloroppgaven. Hovedpunktet har vært dokumentasjonen, men også jobbing med å rydde i kildekode er blitt gjort.

---

07/05.2012

Begge:

4t :

Møte med veileder.

Besluttet å gjøre 7 lisensvalg tilgjengelig for bruker. 6 CreativeCommons + vanlig Copyright.

Laget 7 forskjellige predefinerte tekst dokumenter som automatisk kopieres over og legges til i publiseringen til brukeren avhengi av lisensvalg.

Her er det enkelt å gjøre endringer hvis oppdragsgiver skulle ønske dette senere.

---

04/05.2012

Begge:

4t :

Jobbet videre med lisensvalg og publisering.

---

03/05.2012

Begge:

4t :

Jobbet videre på rapporten. Lagde statusrapport til veileder.

Fra gårdagens møte med oppdragsgiver startet vi med utviklingen av lisensvalg under publisering.

---

02/05.2012

Begge:

5t :

Møte med Einar på IT-Tjenesten for å få formatert matuku.HiG.no med debian slik at vi kan sette opp nyeste DSpace versjon.

Einar ville gjøre dette selv da IT-Tjenesten hadde retningslinjer for dette.

Møte med oppdragsgiver Kjell Are Refsvik. (Se møtereferat)

Startet på ett standarisert system for språkvalg under metadata. (language.php)

Vi vil kun at brukere skal kunne velge språk ut ifra ISO standard: RFC1766

Startet med hovedrapporten. Hva som skal inn og hvordan layout osv skal være.

---

30/04.2012

Begge:

5t:

Lagde total progressbar for opplasting av flere filer.

Slett knappene fjernes etter at bruker trykker "Upload" knappen.

Kommentering av kode og siste finpuss på løsningen.

Avtalte med Einar på IT om at vi skulle komme og formatere og sette opp matuku.HiG.no onsdag kl 10:00.

Vi valgte å sette opp skolens server helt på nytt pga:

-Matuku kjører gammel DSpace versjon.

-Kjører kun tomcat på CentOS. Noe ingen av oss har testet på

-Vi kommer til å kjøre debian. Noe vi har testet med under hele prosjektet så langt.

Vi sier oss ferdig med bash import delen av prosjektet idag. Vi har en fungerende løsning som både vi og oppdragsgiver er fornøyd med.

Det som er igjen er å implementere den på matuku.HiG.no og teste ved hjelp av andre studenter.

Fremover kommer vi til å starte jobben med sword og etterhvert fokusere mer og mer på dokumentasjonen.

Vi setter deadline for sword implementasjonen mandag 14 mai. Den siste uka trenger vi for å ferdigstille dokumentasjonen.

---

27/04.2012

Begge:

4t :

Kommentering av kode.

---

26/04.2012

Begge:

4t :

Fikk løsningen til å fungere på alle browsere (Chrome, Firefox, Safari).

Chrome og Firefox støtter 100%. Mens thumbnail og resume funksjonalitet ikke fungerer på Safari.

---

25/04.2012

Begge:

3t :

Resume funksionalitet er ikke støttet i safari enda (finnes ingen måte å få dette til på med JavaScript per dags dato).

Lagde workaround så safari også kan laste opp.

Generell css justering i forbindelse med dropbox.

Arbeid med å få thumbnail previews på safari, eventuelt et alternativ.

---

24/04.2012

Begge: 6t:

Endret og fikset progressbar så den er laget i css og kan styles.

Noe rydding og omskriving av kode.

Vi startet på oppgaven med å få stil/ny funksionalitet til å fungere på firefox og safari i tillegg til chrome.

---

20/04.2012

Begge 2t:

Endret progressbar slik at den starter på riktig sted på halvferdige filer.

Endret ID på knapper og filliste slik at disse ikke har øvre grense

---

19/04.2012

Begge 3t:

Laget bekreftelse og "venteside" på publisering.

---

18/04.2012

Begge 4t:

Ferdigstilte "Pause/"Resume"  funksjonalitet,

---

17/04.2012

Begge 4t:

Startet på "Pause/"Resume"  funksjonalitet,

Fremgangsmåte:

1.Php sender med filnavn på som ligger igjen på server til JavaScript.

2.Hvis disse filene blir valgt på nytt slices disse av JavaScript og sendes fra etter siste mottatte byte.

3.Php tar i mot filer og appender data på filen, eller oppretter ny fil om den ikke eksisterer.

---

17/04.2012

Begge 4t:

Re-kodet xmlhttprequest og forenklet funksjoner. Gikk gjennom, så over og ryddet kode.

---

15/04.2012

Begge 3t:

Vi har nå kommet til ett punkt hvor all funksjonalitet untatt "Pause/Resume" er fungerende. For å få til den siste biten så vi at endel endringer måtte foretas.

Vi har derfor bestemt oss for å rekode hele prosjektet. Vi starter med å lage ny psykode for hele prosjektet for å få informasjonsflyt mellom server og klient best mulig.

Vi vil selvfølgelig bruker mesteparten av koden vi allerede har laget, men med noen små endringer. De største endringene vil være hvor og når ting gjøres. Koden vi har idag består av endel unødvendige steg som vi kan slå sammen. For eksempel kan vi kutte ned antall xmlhttprequest objekter som sendes mellom server og klient.

Dette høres veldig drastisk ut, men vi kommer ikke til å bruke mye tid siden all funksjonalitet allerede er laget.

Ole 1t:

Innhold i metadataform lagres nå unna før oppfrisking så innholdet forblir i formen selv om ny collection / nye metadtafelt velges.

---

14/04.2012

Ole 3t:

Laget infobokser for beskrivelse av metadata. Små bokser som flyter over siden rett ved musepekeren når du holder den over metadataknappene.

---

09/04-2012

Ole 5t:

Laget ekstra felter som kan legges inn som metadata. Kom fram til et system med knapper som beholder plasseringen sin selv om de er valgt vil være mest brukervennlig da hver enkelt felt kun er ett klikk unna. Dette kombinert med at man slipper å orientere seg på nytt for hvert klikk.

Slik man måtte om knappene skulle forsvunnet etter trykk og neste knapp rykket en plass innover.

Epost validering på plass samt en funksjon som forkorter for lange filnavn i fillisten.

---

30/03-2012

Begge 3t:

Møter med veileder og oppdragsgiver. (Se møterefferater)

---

29/03-2012

Begge 7t:

Alle bilder blir nå lest inn via Filereader metoden på klient siden. Vi genererer en thumbnail for hvert bilde som vises i listen med filer.

Alle filer får en liten thumbnail. Hver filtype har sitt eget ikon som brukes hvis det ikke er ett bilde. Dette for å lett identifisere filtype.

Laget en ny banner og gjorde små justeringer på divs. en liten finpuss på interface før presentasjon.

---

28/03-2012

Begge 4t:

Begynte arbeidet med å få til thumbnails for bilder vi laster opp. Vi bruker dette som forhåndsvisning i lista med filer før opplasting.

---

27/03-2012

Begge 6t:

Ryddet mye i kode. Jobbet mye med kode og filstrukturen. Luket ut noen filer vi ikke trengte lengre, og slo sammen noen filer.

---

26/03-2012

Begge 5t:

Collection må nå være valgt i tillegg til at påbudte metadata må være utfylte. opplastet dato blir nå overført i bakgrunnen, bruker kan fortsatt selv velge dato laget.

Vi diskuterte mest logisk oppsett på div-ene våre i forhold til informasjon og praktiske hensyn. dette er nå implementert. stilen på siden er fortsatt urørt, kun plassering er endret.

---

25/03-2012

Begge 5t:

All data som automatisk dukker opp, pluss data brukeren selv skriver inn i formen på opplastingssiden blir nå sent til php via xmlhttprequest.

På php siden har vi skrevet kode som tar imot all formdata og lager en dublin_core.xml fil. Denne bruker av DSpace bash import scriptet.

Viktige felter i formen blir nå sjekket om er utfylt. om disse ikke er utfylt, deaktiveres opplastningsknappen og feltene merkes med rødt.

Formfeltet er usynlig om ingen filer er droppet. brukernavn og passord blir nå sjekket og renset.

---

23/03-2012

Begge 6t:

Funksjonalitet som er blitt laget:

- Hver fil har  nå sin egen progressbar.

---

22/03-2012

Begge 6t:

Har skrevet om stort sett hele "Drag and Drop" JavaScript delen. Bedre struktur, oversikt og kode.

Laget PHP script som kobler seg på den åpne LDAP serveren til skolen og henter ut Navn, epost og telefonnummer(sistnevnte kun for ansatte).

Dette blir brukt i metadataformen som skal brukes til å bygge en xml fil til publiseringsscriptet.

---

21/03-2012

Begge 6t:

Begynte jobben på å få til progressbar på opplasting av filer.

PHP lager nå mappestrukturen som trengs for å bruke DSpace bash import scriptet. I tillegg lages content fila på server. (Fil som består av alle filnavn som skal importerer inn i DSpace)

---

20/03-2012

Begge 6t:

Som igår har vi satt oss sammen. Litt par-programmering, i tillegg til at vi enkelt kan få feedback på hva vi har gjort.

Det er også veldig lett å spørre hverandre som råd da vi har forskjellige ferdigheter. F.eks Ola kan JavaScript best, Ole kan PHP best.

Funksjonalitet som er blitt laget:

- Kan nå slette filer som har blitt dratt inn.

- Lista over filer oppdateres hvis filer blir adda, eller sletta.

- Metadata kan nå legges til i en dynamisk bygget form. funksjoner er laget slik at brukere etterhvert skal kunne legge til/fjerne de frivillige metadatatypene.

---

19/03-2012

Begge 8t:

Begynte på løsning 2.

Vi har mye av opplastingsgrensesnittet klart alleredet, men fortsatt er det mye arbeid som gjenstår.

Vi startet med å diskutere hva som måtte gjøres for å få dette ferdig.

Vi rakk å starte litt med kodebiten også:

- Collection valgt legges nå i en session variable

- Kan nå velge flere filer å laste opp via "Drag and Drop"

- Kan også velge filer i flere omganger

---

18/03-2012

Ole 10t:

laget ferdig funksjon for innlogging med autobygging av array som støtter ukjent antall workspaces og collections.

brukte simplexml til å lese xml fil og strippet ut kun det vi trengte til et array.

laget automatisk liste for valg av collections som med ajax og php oppdaterer en sessionvariabel med valgte collection.

Dette er laget som egen php-fil og benytter seg av sessionarray slik at listen kan implementeres overalt innad i samme session.

Laget hjelpefunksjoner for forandring av spesialkarakterer som kan utvides ved hjelp av ett før og etter array, kun Ø lagt til hittil

valgte å bruke sessionvariabler til xmlarray og "valgt collection" fordi dette er enklest å forholde seg til når vi enda ikke vet hvilken rekkefølge filene skal kalles. "valgte collection" sendes til server som sessionvariabel fordi den skal senere brukes i serverscript til importering til DSpace.

---

17/03-2012

Ola 3t:

Fikk LDAP authentifisering med hierarki modulen. Dette trengs siden brukere ligger i flere OU i AD.

I tillegg blir brukere som blir automatisk opprettet automatisk medlem i en selvbestemt gruppe som har publiseringsrettigheter som vi enkelt kan sette.

Dette konkluderer mer eller mindre DSpace setup for oppgaven. Serveren har alt vi trenger av funksjoner som vi har tenkt på så langt.

Jeg har lært veldig mye om hvordan DSpace fungerer og hvordan man feilsøker ved hjelp av prøve, feile, debugging, logger osv.

Jeg har også merket at det å sette seg inn i DSpace og hvordan det funker kunne vært en bacheloroppgave i seg selv. Jeg kan basic konfigurasjon og oppsett, men det er så mye mye mer.

---

16/03-2012

Ola 5t:

Fikk til enkel LDAP authentifisering av brukere. Brukere får nå automatisk opprettet en bruker hvis de logger inn med en gyldig AD bruker.

Ole 4t:

Vi bestemte oss for å kutte ut bruk av hjelpefunksjoner og XML klassene til sword og heller implementere en egen løsning som bygger et tredimensjonalt array av xml filen med kun den infoen vi trenger.

Begynte denne prosessen

---

15/03-2012

Ola 5t:

Begynte jobben med å få DSpace til å bruke LDAP til å authentifisere brukere.

Ole 9t:

Lagde cURL kode for henting av servicedokument og brukte dette til innlogging.

Innlogging foregår nå ved at vi requester et servicedokument fra DSpace. dette inneholder informasjon om hvor brukeren har lov til å publisere, hvis brukeren finnes i DSpace.

Om et servicedokument returneres er brukeren autentisert med rettighetene listet i dokumentet.

DSpace skal hente denne informasjonen fra LDAP server.

---

14/03-2012

Begge 2t:

Møte med veileder. Se møtereferat for oppsumering.

---

13/03-2012

Ola 5t:

DSpace fungerer nå slik som det skal i produksjon. Tenker da på at brukere kan registrere seg og motta epost fra server.

Har brukt Gmail istedenfor en lokal server pga domene vi tester med bare fungerer lokalt på nettverket.

Nå gjenstår bare LDAP support slik at brukere kan logge seg på via sin feide bruker og automatisk få en DSpace bruker ved første innlogging.

---

12/03-2012

Ola 3t:

Begynte prosessen med å få epost modulen i DSpace til å fungere.

---

11/03-2012

Ola 5t:

DSpace testing og leser mer om DSpace import via "Simple Archive Format"

---

10/03-2012

Ole 5t:

Tatt noen avgjørelser på hvordan implementeringen skal foregå (rekkefølge av funkjsoner i tråd med sword), Ola var enig.

Laget en arbeidsskisse av "use case" diagram. spesifisert to "use case" og er klare

for å begynne å implementere de.

Ola 5t:

Kom over ett alternativ til sword. DSpace kan importere filer via ett bashscript fra mapper en enn viss struktur.

Vi vurderer nå dette opp mot sword. Dette blir en løsning der vi kan bruke det vi har tenkt i planleggingen mer samtidlig som det blir en løsning vi lager alt på selv.

Istedet for å bruke sword sitt api for det meste.

---

08/03-2012

Ole 4t:

Mere Sword studering. metadata per fil kan ikke endres direkte via sword og må enten endres med JavaScript før opplasting eller med egnet verktøy på server før publisering.
Vi må vurdere hvorvidt det er nødvendig å endre metadata per fil eller om det holder per publiserte pakke
Tegnet forslag til flytskjema for hele løsningen og begynte på praktisk design av skjermbilder(webgrensesnitt)

Ola 4t:

DSpace og sword lesing/testing.

---

07/03-2012

Ola 5t:

DSpace og sword lesing/testing.

Ole 4t:

Studering av sword php med eksempler. Det ser ut som vi enkelt kan implementere sword med stream-metoden vi nå bruker til opplastning. Sword php behandler filene etter at de er lastet opp til server, så dette kan enkelt endres senere. Autentisering gjøres direkte mot DSpace via sword.

---

06/03-2012

Ola 5t:

Satte opp DSpace på nytt på kjelsrud.org. Skal også sette opp epostserver for å teste brukeraktivering osv.

---

02/03-2012

Ola 4t:

Leser DSpace og sword dokumentasjon.

Fikk oppdatert nettsiden med statusrapport og send dette til veildeder.

---

28/02-2012

Ola 5t:

DSpace fungerer. Vi kan lage collections og laste opp filer til de med det tungvidte DSpace interfacet.

Vi fikk også testet sword. Dette fungerer utmerket for opplasting av filer.

Vi har derfor mer eller mindre bestemt oss for at vi bruker sword til opplasting av filer inn til DSpace.

Big breakthrought ettersom dette har vært ett av de større spørsmålene vi har hatt i løpet av prosjektet sålangt.

Dagene fremover vil bli brukt til å komme mer inn i hvordan DSpace fungere + lese seg opp på sword.

Hvordan det  fungerer + hvordan vi utvikler en sword client for å håndtere opplasting.

Ole 5t:

Lest og satt opp en swordapp (easydeposit) med komponenter for sword php api for opplasting mot

dspaceserver. får koblet til og listet ut rettigheter fra lokal dspaceserver.

---

27/02-2012

Ola 6t:

Hurra! Fikk installert DSpace etter mye krøll med forskjellige versjoner av de programmene som kreves.

Ole 5t:

Lest og satt opp en swordapp (easydeposit) med komponenter for sword php api for opplasting mot

dspaceserver. får koblet til og listet ut rettigheter fra lokal dspaceserver.

---

26/02-2012

Ole 4t:

Lest og testet php i forbindelse med filbehandling lokalt og php generelt.

---

25/02-2012

Ola 4t:

Satt opp vmware vspare server.

Denne brukes til å få isntallert DSpace + andre vm maskiner for å teste ting.

Begynte også å prøve å få installert DSpace. Lettere med vmware siden jeg kan bare rulle tilbake hvis det skjer noe galt.

---

23/02-2012

Ola 5t:

Begynte prosessen med å insatallere DSpace.

DSpace krever flere andre programmer før det kan installeres som:

Java, apache-maven, apache-ant ++

---

21/02-2012

ola: 4t

Fant noe som heter sword. Client til DSpace for å laste inn filer.

Leste meg opp på DSpace.

Ole 4t:

Reasearch på sword for DSpace publisering. og fundert på om sword kan batch behandle

filer etter opplastning eller om sword kan støtte resume direkte, eventuelt modifiseres til det.

---

15/02-2012

Ola 4t:

---

14/02-2012

Ole 4t:

LDAP php authentication(research), sett på skolens LDAP server public, og forsøkt å skaffe tilgang til

autentisering fra testserver via vpn.

DSpace; lesing av manual

---

13/02-2012

Ola 8t.

Inkludert litt av og på i helgen.

Litt frustrerende fremgang nå. Blir mye lesing uten resultat.

Fått litt oversikt over hvordan DSpace fungerer.

Problemet her blir hvordan vi skal få lastet inn filene i DSpace.

Skal vi bruke vår egenlagde opplastingsgrensesnitt som fungerer + filoverføring på server.

Eller må vi legge til en egen/Modifisere DSpace's egen opplastingsgrensesnitt.

Ole 1t.

LDAP php authentication

---

10/02-2012

Ole 3t.

Ajax research

---

08/02-2012

Ola 7t.

DSpace research.

---

06/02-2012

Ola: 6t.

Ole 7t.

Innlogging https pluss research apache, ssl php

---

05/02-2012

Ole 3t.

Innlogging http mot mysql

---

03/02-2012

Ola:2t.

Kommet frem til at jeg ikke kan bruke FormData() objektet i JavaScript hvis jeg bruker php://input.

For å sende info til php kan jeg enten bruke XMLHttpRequest.SetRequestHeader() eller sende info seperat fra filen.

Ole: 2t.

Reasearch JavaScript  XMLHttpRequest.

---

02/02-2012

Begge: 1t.

Møte med veileder.

Ola: 4t.

Fikk på møtet bekreftet at jeg var på riktig vei med opplastingsbiten.

Beslutninger tatt pga møtet:

-Bruke PHP til server side. (Foretrukket fremfor perl)

-Bruke php://input stream for opplasting pga "pause/resume" funksjonalitet

-I JavaScript bruke XMLHttpRequest for å sende filen til php via POST.

---

01/02-2012

Ola: 8t.

Mye research og testing av XMLHttpRequest objektet + php for opplasting til server.

Ole: 4t.

XMLHttpRequest2 noe testing

---

31/01-2012

Ola: 8t.

Proof of concept side oppe. Drag and drop opplasting.

Ole: 5t.

mer fileapi eksempler og testing

---

30/01-2012

Ola: 7t.

Satt opp test side med drag and drop funksjonalitet. Filer kan droppes inn og browser

leser filinformasjon via JavaScript. Testet mye forskjellig funksjonalitet for å se hva som passer oss best.

Ole: 4t.

Research på JavaScript fileapi (filereader, file ...), drag and drop events

---

29/01-2012

Ole: 7t.

Research og testing. CSS3 tutorials CSS3 utprøving  blant annet animation og trasform.

Jquery og fileapi.

Ola: 4t.

JavaScript tutorials og testing.

---

25/01-2012

Ole: 3t.

Research og testing. JavaScript tutorials.

Ola: 2t.

Levering av forprosjekt. Research og testing.

---

24/01-2012

Ole: 3t.

Reasearch, oppsett testserver.

Ola: 3t.

Research og testing. JavaScript tutorials. Funksjonalitet og enkle eksempler.

---

23/01-2012

Begge: 2t.

Forprosjekt. Finpuss av rapport. RC-1 ferdig og levert til veileder for eventuelle innspill før levering.

---

21/01-2012

Ola: 3t.

Forprosjekt. Omfang, avgrensning og planlegging.

Ole: 2t

Risikoanalyse.

---

20/01-2012

Ola: 3t.

Forprosjekt. Organisering, mål og rammer.

---

19/01-2012

Begge:

Møte med Øivind Kolloen.

Fikk bekreftet at headerinfo skal la seg hente med JavaScript.

La fram våre prioriteringer og fikk tips om hva vi burde jobbe med først.

Skal levere utkast av forprosjekt til Øivind på tirsdag for tilbakemeldinger før innleveringen fredag.

---

18/01-2012

Begge:

Møte med Kjell Are Refsvik. Kjell Are var enig i fremmgangsmåten og prioriteringene.

Kom med innspill om tilbakeføring av metadata til fil før opplastning,

Ola: Forprosjekt. 3t.

Ole: Forprosjekt 4t.

---

17/01-2012

Begge: Forprosjekt 4t.

Diskuterte prioriterte arbeidsoppgaver, grovinndeling av prosjektet

Beslutninger:

Starter med en web basert html5 løsning.

Valgte utviklingsmodell. XP som rammeverk, men med visse modifikasjoner.

---

13/01-2012

Ola: Forprosjekt. 2t.

---

12/01-2012

Ola: Forprosjekt. 2t.

---

10/01-2012

Veiledermøte: Diskutert viktighet av kompabilitet, alternative løsninger og videre fremgang.

Lynkurs: Forprosjekt med Tom Røise.

---

09/01-2012

Oppdatering av hjemmeside, fotosesjon og planlegging.

---

07/01-2012

Konfigurering og oppdatering av hjemmeside

---

06/01-2012

Oppsett av hjemmeside.

---

04/01-2012

Oppstartsmøte.

Planlegging om hva som skal gjøres frem mot første innlevering.

Forespørsel om møte med veileder m.m.

## Vedlegg G - Dublin Core metadata

# Liste over mulige metadata ved DSpace import

| | | |
|---|---|---|
| contributor | | A person, organization, or service responsible for the content of the resource. Catch-all for unspecified contributors. |
| contributor | advisor | Use primarily for thesis advisor. |
| contributor | author | |
| contributor | editor | |
| contributor | illustrator | |
| contributor | other | |
| coverage | spatial | Spatial characteristics of content. |
| coverage | temporal | Temporal characteristics of content. |
| creator | | Do not use; only for harvested metadata. |
| date | | Use qualified form if possible. |
| date | accessioned | Date DSpace takes possession of item. |
| date | available | Date or date range item became available to the public. |
| date | copyright | Date of copyright. |
| date | created | Date of creation or manufacture of intellectual content if different from date.issued. |
| date | issued | Date of publication or distribution. |
| date | submitted | Recommend for theses/dissertations. |
| identifier | | Catch-all for unambiguous identifiers not defined by qualified form; use identifier.other for a known identifier common to a local collection instead of unqualified form. |

| | | |
|---|---|---|
| identifier | citation | Human-readable, standard bibliographic citation of non-DSpace format of this item |
| identifier | govdoc | A government document number |
| identifier | isbn | International Standard Book Number |
| identifier | issn | International Standard Serial Number |
| identifier | sici | Serial Item and Contribution Identifier |
| identifier | ismn | International Standard Music Number |
| identifier | other | A known identifier type common to a local collection. |
| identifier | uri | Uniform Resource Identifier |
| description | | Catch-all for any description not defined by qualifiers. |
| description | abstract | Abstract or summary. |
| description | provenance | The history of custody of the item since its creation, including any changes successive custodians made to it. |
| description | sponsorship | Information about sponsoring agencies, individuals, or contractual arrangements for the item. |
| description | statementofresponsibility | To preserve statement of responsibility from MARC records. |
| description | tableofcontents | A table of contents for a given item. |
| description | uri | Uniform Resource Identifier pointing to description of this item. |
| format | | Catch-all for any format information not defined by qualifiers. |
| format | extent | Size or duration. |
| format | medium | Physical medium. |

| | | |
|---|---|---|
| format | mimetype | Registered MIME type identifiers. |
| language | | Catch-all for non-ISO forms of the language of the item, accommodating harvested values. |
| language | iso | Current ISO standard for language of intellectual content, including country codes (e.g. "en_US"). |
| publisher | | Entity responsible for publication, distribution, or imprint. |
| relation | | Catch-all for references to other related items. |
| relation | isformatof | References additional physical form. |
| relation | ispartof | References physically or logically containing item. |
| relation | ispartofseries | Series name and number within that series, if available. |
| relation | haspart | References physically or logically contained item. |
| relation | isversionof | References earlier version. |
| relation | hasversion | References later version. |
| relation | isbasedon | References source. |
| relation | isreferencedby | Pointed to by referenced resource. |
| relation | requires | Referenced resource is required to support function, delivery, or coherence of item. |
| relation | replaces | References preceeding item. |
| relation | isreplacedby | References succeeding item. |
| relation | uri | References Uniform Resource Identifier for related item |
| rights | | Terms governing use and reproduction. |
| rights | uri | References terms governing use and reproduction. |
| source | | Do not use; only for harvested metadata. |

| | | |
|---|---|---|
| source | uri | Do not use; only for harvested metadata. |
| subject | | Uncontrolled index term. |
| subject | classification | Catch-all for value from local classification system. Global classification systems will receive specific qualifier |
| subject | ddc | Dewey Decimal Classification Number |
| subject | lcc | Library of Congress Classification Number |
| subject | lcsh | Library of Congress Subject Headings |
| subject | mesh | MEdical Subject Headings |
| subject | other | Local controlled vocabulary; global vocabularies will receive specific qualifier. |
| title | | Title statement/title proper. |
| title | alternative | Varying (or substitute) form of title proper appearing in item, e.g. abbreviation or translation |
| type | | Nature or genre of content. |

# Liste over mulige mimetyper ved DSpace import

| Mimetype | Short Description | Description | Support Level | Internal | Extensions |
|---|---|---|---|---|---|
| application/octet-stream | Unknown | Unknown data format | Unknown | false | |
| text/plain | License | Item-specific license agreed upon to submission | Known | true | |
| application/marc | MARC | Machine-Readable Cataloging records | Known | false | |
| application/mathematica | Mathematica | Mathematica Notebook | Known | false | ma |
| application/msword | Microsoft Word | Microsoft Word | Known | false | doc |
| application/pdf | Adobe PDF | Adobe Portable Document Format | Known | false | pdf |
| application/postscript | Postscript | Postscript Files | Known | false | ai, eps, ps |
| application/sgml | SGML | SGML application (RFC 1874) | Known | false | sgm, sgml |
| application/vnd.ms-excel | Microsoft Excel | Microsoft Excel | Known | false | xls |
| application/vnd.ms- | Microsoft | Microsoft | Known | false | ppt |

| | | | | | |
|---|---|---|---|---|---|
| powerpoint | Powerpoint | Powerpoint | | | |
| application/vnd.ms-project | Microsoft Project | Microsoft Project | Known | false | mpd, mpp, mpx |
| application/vnd.visio | Microsoft Visio | Microsoft Visio | Known | false | vsd |
| application/wordperfect5.1 | WordPerfect | WordPerfect 5.1 document | Known | false | wpd |
| application/x-dvi | TeX dvi | TeX dvi format | Known | false | dvi |
| application/x-filemaker | FMP3 | Filemaker Pro | Known | false | fm |
| application/x-latex | LateX | LaTeX document | Known | false | latex |
| application/x-photoshop | Photoshop | Photoshop | Known | false | pdd, psd |
| application/x-tex | TeX | Tex/LateX document | Known | false | tex |
| audio/basic | audio/basic | Basic Audio | Known | false | au, snd |
| audio/x-aiff | AIFF | Audio Interchange File Format | Known | false | aif, aifc, aiff |
| audio/x-mpeg | MPEG Audio | MPEG Audio | Known | false | abs, mpa, mpega |
| audio/x-pn-realaudio | RealAudio | RealAudio file | Known | false | ra, ram |
| audio/x-wav | WAV | Broadcase Wave Format | Known | false | wav |
| image/gif | GIF | Graphics Interchange Format | Known | false | gif |
| image/jpeg | JPEG | Joint | Known | false | jpeg, jpg |

| | | | | | |
|---|---|---|---|---|---|
| | | Photographic Experts Group/JPEG File Interchange Format (JFIF) | | | |
| image/png | image/png | Portable Network Graphics | Known | false | png |
| image/tiff | TIFF | Tag Image File Format | Known | false | tif, tiff |
| image/x-ms-bmp | BMP | Microsoft Windows bitmap | Known | false | bmp |
| image/x-photo-cd | Photo CD | Kodak Photo CD image | Known | false | pcd |
| text/css | CSS | Cascading Style Sheets | Known | false | css |
| text/html | HTML | Hypertext Markup Language | Known | false | htm, html |
| text/plain | Text | Plain Text | Known | false | asc, txt |
| text/richtext | RTF | Rich Text Format | Known | false | rtf |
| text/xml | XML | Extensible Markup Language | Known | false | xml |
| video/mpeg | MPEG | Moving Picture Experts Group | Known | false | mpe, mpeg, mpg |

| | | | | | |
|---|---|---|---|---|---|
| video/quicktime | Video Quicktime | Video Quicktime | Known | false | mov, qt |