

IT-DIPLOMUDDANNELSEN

AFGANGSPROJEKT

AF
JAN SCHRØDER HANSEN

EFTERÅR/VINTER 2011

INDHOLD

1.	Indledning.....	3
2.	Opgavebeskrivelse.....	3
3.	Krav	5
3.1.	Funktionelle krav	5
3.2.	Ikke funktionelle krav	7
4.	Analyse	8
4.1.	Administrative use cases.....	8
4.2.	Primære use cases	12
4.3.	Krav/use case matrix.....	22
4.4.	Analysediagram.....	23
5.	Design.....	25
5.1.	Webside design	31
5.2.	Frameworks	37
5.3.	Sikkerhed.....	38
6.	Idriftsættelse	39
7.	UP Iterationer	40
8.	Konklusion	41
9.	Bilag	42
9.1.	Danske/engelske termer.....	42
9.2.	Brugervejledning.....	43
9.3.	Udviklingsmiljø.....	54
9.4.	Kode	55
9.5.	Indhold på den vedlagte cd.....	116
10.	Noter.....	117

1. INDLEDNING

Dette afgangsprøveprojekt er lavet i forbindelse med IT-Diplomuddannelsen på Ingeniørhøjskolen i København.

2. OPGAVEBESKRIVELSE

Da min samlever er tegnsprogstolk, og jeg igennem hende ved at der ikke findes bruger-drevne danske tegnsprogsordbøger tilgængelig på internettet, vil jeg udvikle en. Der er en officiel ordbog, men denne vedligeholdes fra central side.

Jeg har fundet følgende to sider på internettet for dansk tegnsprog:

Side	Beskrivelse
www.tegnsprog.dk	Den officielle danske tegnsprogsordbog. Indeholder pt. ca. 2.000 ord.
www.streetsigners.dk	Siden for de unge, med mange "hippe" tegn.

For begge sider gælder, at de vedligeholdes af personer bag siderne. Der er ikke mulighed for, at brugerne selv kan komme med indhold til disse sider.

Igennem min samlever ved jeg, at tolke og dæve tit mangler et sted hvor de kan spørge til ord og begreber, som er oppe i tiden. Et eksempel kunne være "Det arabiske forår", hvordan siger man "Egypten" og "Libyen" etc. på tegnsprog.

Det skal først og fremmest være en web-løsning, senere kunne den udvides til smartphones.

Alle skal kunne søge efter ord på siden, men for at lave forespørgsler på et ord, eller uploade video forslag til et ord, skal man være en kendt bruger af systemet. Ord skal kunne grupperes, f.eks. kan der laves en gruppe der hedder "Det arabiske forår", eller "Lande i Nordafrika".

Man skal kunne oprette sig som bruger vha. fuldt navn og e-mail. Før man kan logge på første gang, skal e-mailen verificeres.

Søgningen skal kunne være på ord eller grupper. Så skal man kunne tilmelde sig en e-mail notifikation på et eller flere ord. Således at man får en e-mail, når der sker noget på de ord man interesserer sig for.

Løsningens navn bliver "Tegn til tiden".

Teknik: Java web løsning, vha. Java¹, Apache Wicket², Hibernate³, JavaDB⁴ og Apache Tomcat⁵.

Da jeg altid skriver på engelsk i min kode, har jeg valgt at mine UML⁶ diagrammer også er på engelsk. Men da rapporten her er på dansk, har jeg vedlagt en dansk/engelsk ordliste under bilag. Se afsnit 9.1. Dette gælder dog ikke for use case⁷ diagrammer og use cases. Som jo er det UML værktøj, som kan bruges overfor mennesker, som ikke arbejder med it udvikling til dagligt.

Jeg vil arbejde efter UP⁸ (Unified Process), som er en iterativ udviklingsproces. Men denne rapport vil følge den gamle "Vandfalds model", da det giver et naturligt flow i beskrivelsen af systemet.

Alle diagrammer er udarbejdet vha. af programmer MagicDraw⁹.

I de følgende afsnit gennemgås de forskellige udviklingsfaser, startende med krav.

3. KRAV

Følgende to afsnit indeholder lister med krav til systemet, opdelt efter funktionelle og ikke funktionelle krav. De enkelte krav prioriteres efter MoSCoW (Must have, Should Have, Could Have, Wants to have).

3.1. FUNKTIONELLE KRAV

ID	Krav	Prioritet
A1	Systemet skal være tilgængeligt for alle ved søgninger efter ord.	M
A2	Systemet skal kræve at man er logget på for at uploade videofiler, slette videofiler, kommentere og bedømme videofiler, samt for at deltage i diskussioner.	M
A3	Systemet skal kende en brugers fulde navn og en gyldig e-mail adresse. Dvs. at nye e-mail adresser skal verificeres.	M
A4	Systemet skal give mulighed for at bedømme videosekvenser med 1 til 5 stjerner. En bruger kan kun give en bedømmelse pr. video. Men kan ændre eller slette sin bedømmelse senere.	S
A5	Systemet skal give mulighed for at kommentere videosekvenser.	S
A6	Systemet skal kunne give mulighed for at anmelde videosekvenser, som er anstødelige. Dette skal medføre en e-mail til en systemansvarlig brugere.	S
A7	Systemet skal kunne håndtere to slags indloggede brugere. Admin (superuser) og almindelig. Se efterfølgende rollematrix.	M
A8	Systemet skal kunne generere en ny adgangskode, hvis en bruger har glemt sin adgangskode. Brugeren skal kunne huske den e-mail som er i systemet.	M
A9	Systemet skal give mulighed for at ændre navn, adgangskode og e-mail	M
A10	Systemet skal ikke kunne oprette systembrugere. Dette skal gøres direkte i database.	M
A11	Systemet skal give mulighed for at diskutere ord.	S
A11	Systemet skal give mulighed for at gruppere ord sammen.	M
A12	Systemet skal give mulighed for at søge efter ord og efter ordgrupper	M
A13	Systemet skal give brugerne et hurtigt overblik over egne ord/forespørgsler, egne ordgrupper og egne uploads af videosekvenser.	M
A14	Systemet skal give brugerne et hurtigt overblik over ord/forespørgsler som mangler forslag, samt hvilke ord der indgår i en ordgruppe.	M

Rollematrix for krav A7

	Ikke logget ind	Alm.	System
Søge	Ja	Ja	Ja
Uploade filer	Nej	Ja	Ja
Slette uploads	Nej	Ja – Kun egne	Ja
Bedømme og kommentere uploads	Nej	Ja – Ikke sin egen	Ja
Diskutere ord	Nej	Ja	Ja
Anmelde som anstødelig	Nej	Ja	Ja
Forespørge på ord	Nej	Ja	Ja
Oprette nye grupper	Nej	Ja	Ja
Slette grupper og uploads som kun er knyttet til denne bruger	Nej	Ja	Ja
Slette/rette grupper, ord og uploads for alle brugere	Nej	Nej	Ja

3.2. IKKE FUNKTIONELLE KRAV

ID	Krav	Prioritet
B1	Systemet skal være en WEB-Løsning.	M
B2	Systemet skal kunne køre på en Apache Tomcat version 7 eller nyere webserver.	M
B3	Systemet skal benytte Java version 1.6 eller nyere.	
B4	Systemet skal benytte følgende 2 Java frameworks: <ul style="list-style-type: none"> • Apache Wicket – version 1.5 – som web framework • Hibernate - version 4.0 – framework som bygger bro mellem den objektorienteret verden og den rationelle database verden 	M
B5	Systemet skal persistere data i en JavaDB version 10 eller nyere, som er en del af standard Java.	M
B6	Systemet skal kunne benyttes sammen med HTTPS ²⁹ . Som minimum under login, opret ny bruger og ret bruger.	M
B7	Systemet skal benytte optimistisk låsning, vha. Hibernate.	M
B8	De mest gængse videoformater skal kunne uploades, og transformeres til videoformater som understøtter HTML5's videotag. Pt. er det OGG formatet.	M
B9	FFMPEG ¹⁵ benyttes til konvertering af videofiler.	M

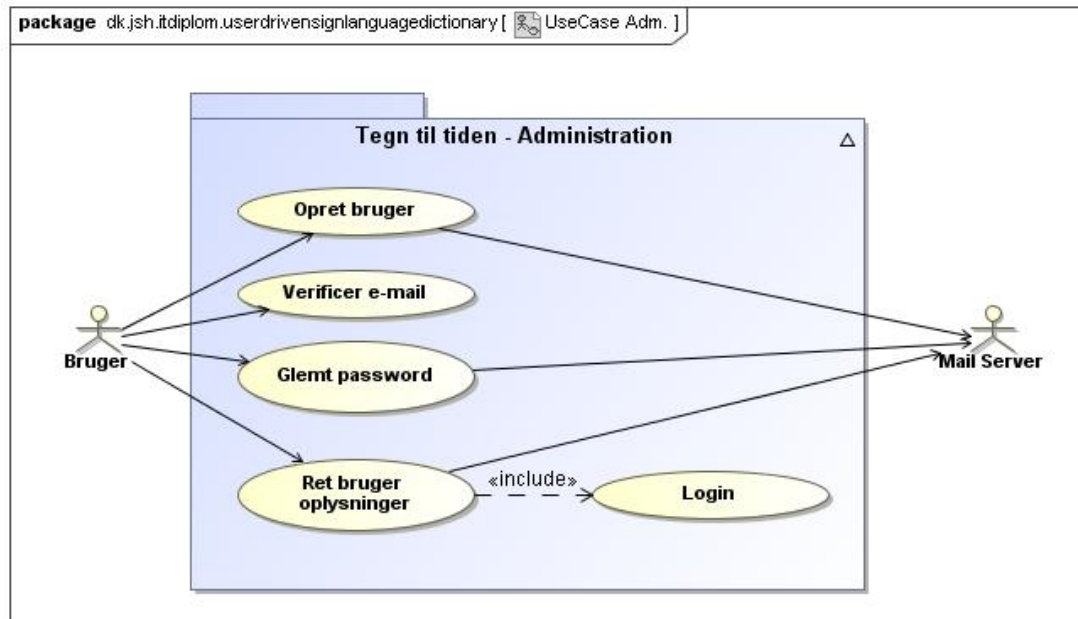
Det afslutter krav indsamlingen, og det bringer os videre til analysefasen.

4. ANALYSE

Jeg har valgt at lave to use case diagrammer, selvom der kun er et system. Diagrammerne er opdelt efter administrative use cases og use cases som fortæller hvad systemets primære opgave bliver.

Følgende use case diagram er de administrative use cases. Det er use cases som omhandler brugeroprettelse m.m.

4.1. ADMINISTRATIVE USE CASES



Figur 1 - Administration use case diagram

På de følgende sider er ovenstående use cases beskrevet.

Use case: Login
ID: UC-1
Kort beskrivelse: En bruger logges på systemet.
Primære aktører: Alle kendte brugere.
Sekundære aktører: Ingen.
Start betingelser: Brugeren er kendt af systemet.
Hovedforløb: <ol style="list-style-type: none"> 1. Brugeren udfylder brugerkode og password felterne og trykker på "Login" knappen. 2. Systemet verificerer brugerkode og password. Hvis brugerkode og password er i orden, logges brugeren på systemet, og der er nu en mulighed for at logge af igen.
Slut betingelser: Brugeren er logget på systemet. Dette giver adgang til upload af ord, forespørgelser på ord m.m.
Alternative forløb: <ul style="list-style-type: none"> - Brugeren kan ikke logges ind, da brugerkode og/eller password ikke kendes af systemet. Systemet melder at enten brugerkode og/eller password er ukendt for systemet. Og henviser til "Glemt password" og "Opret bruger" mulighederne.

Use case: Opret bruger
ID: UC-2
Kort beskrivelse: En ny bruger oprettes.
Primære aktører: Alle.
Sekundære aktører: E-mail server.
Start betingelser: Brugeren skal have en valid e-mail adresse.
Hovedforløb: <ol style="list-style-type: none"> 1. En kommende bruger vælger "Ny bruger" linket 2. Systemet viser en ny side, som giver mulighed for indtastning af navn, brugerkode, e-mail, password og gentag password. 3. Brugeren udfylder felter og trykker på "Gem" knappen. 4. Systemet verificerer alle felterne: <ol style="list-style-type: none"> 4.1. Navn skal udfyldes. 4.2. Brugerkode skal udfyldes 4.3. E-mail skal udfyldes med en valid e-mail adresse. Dvs. skal indeholde @ m.m. 4.4. Begge password felter skal være udfyldt med samme tekst. Og teksten skal være mindst 8 tegn lang og indeholde både bogstaver og tal. 4.5. Hvis alle felter er valide, gemmer systemet navn, e-mail og password. Og sender en e-mail til den nye bruger. E-mailen indeholder et link til systemet som skal aktiveres før brugeren kan logge på systemet. Dette gøres for at sikre at e-mail adressen er valid.
Slut betingelser: Bruger er delvist oprettet i systemet. Men kan først logge på systemet, når brugers e-mail er verificeret.
Alternative forløb: <ul style="list-style-type: none"> - Brugeren fortryder.

Use case: Verificer e-mail
ID: UC-3
Kort beskrivelse: En brugers mail er verificeret.
Primære aktører: Alle kendte brugere.
Sekundære aktører: Ingen.
Start betingelser: En bruger som er oprettet i systemet, og brugeren har modtaget en e-mail verifikations e-mail fra systemet.
Hovedforløb: <ol style="list-style-type: none"> 1. Brugeren åbner e-mailen fra systemet og trykker på linket i denne. 2. Systemet viser en side, som fortæller at brugerens e-mail nu er verificeret. 3. Systemet gemmer oplysninger om verificering af brugers e-mail. 4. Brugeren kan nu fortsætte med at bruge systemet, som en bruger der er logget på systemet.
Slut betingelser: Ingen.

Use case: Glemt password
ID: UC-4
Kort beskrivelse: Bestilling af nyt midlertidigt password.
Primære aktører: Alle kendte brugere.
Sekundære aktører: E-mail server.
Start betingelser: En bruger som er oprettet i systemet med en valid e-mail.
Hovedforløb: <ol style="list-style-type: none"> 1. Brugeren trykker på linket "Glemt password" på login siden. 2. Systemet viser en side, hvor der skal indtastes den e-mail, som er kendt af systemet for den aktuelle bruger. 3. Brugeren indtaster sin e-mail adresse og trykker på knappen "Dan nyt password" 4. Systemet generer et nyt password til brugeren, og sender dette med en e-mail til denne. I e-mailen skal der være en opfordring til at ændre passwordet.
Slut betingelser: Ingen.
Alternative forløb: <ul style="list-style-type: none"> - Brugeren fortryder.

Use case: Ret brugeroplysninger
ID: UC-5
Kort beskrivelse: Rette brugeroplysninger som navn og e-mail.
Primære aktører: Alle kendte brugere.
Sekundære aktører: E-mail server.
Start betingelser: En bruger som er oprettet i systemet.
Hovedforløb: <ol style="list-style-type: none"> 1. include (Login) 2. Brugeren trykker på linket "Ret brugeroplysninger" linket. 3. Systemet viser en side med følgende felter: Navn, brugerkode, e-mail, password og gentag password. Navn, brugerkode og e-mail er forudfyldt. Brugerkode kan ikke rettes. 4. Brugeren indtaster sine rettelser. 5. Systemet gemmer rettelserne, hvis e-mail adressen er rettet, sender systemet en e-mail verifikations mail.
Slut betingelser: Ingen.
Alternative forløb: <ul style="list-style-type: none"> - Brugeren fortryder.

Følgende use case diagram og use case beskrivelser er systemets primære use cases.



12

Use case: Søg
ID: UC-6
Kort beskrivelse: Søg efter et ord.
Primære aktører: Alle.
Sekundære aktører: Ingen.
Start betingelser: Ingen.
Hovedforløb: <ol style="list-style-type: none"> 1. Brugeren navigerer til systemets startside. 2. Systemets startside indeholder et søgefelt samt et dropdown felt, hvor der vælges mellem ord eller gruppe. Ord er valgt som standard. I søgefeltet kan tegn udskiftes med wildcards * for et eller flere tegn og ? for et tegn. 3. Brugeren udfylder søgefeltet. 4. Hvis ord er valgt i dropdown listen <ol style="list-style-type: none"> 4.1. Søger systemet efter de ord, som opfylder søgningen. 5. Hvis gruppe er valgt i dropdown listen <ol style="list-style-type: none"> 5.1. Søger systemet efter alle de ord, der indgår i grupper, som opfylder søgningen. 6. Systemet viser en liste med links til de enkelte ord, sorteret alfabetisk. For hvert ord vises dato for oprettelse af ordet, samt de ordgrupper som ordet indgår i. Selve ordet i listen er et link til en side med detaljer om ordet, samt forslag til ordet.
Slut betingelser: Ingen.
Alternative forløb: <ul style="list-style-type: none"> - Brugeren fortryder.

Use case: Vis alle forespørgsler
ID: UC-7
Kort beskrivelse: Viser oversigt over alle ord som mangler forslag/uploads.
Primære aktører: Alle.
Sekundære aktører: Ingen.
Start betingelser: Ingen.
Hovedforløb: <ol style="list-style-type: none"> 1. Brugeren vælger menu linket "Alle forespørgsler". 2. System viser en liste med ord, som mangler forslag. Sorteret alfabetisk. For hvert ord vises hvornår det er oprettet, samt hvilke ordgrupper det tilhører. Selve ordet er et link til en side med detaljer om ordet. Derudover er der et link til en uploadside, for det aktuelle ord (dog kun for brugere som er indlogget, ellers vises linket som ikke aktivt.).
Slut betingelser: Ingen.
Alternative forløb:

Use case: Vis alle Grupper
ID: UC-8
Kort beskrivelse: Viser oversigt over alle ordgrupper.
Primære aktører: Alle.
Sekundære aktører: Ingen.
Start betingelser: Ingen.
Hovedforløb: <ol style="list-style-type: none"> 1. Brugeren vælger menu linket "Alle grupper". 2. System viser en liste med grupper, sorteret alfabetisk. For hver gruppe vises hvornår den er oprettet. Selve gruppenavnet er et link til en side med detaljer om gruppen, samt en liste over de ord som indgår i gruppen.
Slut betingelser: Ingen.
Alternative forløb:

Use case: Vis egne forespørgsler
ID: UC-9
Kort beskrivelse: Viser oversigt over en brugers egne forespørgsler.
Primære aktører: Alle kendte brugere.
Sekundære aktører: Ingen.
Start betingelser: Ingen.
Hovedforløb: <ol style="list-style-type: none"> 1. include (Login) 2. Brugeren vælger menu linket "Mine forespørgsler". 3. System viser en liste med ord, som brugeren selv har ønsket forslag til, sorteret alfabetisk. For hvert ord vises hvornår det er oprettet, samt hvilke grupper det tilhører. Selve ordet er et link til en side med detaljer om ordet. Derudover er der et link til en ret side, for det aktuelle ord.
Slut betingelser: Ingen.
Alternative forløb:

Use case: Vis egne grupper
ID: UC-10
Kort beskrivelse: Viser oversigt over en brugers egne grupper.
Primære aktører: Alle kendte brugere.
Sekundære aktører: Ingen.
Start betingelser: Ingen.
Hovedforløb: <ol style="list-style-type: none"> 1. include (Login) 2. Brugeren vælger menu linket "Mine grupper". 3. System viser en liste med grupper, som brugeren selv har oprettet, sorteret alfabetisk. For hver gruppe vises hvornår gruppen er oprettet. Selve gruppenavnet er et link til en side med detaljer om gruppen, samt en liste over de ord som indgår i gruppen. Derudover er der et link til en ret side, for den aktuelle gruppe. (Ret siden er ikke beskrevet yderligere)
Slut betingelser: Ingen.
Alternative forløb:

Use case: Vis egne uploads
ID: UC-11
Kort beskrivelse: Viser oversigt over en brugers egne uploads.
Primære aktører: Alle kendte brugere.
Sekundære aktører: Ingen.
Start betingelser: Ingen.
Hovedforløb: <ol style="list-style-type: none"> 1. include (Login) 2. Brugeren vælger menu linket "Mine uploads". 3. System viser en liste med ord, som brugeren selv har uploadet forslag til, sorteret alfabetisk. For hvert ord vises hvornår forslaget er uploadet. Selve ordet er et link til en side med detaljer om ordet. Derudover er der et link til en ret side, for det aktuelle upload. Hvis en bruger har oploadet flere forslag til et ord, så vil ordet gå igen på listen.
Slut betingelser: Ingen.
Alternative forløb:

Use case: Opret forespørgsel
ID: UC-12
Kort beskrivelse: Oprette en forespørgsel til en et ord
Primære aktører: Alle kendte brugere.
Sekundære aktører: Ingen.
Start betingelser: Ingen.
Hovedforløb: <ol style="list-style-type: none"> 1. include (Login) 2. Brugeren vælger linket "Opret ny forespørgsel" på siden "Mine forespørgsler". 3. Systemet viser en side med følgende felter: "Ord der ønskes forslag til" samt "Beskrivelse af ordet". 4. Brugeren udfylder begge felter og trykker "Gem" eller "Gem og tilknyt grupper". 5. Hvis brugeren vælger "Gem" gemmer systemet forespørgslen, dato/tid for oprettelsen af denne, samt brugeren og returnerer til oversigtssiden. 6. Hvis brugen vælger "Gem og tilknyt grupper", gemmer systemet forespørgslen og viser en ny side, hvor den kan tilknyttes eksisterende eller nye grupper. (Denne side er ikke beskrevet yderligere)
Slut betingelser: En forespørgsel er oprettet med eller uden tilknytning til grupper.
Alternative forløb: <ul style="list-style-type: none"> - Brugeren fortryder.

Use case: ret/slet forespørgsel
ID: UC-13
Kort beskrivelse: Rette eller slette egne forespørgsler til en et ord.
Primære aktører: Alle kendte brugere.
Sekundære aktører: Ingen.
Start betingelser: Ingen.
Hovedforløb: <ol style="list-style-type: none"> 1. include (Login) 2. Brugeren vælger et "Ret" link fra listen på "Mine forespørgsler" siden. Se use case Vis egne forespørgsler. 3. System viser en side med følgende felter: "Ord der ønskes forslag til", feltet er låst for rettelser, hvis der er uploads til ordet, samt "Beskrivelse af ordet". 4. Brugeren udfylder felterne med sine rettelser og trykker "Gem", "Slet" eller "Ret grupper". Slet kan kun vælges hvis, der ikke er uploadet forslag til det aktuelle ord. 5. Hvis brugeren vælger "Gem", gemmer systemet rettelserne og returnerer til oversigtssiden. 6. Eller hvis brugeren vælger "Slet", slettes forespørgslen og der returneres til oversigtssiden. 7. Hvis brugen vælger "Ret grupper", vises en ny side, hvor der kan fjernes og tilknyttes eksisterende grupper eller oprettes nye grupper (Denne side er ikke beskrevet yderligere).
Slut betingelser: En forespørgsel er rettet/slettet. Tilknytning til grupper er evt. også rettet.
Alternative forløb: <ul style="list-style-type: none"> - Brugeren fortryder.

Use case: Upload forslag
ID: UC-14
Kort beskrivelse: Opload af et video forslag til et ord/forespørgsel.
Primære aktører: Alle kendte brugere.
Sekundære aktører: Ingen.
Start betingelser: Ingen
Hovedforløb: <ol style="list-style-type: none"> 1. include (Login) 2. Brugeren vælger et "Upload forslag" link fra "Alle forespørgsler" sideoversigten eller linket "Upload forslag" fra "Ord" siden, som vises hvis der trykkes på et ord fra søgeoversigten m.fl. 3. Systemet viser en side, hvor der kan vælges en videofil ved at trykke på en "Browse.." knap. Filnavnet vises derefter i et låst felt. Derudover er der et beskrivelsesfelt, som bruges til at beskrive indholdet på videoen. 4. Brugeren vælger en fil, indtaster en beskrivelse og trykker på knappen "Upload og gem" 5. Systemet konvertere filen, til et HTML5 format og gemmer filen, beskrivelsen, dato/tid for upload og bruger. Derefter returner systemet til oversigten brugeren kom fra.
Slut betingelser: En videofil er uploadet som et forslag til et ord/forespørgsel.
Alternative forløb: <ul style="list-style-type: none"> - Brugeren fortryder. - Filen kan ikke konverteres, brugeren får besked om dette.

Use case: ret/slet upload
ID: UC-15
Kort beskrivelse: Rette eller slette egne videofil uploads til en et ord.
Primære aktører: Alle kendte brugere.
Sekundære aktører: Ingen.
Start betingelser: Ingen.
Hovedforløb: <ol style="list-style-type: none"> 1. include (Login) 2. Brugeren vælger et "Ret" link fra listen på "Mine uploads" siden. Se use case "Vis egne uploads". 3. System viser en side med selve videoen, samt beskrivelsesfeltet til denne. 4. Brugeren kan vælge at rette feltet og trykke "Gem" eller "Slet". 5. Hvis brugeren vælger "Gem", gemmer systemet rettelsen og returnerer til oversigtssiden. 6. Eller hvis brugeren vælger "Slet", slettes videofil og der returneres til oversigtssiden.
Slut betingelser: En uploadet videobeskrivelse er rettet eller videofil og beskrivelsen er slettet.
Alternative forløb: <ul style="list-style-type: none"> - Brugeren fortryder.

Use case: Tilføj dissussionsindlæg
ID: UC-16
Kort beskrivelse: Tilføj et diskussionsindlæg til et ord. For at give mulighed for at diskutere den "rette" måde at gengive ordet på tegnsprog.
Primære aktører: Alle kendte brugere.
Sekundære aktører: Ingen.
Start betingelser: Ingen.
Hovedforløb: <ol style="list-style-type: none"> 1. include (Login) 2. Brugeren er på siden for et valgt ord. Enten via en søgning (se use casen Søg) eller via andre sider som "Alle forespørgsler" eller "Mine forespørgsler". 3. Brugeren udfylder et "Tilføj diskussionsindlæg" felt og trykker på en "Gem" knap. 4. Systemet gemmer diskussionsindlægget, dato/tid for indlægget og brugeren som har oprettet indlægget. Og viser indlægget sammen med de eksisterende indlæg
Slut betingelser: Et diskussionsindlæg er tilføjet til et ord.
Alternative forløb: <ul style="list-style-type: none"> - Brugeren fortryder.

Use case: Kommenter og bedøm forslag
ID: UC-17
Kort beskrivelse: Tilføj en bedømmelse og en kommentar til et forslag.
Primære aktører: Alle kendte brugere.
Sekundære aktører: Ingen.
Start betingelser: Det valgte forslag er ikke bedømt af den aktuelle bruger. Dvs. at et forslag kun kan bedømmes en gang pr. bruger.
Hovedforløb: <ol style="list-style-type: none"> 1. include (Login) 2. Brugeren er på siden for et valgt forslag. 3. Brugeren vælger et "Tilføj bedømmelse" link. 4. Systemet viser en side med en tekstboks til en kommentar og 5 stjerner til bedømmelse. Når en bruger har musen over en af stjernerne så fremkommer en af følgende tekster: "Dårlig", "Under middel", "Middel", "God" og "Perfekt". 5. Brugeren udfylder tekstboksen med sin bedømmelse, trykker på den stjerne som brugeren synes indlægget fortjener og trykker på en "Gem" knap. 6. Systemet gemmer bedømmelsen, dato/tid og brugeren for denne.
Slut betingelser: En bedømmelse er tilføjet til et forslag.
Alternative forløb: <ul style="list-style-type: none"> - Brugeren fortryder.

Use case: Vis ord
ID: UC-18
Kort beskrivelse: Vise et ord med beskrivelse. Liste med forslag og en liste med diskussionsindlæg.
Primære aktører: Alle.
Sekundære aktører: Ingen.
Start betingelser: Ingen.
Hovedforløb: <ol style="list-style-type: none"> 1. Brugeren vælger et ord enten fra en søgning (se use case "Søg") eller fra siden "Alle forespørgsler" (se use case "Vis alle forespørgsler") eller fra siden "Mine forespørgsler (se use case "Vis egne forespørgsler") m.fl. 2. Systemet viser en side med ordet, beskrivelse af dette samt en liste med forslag til ordet, sorteret efter bedømmelse (dvs. en gennemsnitlig bedømmelse af forslaget) og en liste med diskussionsindlæg sorteret efter dato. 3. Hvis det er en bruger som er indlogget, er der mulighed for at afkrydse et felt, som indikerer at den aktuelle bruger, ønsker e-mail notificering når der er nye forslag til aktuelle ord.
Slut betingelser: Ingen.
Alternative forløb: <ul style="list-style-type: none"> - Brugeren fortryder.

Use case: Vis forslag
ID: UC-19
Kort beskrivelse: Vise et forslag med beskrivelse. Samt en liste med bedømmelser.
Primære aktører: Alle.
Sekundære aktører: Ingen.
Start betingelser: Ingen.
Hovedforløb: <ol style="list-style-type: none"> 1. Brugeren vælger et forslag enten fra en ord siden (se use case "Vis ord") eller fra siden "Mine uploads" (se use case "Vis egne uploads"). 2. Systemet viser en side hvor forslag til ordet kan afspilles, en beskrivelse af forslaget samt en liste med bedømmelser af forslaget sorteret efter dato.
Slut betingelser: Ingen.
Alternative forløb: <ul style="list-style-type: none"> - Brugeren fortryder.

Use case: Anmeld
ID: UC-20
Kort beskrivelse: Anmeld et forslag/video for at være upassende.
Primære aktører: Alle kendte brugere.
Sekundære aktører: E-mail server, systemadministrator.
Start betingelser: Ingen.
Hovedforløb: <ol style="list-style-type: none"> 1. include (Login) 2. Brugeren befinder på siden "Vis forslag" se use case "Vis forslag". Her benytter brugeren "Anmeld som upassende" linket. 3. Systemet viser en side med en tekstboks, til beskrivelse af det man finder upassende. 4. Brugeren udfylder ovenstående felt, og trykker på knappen "Anmeld". 5. Systemet sender en e-mail til en af systemet kendt systemadministrator, med følgende oplysninger: Videofil navn, uploadet af brugernavn samt anmeldt af brugernavn.
Slut betingelser: Ingen.
Alternative forløb: <ul style="list-style-type: none"> - Brugeren fortryder.

Use case: E-Mail notifikation
ID: UC-21
Kort beskrivelse: Brugerne modtager hver mandag morgen ca. kl. 8 en e-mail, med links til de ord som har fået nye forslag og som de har ønsket e-mail notifikation på.
Primære aktører: Alle kendte brugere.
Sekundære aktører: E-mail server.
Start betingelser: Ingen.
Hovedforløb: <ol style="list-style-type: none"> 1. Systemet starter e-mail notifikation hver mandag kl. 8. 2. For alle brugere som har en eller flere ord de ønsker e-mail notifikation på. <ol style="list-style-type: none"> 2.1. Er der en eller flere ord som brugeren ønsker at blive notificeret om, som har nye upload siden forrige mandag. 2.2. Hvis ja, så danner og sender systemet en e-mail til brugeren, med links til de ord, som har fået nye uploads.
Slut betingelser: Ingen.
Alternative forløb: Ingen

Brugerrollen "Administrator" er ikke beskrevet i nogle af de ovenstående use cases. Denne bruger skal kunne alt det, som de alm. brugere kan på deres egne sager. Dvs. at administrator har ret til at rette på alt, for at gøre "Tegn til tiden" siden mere strømlinet. En redaktør med andre ord.

4.3. KRAV/USE CASE MATRIX

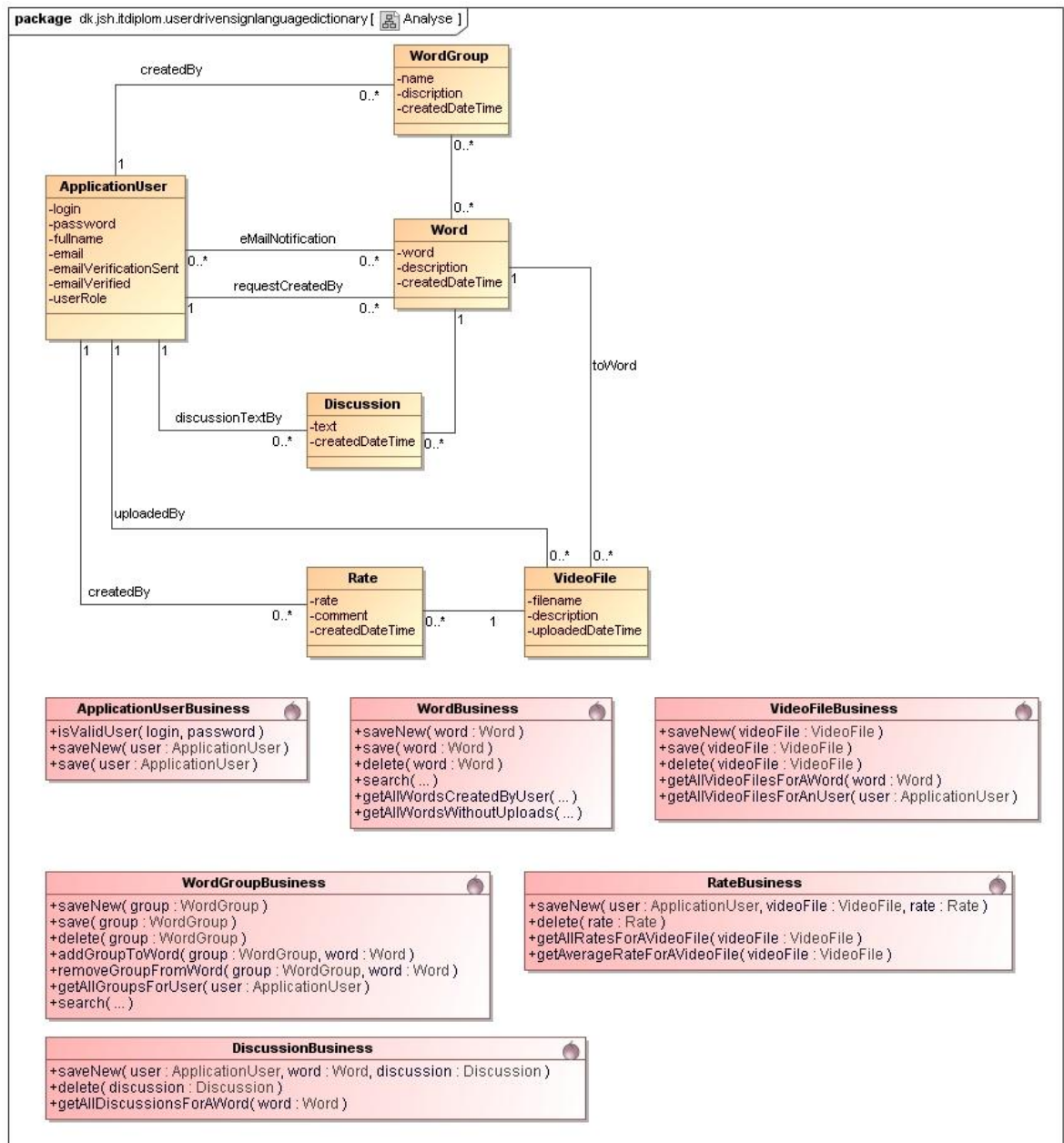
Følgende krav/use case matrix er lavet for at sikre at alle krav er behandlet i en eller flere use cases.

Krav:	A1	A2	A3	A4	A5	A6	A7	A8	A9	A10	A11	A12	A13	A14
UC-1		√					√	√	√					
UC-2			√				√							
UC-3			√											
UC-4								√						
UC-5									√					
UC-6	√											√		
UC-7														√
UC-8														√
UC-9													√	
UC-10													√	
UC-11													√	
UC-12														
UC-13														
UC-14														
UC-15														
UC-16											√			
UC-17				√	√									
UC-18														
UC-19														
UC-20			√			√								
UC-21			√											

Som det fremgår af ovenstående matrix så er krav 10, ikke beskrevet i nogen use case, dette er også i orden da det omhandler system brugeren, som skal oprettes direkte i databasen.

4.4. ANALYSEDIAGRAM

For at komme videre i analysefasen har jeg udarbejdet følgende klassediagram.



Figur 3 - Analyse klassediagram

Alle de gule klasser er entitetsklasser. Her har jeg valgt ikke at have nogle get og set metoder, da disse ikke giver nogen værdi for diagrammet. Jeg har valgt at opdele data og forretningslogik i hver sit sæt klasser. Bl.a. fordi jeg har valgt at bruge Hibernate, til at lette overgangen mellem den objekt orienteret verden og den relationelle database verden. Mere om det under design.

Den mest centrale entitetsklasserne er `ApplicationUser` som alle de andre entitetsklasser refererer til. `ApplicationsUser` indeholder oplysninger og systemes brugere. `Word` entitetsklassen benyttes til de ord som de enkelte brugere, ønsker forslag til. Det er også den der har e-mail

notifikations link til ApplicationUser. For at gruppere ord i ordgrupper benyttes klassen WordGroup. Discussion klassen benyttes til diskussionsindlæg til et ord. Videofile klassen indeholder oplysninger om forslag, som er uploadet til et ord. Og sidst men ikke mindst er der Rate som benyttes til at bedømme video uploads af forslag.

De lilla business klasser (også kaldet kontrolklasser, cirklen er en stereotype) er delt op i følgende klasser:

- ApplicationUserBusiness – til håndtering af brugere (opret, ret og login validering)
- WordBusiness – til håndtering af Word (Opret, ret, slet og søgning efter ord og grupper)
- WordGroupBusiness – til håndtering af WordGroup (Opret, ret, slet, tilføj ord til gruppe, fjern ord fra gruppe og find alle grupper som er oprettet af en bruger)
- DiscussionsBusiness – til håndtering af Discussions (Opret, slet og find indlæg til et ord)
- VideoFileBusiness – til håndtering af videofile uploads (Opret, ret, slet, find alle upload til et ord og find en brugers uploads)

Som det fremgår af overstående diagram, har jeg to typer klasser, de gule entitetsklasser og de lilla kontrolklasser. Disse to typer har jeg valgt at placere i hver sin pakke, det vil fremgå af mine design diagrammer.

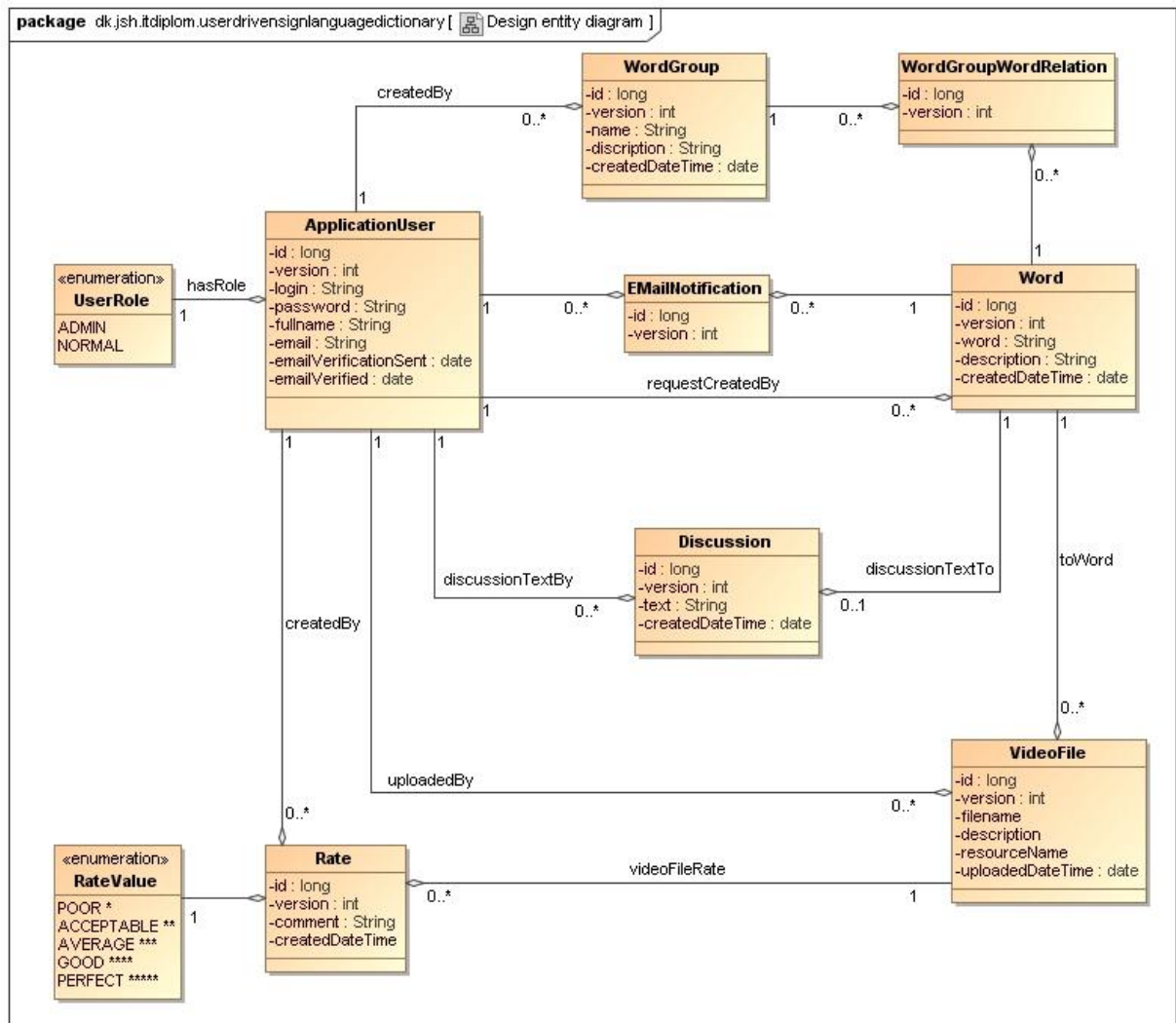
Jeg har valgt at "låse" analysen, det skal forstås på den måde, at hvis jeg bliver "klogere" senere i processen, f.eks. under design, så går jeg ikke tilbage og tilretter analysemodellerne, men indarbejder rettelserne i designmodellerne. Det mener jeg godt det kan forsvares da modellen er ret simpel.

For at afgrænse opgaven har jeg ikke medtaget nogen analyse use case realiseringer. Det mener jeg også kan forsvares, da de fleste operationer er meget simple: læs, opret, ret og slet operationer også kendt som CRUD (Create, Read, Update, Delete) operationer.

Det afslutter så analysedelen, og bringer os videre til designdelen.

5. DESIGN

Jeg er kommet frem til følgende klassediagram for entitetsklasserne.



Figur 4 - Design entitetsklassediagram

Ovenstående entitetsklasser placeres i en pakke kaldet:
`dk.jsh.itdiplom.userdrivensignlanguagedictionary.entity`

Dette er måden man navngiver pakker på i Java.

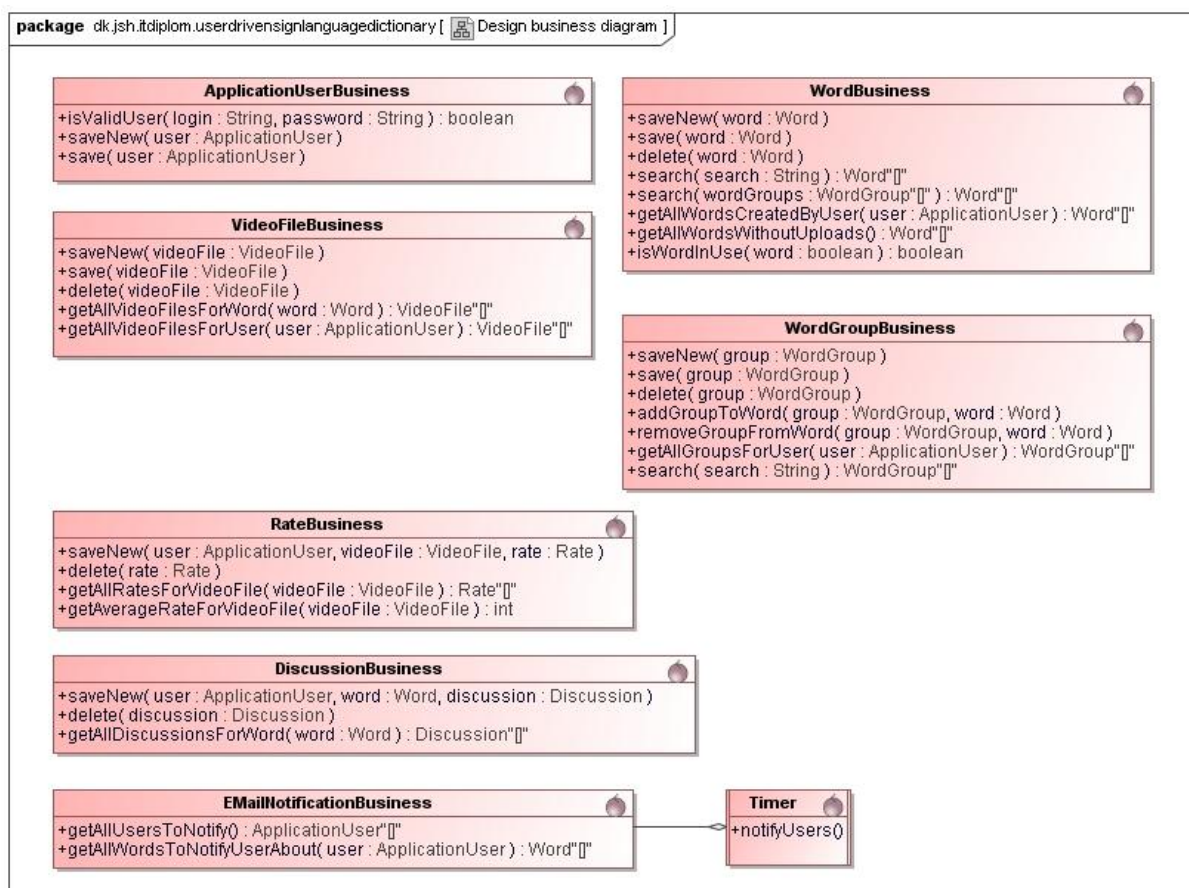
I forhold til analyseklassediagrammet, så er der to "mange til mange" relationer som er blevet til klasser, det er `eMailNotification` relationen, som er blevet til klassen `EmailNotification` og relationen mellem `Word` og `WordGroup`, som er blevet til `WordGroupWordRelation`.

Så er der to attributter, som jeg har lavet om til klasser med stereotype "enumeration". Det er attributten `"userRole"` som er blevet til `UserRole` klassen, samt attributten `rate` fra `Rate` klassen, som er blevet til `RateValue` klassen. Disse klasser laves som Java enums, som er en form for konstanter.

Derudover er alle associationer fra analysediagrammer rettet til aggregeringer.

Alle ovenstående klasser, minus enumeration klasser, skal mærkes med Hibernate/Java annotations¹⁰. Så Hibernate ved hvordan de enkelte klasser, skal mappes til databasetabeller. Disse annotations benyttes også til at generere et DDL¹¹ database skema. Alle klasser har også fået en id og en version attribut. Id bliver brugt som primær nøgle i database, og version bliver brugt i forbindelse med optimistisk låsning¹².

Mine kontrolklasser kan ses af følgende diagram, som ligger i pakken:
dk.jsh.itdiplom.userdrivensignlanguagedictionary.business.

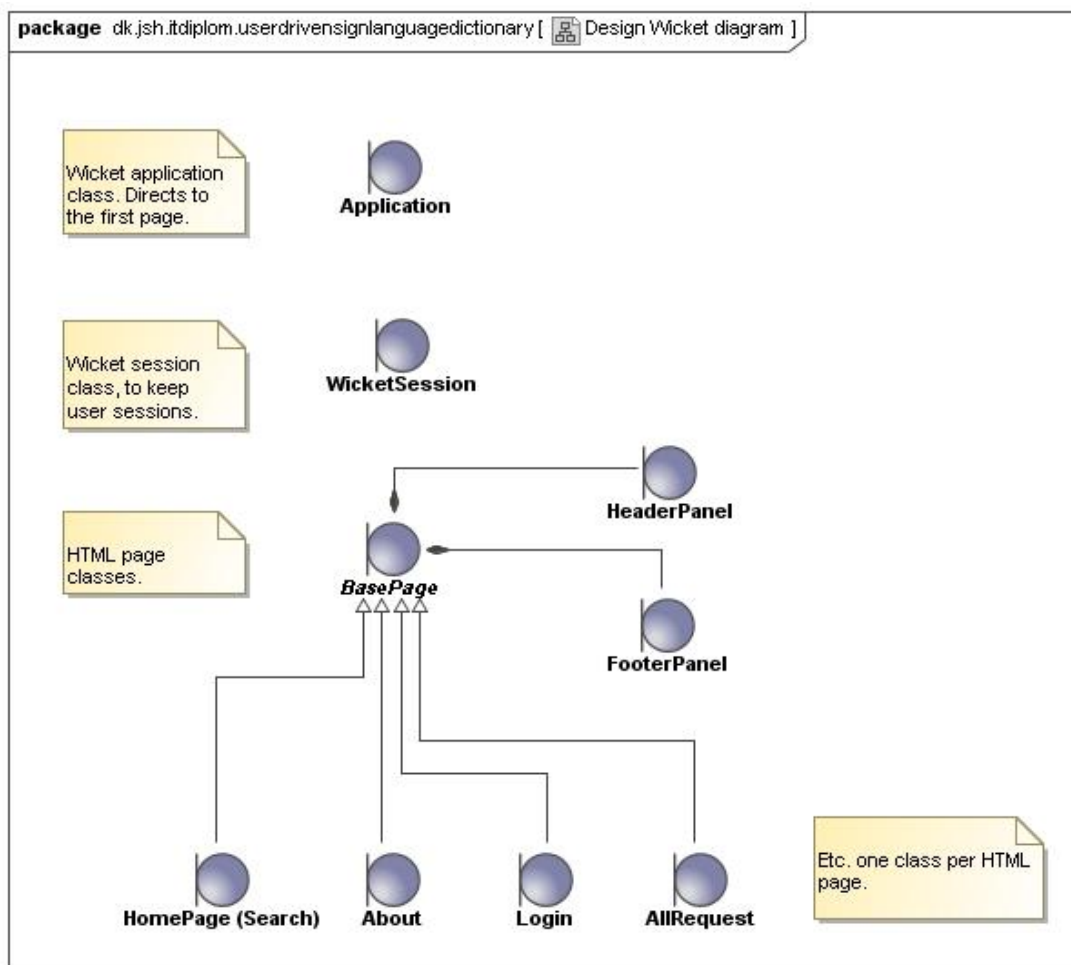


Figur 5 – Design kontrolklassediagram

Her har jeg tilføjet nogle ekstra klasser, bl.a. en Timer, som er en singleton¹³ klasse, der vha. EMailNotificationBusiness klassen (som også er ny i forhold til analyseklasse diagrammet), starter e-mail notifikation, hver mandag morgen kl. 8. Dette skal foregå i en separat programtråd¹⁴.

Derudover har alle metoder nu fået attributter og evt. return værdier. De steder hvor der returneres arrays, vil i Java blive implementeret som lister, dvs. at der vil blive returneret List<type>. F. eks metoden search(String search) i klassen WordBusiness er i diagrammet beskrevet som om den returnere en array af Word objekter. Det vil i Java blive implementeret som List<Word>.

Den næste pakke som jeg vil beskrive er de klasser som skal bruges til brugerflade programmeringen, disse klasser kommer til at ligge i pakken:
dk.jsh.itdiplom.userdrivensignlanguagedictionary.wicket



Figur 6 – Design userinterface (boundary) diagram

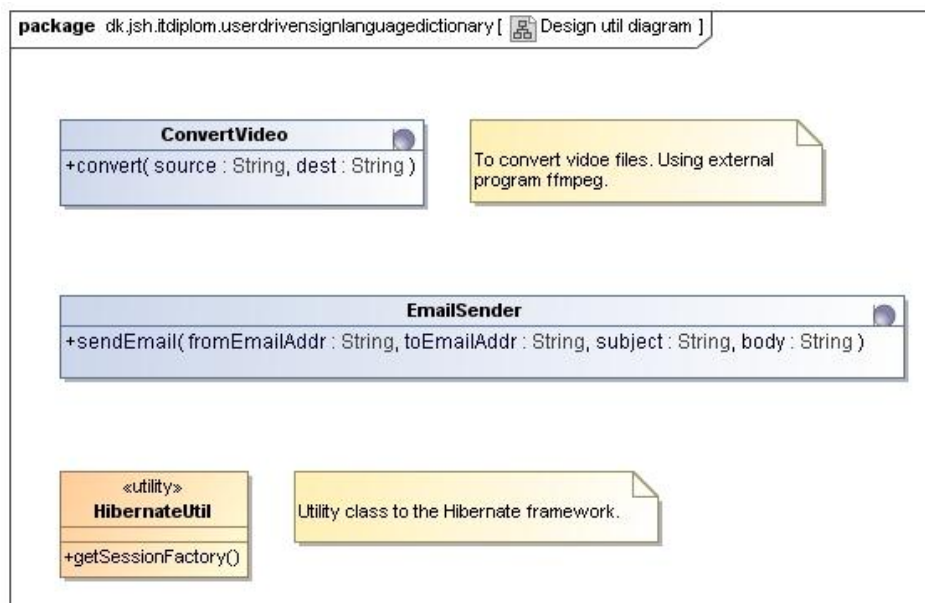
Her har jeg brugt stereotypen "boundary", da det er brugerfladeklasser. Jeg har ikke beskrevet nogle attributter til de enkelte klasser. Diagrammet skal bruges til at beskrive strukturen i mine brugerfladeklasser. Pakkenavnet ender med .wicket for at indikere at det er dette Java framework som benyttes. Jeg kommer lidt mere ind på dette framework senere.

En kort beskrivelse af de enkelte klasser:

- Application – Er Wicket's start klasse, benyttes bl.a. til at pege på den første side brugerne skal møde.
- WicketSession – Er Wicket's session klasse, til at holde de enkelte brugeres web sessioner.
- BasePage – Er en abstrakt klasse, som de enkelte sider arver fra. Med følgende to klasser, som alle sider benytter
 - HeaderPanel – som beskriver toppen af alle sider
 - FooterPanel – som beskriver bunden af alle sider
- Og i bunden af diagrammet komme de enkelte sider, som alle arver fra BasePage. Ikke alle sider er med. Men Wicket frameworket ligger op til en klasse per side.

Den sidste pakke som jeg vil beskrive er pakken:

dk.jsh.itdiplom.userdrivensignlanguagedictionary.util



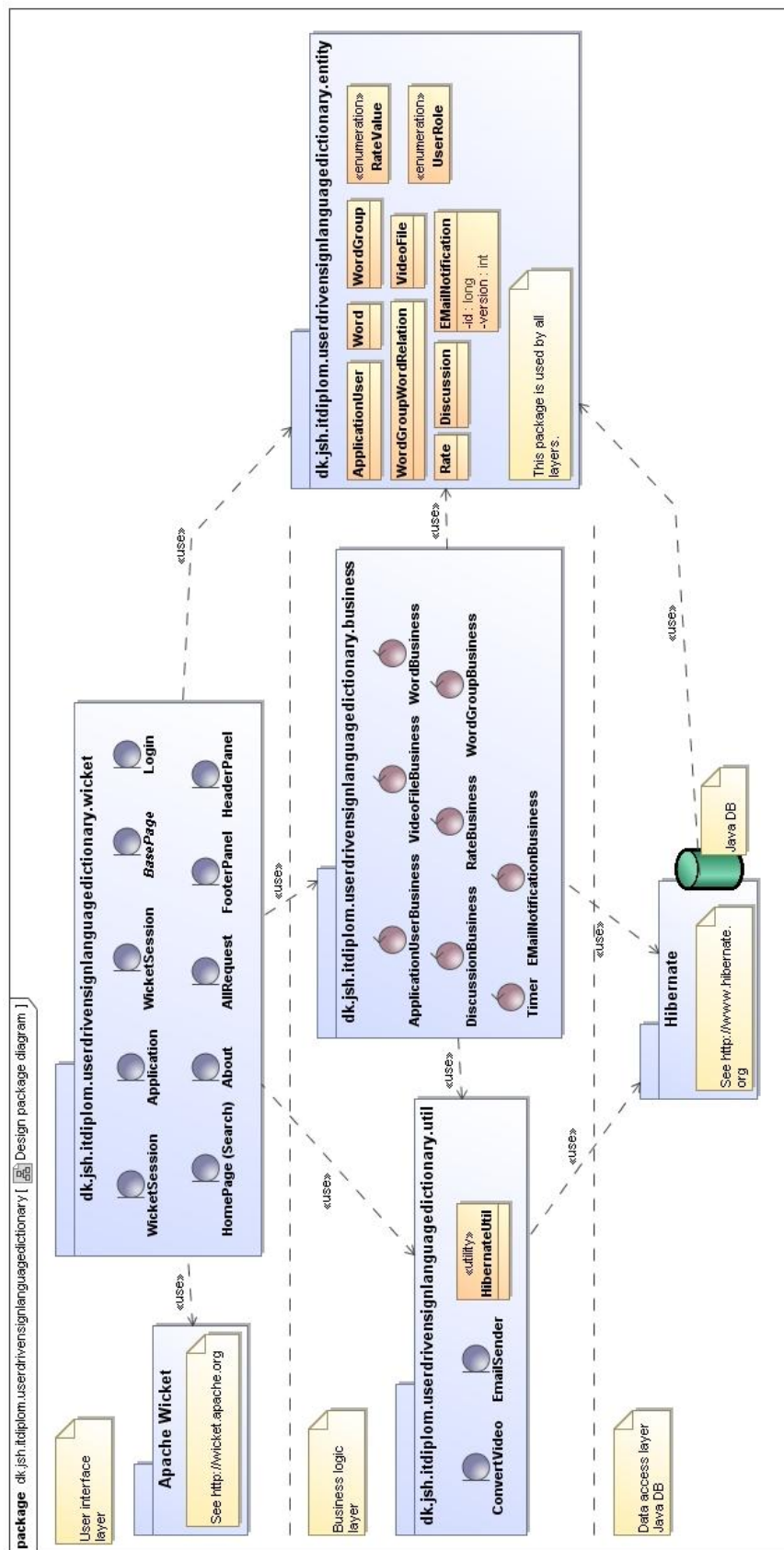
Figur 7 – Design utility diagram

Denne pakke er en "rode" kasse for klasser, som ikke tilhører de tre centrale pakker, som er beskrevet ovenover.

Følgende er en kort beskrivelse af de enkelte klasser:

- **ConvertVideo** – klasse til konvertering af videofiler fra f.eks. avi formatet til ogg formatet. Vha. det eksterne program FFMPEG¹⁵.
- **EmailSender** – klasse til at sende e-mails til en e-mail server. Jeg har benyttet Apache James¹⁶ som e-mail server under udviklingen. Denne server har jeg sat op til at dirigere alle e-mails til en fast e-mail modtager, for at lette test af systemet.
- **HibernateUtil** – klasse til Hibernate frameworket, som bruges til at få en Hibernate session (ikke en web session). Disse sessioner bruges til at samle database transaktionen, så der kan laves en commit hvis alt går godt, eller en rollback hvis der er fejl.

Det samlede design kommer til at se således ud.



Figur 8 - Design pakkediagram

Som det ses af pakkediagrammet, er det opdelt i følgende 3 lag:

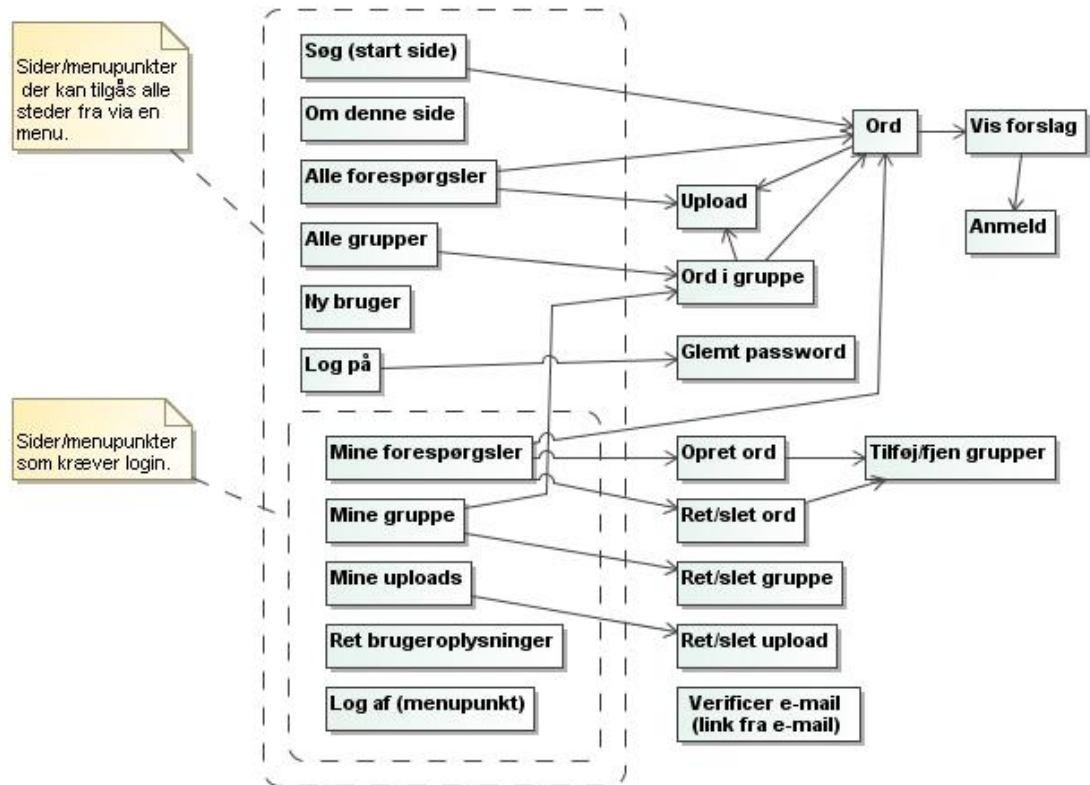
- User interface – til håndtering af websiderne.
- Business logic – til håndtering af forretningslogik.
- Data Access – til håndtering af database delen. Vha. Hibernate.

Pakken `dk.jsh.itdiplom.userdrivensignlanguage.entity`, er til transport klasser mellem lagene. Derudover skal klasserne i entity pakkes have tilføjet Hibernate annotations, så Hibernate kan finde ud at mappe disse klasser til tabeller i databasen. Hibernate kan også ud fra disse annotations danne en database DDL skemafiler.

Pakkerne Apache Wicket og Hibernate, er med for at illustrere at disse to frameworks benyttes.

5.1. WEBSITE DESIGN

Jeg er kommet frem til følge webside flow i "Tegn til tiden".



Figur 9 - Side flow

Som det fremgår af diagrammet, kan nogle sider/menupunkter tilgås af alle, mens andre kræver login. Upload siden kræver også login. Verificer e-mail siden kan kun nås via et link fra en e-mail, som systemet har sendt til brugeren.

I det følgende vil jeg beskrive sideopbygningen for de vigtigste sider. Til dette har jeg også brugt MagicDraw, som har mulighed for at designe brugerflader. Det er derfor de følgende sidelayouts minder lidt om en Windows brugerflader.

Det første sidelayout er startside, som også er søgesiden.

The screenshot shows the 'Tegn til tiden' web application. The browser address bar displays 'package dk.jsh.itdiplom.userdrivensignlanguagedictionary [SearchAndHomePage]'. The page has a header 'Tegn til tiden' and a 'Brugernavn:' label. On the left is a 'Menu' section with links: 'Søg', 'Om denne side', 'Alle forespørgsler', 'Alle Grupper', 'Ny bruger', and 'Log på'. The main area is titled 'Søg' and contains a search form with 'Søg efter:' and two input fields: 'Ord' (a dropdown menu) and 'Lib*' (a text input). A 'Søg' button is to the right. Below the search form is a 'Resultat' section showing 'Libyen', 'Oprettet den dd/mm-yyyy', and 'Grupper: Det arabiske forår og Nordafrika'. At the bottom right of the results is a pagination control '<< < 1 > >>'. Annotations with dashed lines point to various parts of the interface: a top-right note points to the header area; a bottom-left note points to the menu; a bottom-center note points to the search results; and a bottom-right note points to the pagination control.

package dk.jsh.itdiplom.userdrivensignlanguagedictionary [SearchAndHomePage]

Tegn til tiden

Brugernavn:

Menu

- Søg
- Om denne side
- Alle forespørgsler
- Alle Grupper
- Ny bruger
- Log på

Søg

Søg efter: Ord Lib* Søg

Resultat

Libyen
Oprettet den dd/mm-yyyy
Grupper: Det arabiske forår og Nordafrika

<< < 1 > >>

Fast top for alle side, med overskriften "Tegn til tiden" og navnet på den aktive bruger

Fast menu for alle sider. Antallet af menupunkter varierer efter om brugeren er indlogget eller ej.

Arbejdsområde, her er det søge siden der vises.

Liste navigering, hvis listen ikke kan være på en side.

Figur 10 – Start side

I det efterfølgende, følger nogle eksempler på hvordan arbejdsområdet kan se ud. For at afgrænse opgaven, har jeg ikke medtaget alle sider. Det første arbejdsområde er "Log på".

The screenshot shows the 'Log på' login page. The browser address bar displays 'package dk.jsh.itdiplom.userdrivensignlanguagedictionary [Logon]'. The page has a title 'Log på'. It contains a login form with two input fields: 'Brugerkode:' and 'Password: *****'. Below the password field is a link 'Glemt password' and a 'Login' button.

package dk.jsh.itdiplom.userdrivensignlanguagedictionary [Logon]

Log på

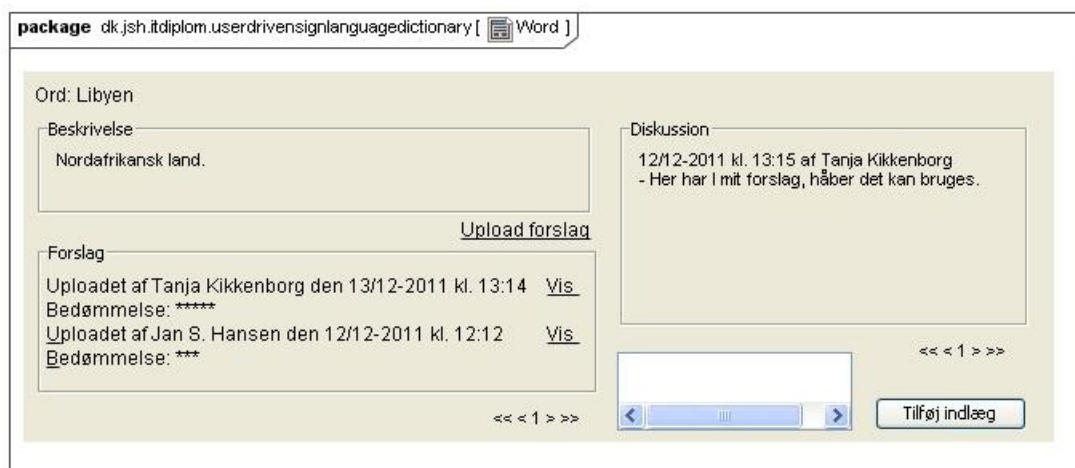
Brugerkode:

Password:

[Glemt password](#) Login

Figur 11 - Loginside

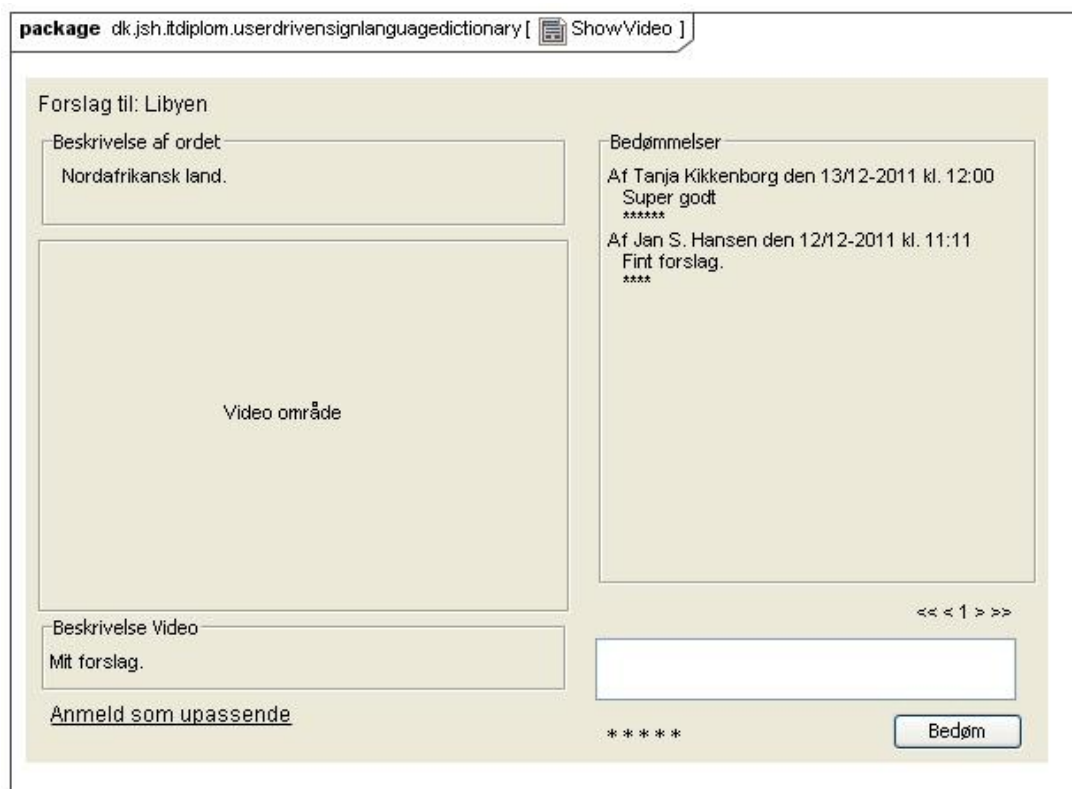
Login siden, linker til en "Glemt password" side, hvor burgeren bliver bedt om at indtaste sin e-mail adresse. Hvis denne er kendt af systemet, sendes der en e-mail med et nyt password. Næste arbejdsområde er "Ord".



Figur 12 – Ord side

Bedømmelsen på denne side er et gennemsnit af alle bedømmelser til den aktuelle video. Man kunne overveje at bruge to faneblade til ovenstående side, så Diskussionsdelen får en fane for sig selv. På de diskussionsindlæg man selv har lavet, vil der være et slet link. Muligheden for at tilføje indlæg er kun for indloggede brugere.

Næste sider er "Vis forslag" siden og "upload" siden, som ovenstående side linker til.



Figur 13 – Vis forslag side

Her kunne man også opdele siden i to faneblade, så bedømmelsesdelen får sin egen fane. Også her vil der være et slet link på den aktuelle brugers bedømmelser. Muligheden for at

bedømme videoen og anmelde som upassende, vil kun være for indloggede brugere. "Anmeld som upassende" linker til en ny side med en enkelt tekstboks og en Anmeld knap. De 5 stjerner skal indikere hvordan man bedømmer, ved at vælge det antal stjerner som man mener video fortjener, som man kender det fra mange andre web sider. Når man "hover" med musen over en af stjernerne, skal følgende tekster fremkomme: "Dårlig", "Under middel", "Middel", "God" og "Perfekt".

Figur 14 – Upload video side

Den næste side er "Mine forespørgsler", den bruges også som udgangspunkt for de andre "liste" sider, som er "Mine grupper", "Mine uploads", "Alle forespørgsler" og "Alle grupper".

Figur 15 – Mine forespørgsler side

Her har man mulighed for at oprette nye forespørgsler vha. "Opret ny forespørgsel" linket. I listen har hver forespørgsel to links, selve ordet som linker til "Ord" siden, og linket "Ret" som linker til en ny side, hvor forespørgslen kan rettes.

package dk.jsh.itdiplom.userdrivensignlanguagedictionary [CreateRequest]

Opret forespørgsel

Ord der ønskes forslag til:

Beskrivelse af ordet:

Figur 16 – Opret forespørgsel side

Ovenstående side er til oprettelse af en ny forespørgsel. Ret/slet forespørgsel siden vil være som denne, bare med en "Slet" knap, som ikke kan benyttes, hvis der er uploads til ordet. Den næste side jeg vil gennemgå er "Tilføj/fjern grupper" som man kommer til ved at trykke på knappen "Gem og tilknyt grupper".

package dk.jsh.itdiplom.userdrivensignlanguagedictionary [AddRemoveGroups]

Tilføj/fjern grupper til ordet: Libyen

Alle grupper:

Nordafrikanske Lande	>	Valgte grupper:
Det arabiske forår		

<

Opret og tilknyt ny gruppe

Navn:

Beskrivelse:

Figur 17 – Tilføj/fjern grupper side

Denne side benyttes til at tilføje og fjerne grupper til et ord. Vha. to lister, en med alle ikke valgte grupper og en med valgte grupper. Her har brugeren mulighed for at flytte grupper frem og tilbage med de to "<" og ">" knapper. Hvis musen holdes stille over en af grupperne i de to lister fremkommer gruppebeskrivelsen. Siden giver også mulighed for at oprette nye grupper.

Jeg vil slutte af med "Opret ny bruger" og verificer e-mail siden.

package dk.jsh.itdiplom.userdrivensignlanguagedictionary [NewUser]

Ny bruger

Navn:

Brugerkode:

Password:

Gentag password:

E-mail:

Figur 18 – Opret ny bruger side

Siden "Ret brugeroplysninger" vil være som ovenstående side, bortset fra at "Brugerkode" ikke kan rettes. Hvis der ikke er problemer på ovenstående side, dvs. alle felter er udfyldt og "Brugerkode" er ikke brugt af en anden, "Password" og "Gentag password" er ens og opfylder password krav, sendes der en e-mail til den indtastede e-mail adresse med følgende tekst:

Velkommen til Tegn til tiden.

Før du kan logge på systemet skal du trykke på følgende link, for at bekræfte din mail adresse.

[Bekræft e-mail](#)

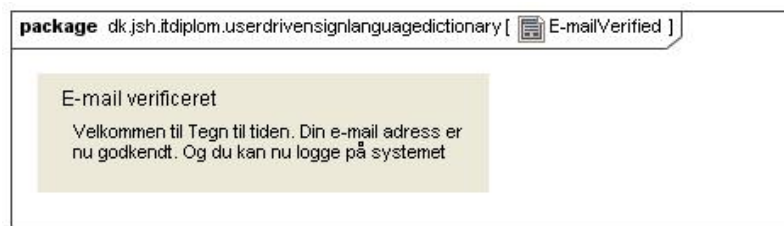
OBS! - Denne mail kan ikke besvares.

Eventuelle spørgsmål kan rettes til Jan Schøder Hansen på e-mail: jan.sch.hansen@gmail.com

Med venlig hilsen

Tegn til tiden

Når brugeren trykker på linket i mailen, kommer denne til følgende side:



Figur 19 – E-mail verificeret side

5.2. FRAMEWORKS

Følgende afsnit er en kort gennemgang, af de to frameworks jeg har valgt at benytte. Dvs. Wicket frameworket til webside programmering og Hibernate frameworket, som bruges til at komme fra det objekt orienteret domaine til en relationel database.

Wicket:

På Wickets hjemmeside kan man læse om de mål, udviklerne har haft med Wicket¹⁷. Jeg vil her komme ind på de punkter, jeg selv som udvikler ligger vægt på.

Som udvikler har jeg arbejdet med Struts¹⁸ 1.3, samt med ASP.NET¹⁹, og en af de problemer som er i disse frameworks, er bl.a. at man blander kode med HTML.

I Struts arbejdes der med Java, JSP²⁰ tags, HTML²¹ tags, JavaScript²² i en og samme fil. Det samme gør sig gældende i ASP.NET, hvor det bare er C#²³ kode, ASP tags, HTML tags. Det giver nogle filer som er svære at overskue, genbruge og vedligeholde.

Helt grundlæggende prøver Wicket, at adskille HTML og kode. Det giver mulighed for, at det faktisk er muligt at få en HTML/CSS²⁴ specialist, til at lave selve HTML/CSS koden, som så kan overtages af en programmør. Det er ikke rigtigt muligt med f.eks. Struts og ASP.NET, da der er så mange specielle tags, som en HTML specialist ikke kender til, og som ikke kan håndteres af webdesignerens værktøjer.

Wicket benytter også nogle gamle Java dyder som, en klasse, en Java fil. I Wicket kan en Wicket webside, beskrives vha. en HTML fil og en Java fil. Begge ligger i samme katalog og hedder det samme, på nær fil endelsen. Det eneste krav Wicket stiller til HTML filerne er, at de HTML elementer der skal være dynamiske, skal have en entydigt Wicket identifikation.

Hibernate:

Hibernate bruges til at komme fra den objekt orienteret verden til den relationelle databasen verden. Med Hibernate kan man selv bestemme, om man vil starte med et database design eller med en nogle entitets klasser. Hvis man vælger at starte med et DDL¹¹ skema, kan Hibernate danne Java eller C# klasser som matcher skemaet. Eller omvendt, som jeg har valgt, at få lavet et DDL skema (se skema under Bilag afsnit 9.4) ud fra mine Java entitetsklasser vha. annotations (Se Bilag afsnit 9.4 for eksempler for dette).

En anden fordel ved Hibernate er at det er nemt at lave optimistisk låsning, som i korte træk består i, at systemet går ud fra, at de enkelte brugerer ikke arbejder på samme data. Hvis der så er konflikter, så er det først til mølle princippet, der bestemmer, hvem der kommer igennem med sin opdatering. Det er derfor alle entitetsklasserne, under design fik attributter version tilføjet. Så kan Hibernate vha. af denne attribut, samt en annotation selv lave optimistisk låsning. F. eks. hvis bruger A læser en række med id = 1 og version = 1, og en bruger B læser den samme række.

Bruger A opdaterer rækken, hvor Hibernate øger version med 1, så den nu er 2. Nu vil bruger B også opdatere denne række, men bruger B får en fejl. Fordi Hibernate prøver at lave en update med følgende where sætning: "where id = 1 and version = 1". Da denne række ikke findes mere, vil Hibernate returnere en exception, som fanges af systemet, og fortæller brugeren at de data han forsøgte at gemme, er rettet af en anden bruger i mellemtiden. Men som min applikation er designet, vil der ikke være mange konflikter. Da brugerne kun kan rette og slette egne ord, grupper m.m. Det er kun i de sjældne tilfælde hvor en administrator retter i det samme som en alm. bruger.

Derudover giver Hibernate mulighed for at bruge HQL²⁵, som er det samme som SQL²⁶ med den krølle, at i stedet for tabelnavne og kolonnenavne benyttes der klassenavne og klasseattributter. Se eksemplet i næste afsnit om sikkerhed.

5.3. SIKKERHED

Man kan ikke lave en offentligt tilgængelig webapplikation, uden at komme ind på sikkerhed. Så her følger de tanker jeg har gjort mig om sikkerhed.

Wicket er ifølge wickets hjemmeside²⁷ "secure by default". Og jeg har heller ikke kunne fremprovokere diverse "Injection flaws"²⁸ angreb.

SQL injections forebygges også vha. Hibernate's måde at lave HQL statements på, f.eks som i følgende kode:

```
StringBuilder hql = new StringBuilder();
hql.append("select word from ");
hql.append("dk.jsh.itdiplom.userdrivensignlanguagedictionary.entity.");
hql.append("Word word ");
if (useLike) {
    hql.append("where lower(word.word) like :search ");
}
else {
    hql.append("where lower(word.word) = :search ");
}
hql.append("order by word.word");
Query query = session.createQuery(hql.toString());
query.setString("search", search);
```

Som det fremgår af koden, bliver ingen variabler direkte indsat i HQL strengen, men via metoden `query.setString("search", search)`, som er en Hibernate metode, der er med til at beskytte mod Injections.

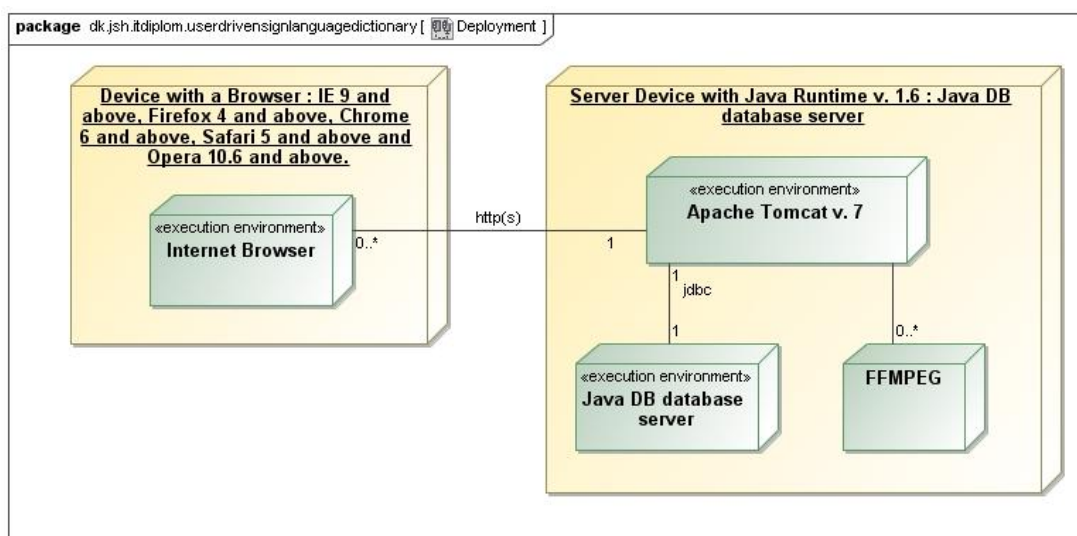
Med hensyn til adgangskontrol vil jeg anbefale at benytte HTTPS/SSL²⁹ under selve login, opret by bruger og ret brugeroplysninger, for at sikre at login og password ikke bliver opsnappet. Derudover mener jeg ikke, at der er noget i denne Webapplikation, som er så følsomt at det kræver HTTPS.

I databasen bør password ikke stå i klar tekst, men som en hashed³⁰ værdi, som kun kan benyttes til at validere et password med.

OWASP siden (se www.owasp.org) er en rigtig god side, om sikkerhedsproblematikker i webapplikationer.

Da "Tegn til tiden" er en Java webapplikation uden afhængigheder til andre systemer, er selve idriftsættelsen enkel. Det kræver en server med installeret Java og en Apache Tomcat Web Server², samt en Java DB⁴. Java DB er en del af Java. Dvs. at der ikke er nogen specielle krav til operativsystem, da Tomcat og Java DB kan køre på bl.a. Linux, Unix og Windows. Andre webservere kunne også benyttes. Det eneste som ikke er en del af Java verdenen er video konverteringsprogrammer FFMPEG¹⁵. Programmet er open source og findes både til Linux og Windows.

Følgende er et deploymentdiagram.



Figur 20 - Deploymentdiagram

For at idriftsætte en webapplikation i en Tomcat server, bygges der en WAR³¹ fil, vha. et udviklingssystem. Denne WAR fil kopieres over i et specielt Tomcat applikationskatalog. Første gang systemet installeres skal databasen klargøres. Dette gøres med et SQL-script, som opretter alle tabellerne, samt indsætter en administrator bruger.

Som det også fremgår af ovenstående diagram, skal "Tegn til tiden" virke sammen med gængse browsere på markedet. Dvs. Internet Explore, Firefox, Opera, Chrome og Safari. Det vil kræve nogle tests. Men da antallet af websider er begrænset, er det en overkommelig opgave. Men for at HTML5 video tagget skal virke, kræver det nyere Browsere. Versioner for disse fremgår af diagrammet.

7. UP ITERATIONER

Indtil nu har denne rapport fuldt den meget udskældte vandfaldsmodel. Dvs. først indsamles der krav, disse analyseres, hvorefter systemet designes. Dette er også en logisk form, hvis et system skal beskrives i rapportformat. Men for at komme tilbage til UP⁸, har jeg valgt at beskrive hvilke iterationer, som kunne benyttes til at udvikle "Tegn til tiden" efter. Iterationerne jeg har valgt, bygger mere på de områder som hører sammen i applikationen, end på hvor lang tid de enkelte iterationer tager.

Iteration	Beskrivelse
1	Indsamle krav, starte på use case beskrivelser, som skal bruges til login, brugeroprettelse/rettelse og e-mail verificering.
2	Starte på analyse og design for bruger vedligeholdelse.
3	Udvikle en web ramme med header, menu, login og brugeroprettelse forretningslogik og database adgang.
4	Use case, analyse, design og udvikling af "Mine forespørgsler" siden samt "Opret ny forespørgsel" og "Ret forespørgsel". Men ikke noget med grupper. I de følgende iterationer er alle faser med, dvs. use case, analyse, design, programmering og test.
5	Tilføj "Søg" siden.
6	Tilføj "Ord" siden, som udbygges når "Upload" siden er færdig.
7	Tilføj "Mine grupper", "Ret grupper" og "Tilføj/fjern grupper" siderne.
8	Tilføj "Upload", "Mine uploads" og "Ret upload".
9	Tilføj "Alle forespørgsler" siden.
10	Tilføj "Alle grupper" siden.
11	Tilføj E-mail notifikation.
12	Afsluttende test.
13	Idriftsættelse.
14	Vedligeholdelse.

Jeg kan desværre ikke påstå at jeg har fulgt ovenstående. Men de iterative processer som UP, SCRUM³² og XP³³, ligger alle op til at man bliver klogere undervejs og må omprioriterer.

8. KONKLUSION

Jeg er desværre ikke blevet helt færdig med selve programmeringen. For at illustrere hvor meget jeg har nået, har jeg under bilag lavet en brugervejledning, med skærmdumps. Men kun af det jeg har udviklet indtil nu. Håber jeg får lidt tid mellem aflevering af denne opgave og til eksamen, til at udvikle lidt mere. Se brugervejledningen under bilag afsnit 9.2. Ligeledes under bilag har jeg koden og lidt om det udviklingsmiljø jeg har brugt. Se bilag afsnit 9.3 for udviklingsmiljø og afsnit 9.4 for selve koden.

Jeg har været så heldig, at jeg kunne genbruge lidt fra mine tidligere eksamensopgaver på IT-Diplomuddannelsen. Bl.a. fra faget "Objektorienterede metoder" hvor jeg har genbrugt selve rapportopbygningen. Derudover har jeg genbrugt lidt fra opgaven til faget "Databasesystemer og Web", hvor jeg har lånt lidt om Hibernate og optimistisk låsning. Og til slut har jeg lånt lidt fra opgaven til faget "Web og serverprogrammering" hvor jeg har lånt lidt om Wicket og sikkerhed.

Det er altid god skik at komme med "billige" alternative løsninger, og i dette tilfælde burde man undersøge diverse CMS³⁴ løsninger, for at se om de ikke kunne bruges. Det er desværre ikke noget jeg har haft tid til. En anden, efter min mening oplagt mulighed er Wikipedia, eller en løsning som bygger på Wiki teknologi. Mig bekendt understøtter disse ikke video. Men det ville være oplagt at der på f.eks. den danske Wikipedia, kunne være en lille videosekvens for de opslag/ord, som det giver mening at kunne på tegnsprog.

En oplagt videre udviklingsmulighed for denne løsning, ville være at udvikle smartphone applikationer (apps). Dette er der mange grunde til. En er at døve har taget smartphonen til sig, specielt dem med et kamera, som vender mod brugeren selv. Da de på denne måde kan kommunikere på tegnsprog. Og selve telefonen kan jo også bruges til at optage forslag med.

Jeg stod over for at flere muligheder mht. hvordan jeg ville vise video i web applikationen. Mit valgt faldt på HTML 5's videotag. Dette medførte dog nogle problemer, bl.a. at det kun er de nyeste browsere som understøtter dette HTML tag. Dertil skal siges at HTML 5 ikke er en færdig standard endnu. En af de ting som de forskellige browserleverandører ikke er blevet enige om endnu, er hvilke formater der skal benyttes. Ifølge W3School siden (se siden www.w3schools.com/html5/html5_video.asp), kan man nøjes med OGG³⁵ og MPEG-4³⁶ formatet, så skulle de gængse browsere været dækket ind. HTML 5 videotag'et kan dog linke til flere filer, så browseren selv kan vælge hvad den foretrækker. Indtil videre understøtter jeg kun OGG formatet. Hvis man ønsker at understøtte ældre browsere, er man nød til at overveje Adobe Flash³⁷. Det har jeg dog fravalgt, da meget tyder på at det er på vej ud. Bl.a. fordi Apple ikke understøtter det, i mange af deres produkter.

Så fik Steve Jobs det sidste ord igen.

9. BILAG

9.1. DANSKE/ENGELSKE TERMER

Følgende tabel er en liste af de termer som benyttes af systemet. Alle termer får et dansk og et engelsk navn. Bl.a. for at sikre overgangen fra use cases til analyse, design og programmering hvor de engelske termer benyttes.

Dansk term	Engelsk term	Beskrivelse
Bruger	User	En bruger af systemet.
Ord	Word	Et ord som der kan uploades forslag til
Forespørgsel	Request	Forespørgsler til ord, som en bruger ønsker forslag til.
Forslag	Proposals	Forslag til et ord
Videofil	Video file	Video forslag til et ord
Bedømmelse	Rate	Video bedømmelse
Ordgruppe	Word group	Ord gruppe
Diskussion	Discussion	Diskussions indlæg
Brugerrolle	User role	Bruger roller

9.2. BRUGERVEJLEDNING

Følgende er en brugervejledning til "Tegn til tiden". Dog har jeg kun medtaget det som jer har nået at udvikle.

Den første side brugerne af "Tegn til tiden" møder er "Søge siden".

Tegn til tiden

Bruger: Ikke logget på

Menu

- Søg
- Om denne side
- Alle forespørgsler
- Alle grupper
- Ny bruger
- Log på

Søg

Søg efter: Ord

Resultat

<< < > >>

Udviklet af Jan Schrøder Hansen - Efteråret 2011

Som det fremgår af siden, er man ikke logget på systemet endnu. Til venstre er der en menu med de muligheder, der er for ikke indloggede brugere. På selve siden er der mulighed for at søge efter ord eller efter ord i grupper. I søgefeltet kan der bruges følgende wildcards:

Wildcard	Beskrivelse
*	Erstatter 1 eller flere tegn. F. eks. hvis man ønsker alle ord som starter med "Nord" så kan man skrive "Nord*" i feltet.
?	Erstatter 1 og kun et tegn.

Hvis der f.eks. søges efter grupper som starter med "Nord" så kunne følgende side fremkomme.

Tegn til tiden

Bruger: Ikke logget på

Menu

- [Søg](#)
- [Om denne side](#)
- [Alle forespørgsler](#)
- [Alle grupper](#)
- [Ny bruger](#)
- [Log på](#)

Søg

Søg efter:

Resultat

Egypten

Oprettet den 20/11-2011 20:31
Grupper: Det arabiske forår og Nordafrika.

Libyen

Oprettet den 19/11-2011 14:40
Grupper: Det arabiske forår og Nordafrika.

<< < 1 > >>

Udviklet af Jan Schrøder Hansen - Efteråret 2011

Som det fremgår her er der to ord som indgår i en gruppe som hedder "Nordafrika".

For alle lister i "Tegn til tiden" er der max. 4 "linier" per side. Og man kan navigere i siderne ved at trykke på symbolerne i højre hjørne af listerne.

Symbol	Beskrivelse
<<	Gå til første side.
<	Gå til forrige side.
1	Gå til den valgte side. Der kan f.eks. stå 1 2. Det vil sige at der er to sider.
>	Gå til næste side.
>>	Gå til sidste side.

Hvis man så trykker på "Egypten" linket så kommer følgende side frem.

Tegn til tiden

Bruger: Ikke logget på

Menu

- [Søg](#)
- [Om denne side](#)
- [Alle forespørgsler](#)
- [Alle grupper](#)
- [Ny bruger](#)
- [Log på](#)

Ord: Egypten

Beskrivelse

Forslag

<< < 1 > >>

Udviklet af Jan Schrøder Hansen - Efteråret 2011

Her fremgår det at der kun er et forslag til ordet. Hvis man trykker på "Vis" linket kommer følgende side.

Tegn til tiden

Bruger: Ikke logget på

Menu


- [Søg](#)
- [Om denne side](#)
- [Alle forespørgsler](#)
- [Alle grupper](#)
- [Ny bruger](#)
- [Log på](#)

Forslag til: Egypten

Tilbage

Beskrivelse af ordet

Landet i nordafrika.



Beskrivelse af video

Mit forslag til Egypten. Men tegner en pyramide med hænderne.

Udviklet af Jan Schrøder Hansen - Efteråret 2011

Her kan forslaget til ordet ses ved af afspille videoen. Derudover er der en beskrivelse af ordet og af videoen.

Oprettelse af en ny bruger

Ved at trykke på menulinket "Opret ny bruger" kommer følgende side.

Tegn til tiden

Bruger: Ikke logget på

Menu

- Søg
- Om denne side
- Alle forespørgsler
- Alle grupper
- Ny bruger
- Log på

Ny bruger

Navn:

Brugerkode:

Password:

Gentag password:

E-mail:

Gem

Udviklet af Jan Schrøder Hansen - Efteråret 2011

Denne side udfyldes.

Tegn til tiden

Bruger: Ikke logget på

Menu

- Søg
- Om denne side
- Alle forespørgsler
- Alle grupper
- Ny bruger
- Log på

Ny bruger

Navn:

Brugerkode:

Password:

Gentag password:

E-mail:

Gem

Udviklet af Jan Schrøder Hansen - Efteråret 2011

Ved tryk på "Gem" kommer følgende side.

Tegn til tiden

Bruger: Ikke logget på

Menu

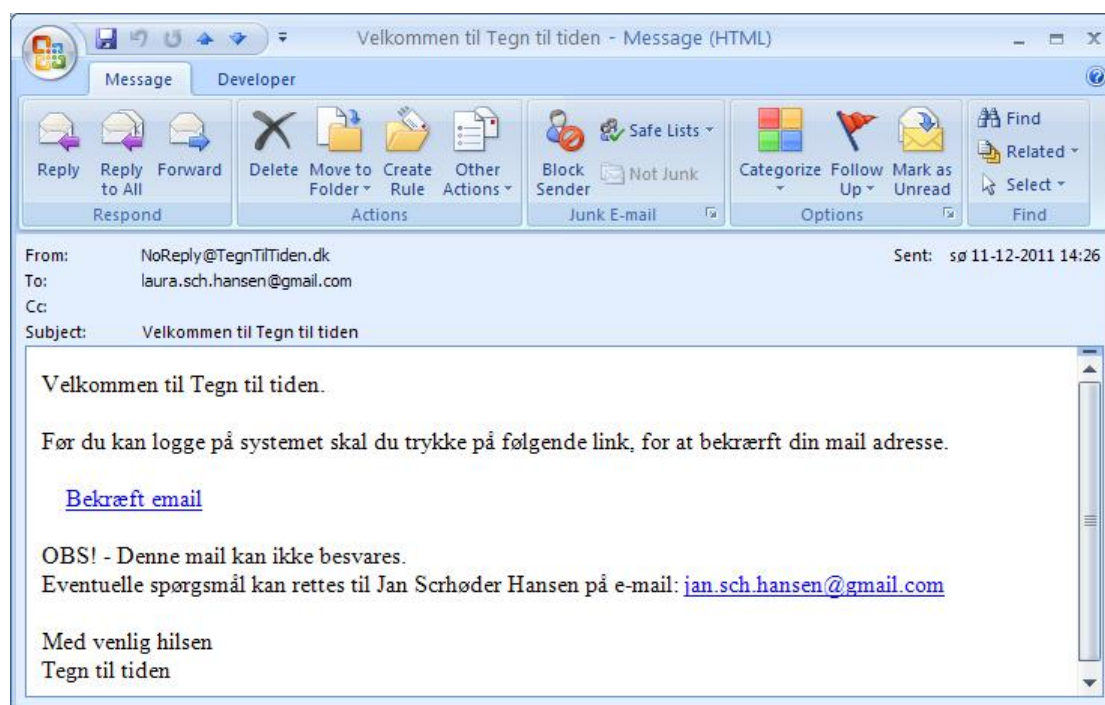
- Søg
- Om denne side
- Alle forespørgsler
- Alle grupper
- Ny bruger
- Log på

Bruger oprettet

Dine bruger oplysninger er nu gemt. Der er sendt en e-mail til den e-mail adresse du har angivet. I e-mailen er der et link, til dette system, som skal aktiveres før du kan logge på systemet første gang.

Udviklet af Jan Schrøder Hansen - Efteråret 2011

Som det fremgår af siden, har systemet sendt en mail til brugeren. Så e-mail adressen kan verificeres.



Her skal brugeren så trykke på "Bekræft email" linket, som viser følgende side.

Tegn til tiden

Bruger: Ikke logget på

Menu <ul style="list-style-type: none">• Søg• Om denne side• Alle forespørgsler• Alle grupper• Ny bruger• Log på	E-mail verificeret <p>Velkommen til Tegn til tiden. Din e-mail adress er nu godkendt. Og du kan nu logge på systemet</p>
--	---

Udviklet af Jan Schrøder Hansen - Efteråret 2011

Så er brugeren klar til at logge sig på systemet. Ved at trykke på "Log på" linket.

Tegn til tiden

Bruger: Ikke logget på

Menu <ul style="list-style-type: none">• Søg• Om denne side• Alle forespørgsler• Alle grupper• Ny bruger• Log på	Log på Brugerkode: <input type="text" value="lsh"/> Password: <input type="password" value="....."/> <input type="button" value="Login"/>
--	---

Udviklet af Jan Schrøder Hansen - Efteråret 2011

Brugeren trykker på "Login" knappen og kommer til Søge siden, men nu som indlogget bruger med flere valgmuligheder i Menu boksen.

Tegn til tiden

Bruger: Laura Schrøder Hansen - laura.sch.hansen@gmail.com


Menu <ul style="list-style-type: none">• Søg• Om denne side• Alle forespørgsler• Alle grupper• Mine forespørgsler• Mine grupper• Mine uploads• Ret brugeroplysninger• Log af	Søg Søg efter: Ord <input type="text"/> <input type="button" value="Søg"/> Resultat: <input type="text"/> << < > >>
---	--

Udviklet af Jan Schrøder Hansen - Efteråret 2011

"Om denne side" menulinket viser følgende side. Med en lille velkommen video, samt en kort beskrivelse af formålet med denne applikation.

Menu

- Søg
- Om denne side
- Alle forespørgsler
- Alle grupper
- Mine forespørgsler
- Mine grupper
- Mine uploads
- Ret brugeroplysninger
- Log af

Om denne side

Denne side er tænkt som en brugerdriven dansk tegnsprogsordbog. Dvs. at man som bruger, af denne side, kan spørge andre tegnsprogskyndige, om hjælp til ord man ikke selv kender.

Eventuelle spørgsmål kan rettes til Jan Schrøder Hansen via e-mail: jan.sch.hansen@gmail.com

Udviklet af Jan Schrøder Hansen - Efteråret 2011

"Alle forespørgsler" menulinket viser følgende side.

Tegn til tiden

Bruger: Laura Schrøder Hansen - laura.sch.hansen@gmail.com

Menu

- Søg
- Om denne side
- Alle forespørgsler
- Alle grupper
- Mine forespørgsler
- Mine grupper
- Mine uploads
- Ret brugeroplysninger
- Log af

Alle forespørgsler

Alle forespørgsler uden uploads

Danmark

Oprettet den 20/11-2011 20:33

Grupper: Ikke tilknyttet nogen gruppe. **Upload forslag**

Irak

Oprettet den 20/11-2011 20:33

Grupper: og Det arabiske forår. **Upload forslag**

Iran

Oprettet den 20/11-2011 20:32

Grupper: og Det arabiske forår. **Upload forslag**

Libyen

Oprettet den 19/11-2011 14:40

Grupper: Det arabiske forår og Nordafrika. **Upload forslag**

Syrien

Oprettet den 20/11-2011 20:32

Grupper: og Det arabiske forår. **Upload forslag**

<< < 1 > >>

Udviklet af Jan Schrøder Hansen - Efteråret 2011

Denne liste indeholder alle de ord som der mangler forslag til. Hvis brugeren trykker på et af "Upload forslag" linkene, kommer følgende side.

Tegn til tiden

Bruger: Laura Schrøder Hansen - laura.sch.hansen@gmail.com

Menu

- Søg
- Om denne side
- Alle forespørgsler
- Alle grupper
- Mine forespørgsler
- Mine grupper
- Mine uploads
- Ret brugeroplysninger
- Log af

Upload forslag til ord: Danmark

Tilbage

Vælg video fil:

Beskrivelse af video:

Udviklet af Jan Schrøder Hansen - Efteråret 2011

Her vælger brugeren en videofil vha. af "Browse..." knappen, som viser en alm. fildialog, hvor en fil kan vælges, brugeren udfylder beskrivelse feltet og trykker på "Upload og gem" knappen. Systemet henter filen og konvertere denne til et HTML5 videoformat og gemmer denne. Denne proces tager lidt tid. Hvis systemet ikke kan konvertere filen, kommer den en fejl besked.

Tegn til tiden

Bruger: Laura Schrøder Hansen - laura.sch.hansen@gmail.com

Menu

- Søg
- Om denne side
- Alle forespørgsler
- Alle grupper
- Mine forespørgsler
- Mine grupper
- Mine uploads
- Ret brugeroplysninger
- Log af

Upload forslag til ord: Danmark

Tilbage

Vælg video fil:

Beskrivelse af video:

Test

 Fejl ved konvertering af filen.

Udviklet af Jan Schrøder Hansen - Efteråret 2011

"Mine forespørgsler" menulinket viser følgende side.

Tegn til tiden

Bruger: Laura Schrøder Hansen - laura.sch.hansen@gmail.com

Menu

- Søg
- Om denne side
- Alle forespørgsler
- Alle grupper
- Mine forespørgsler
- Mine grupper
- Mine uploads
- Ret brugeroplysninger
- Log af

Mine forespørgsler

Opret ny forespørgsel

Mine forespørgsler

• Ingen forespørgelser.

<< < > >>

Udviklet af Jan Schrøder Hansen - Efteråret 2011

Her fremgår det at brugeren ikke har oprettet nogle forespørgsler endnu. Dette gøres ved at trykke på "Opret ny forespørgsel" linket, som viser følgende side.

Tegn til tiden

Bruger: Laura Schrøder Hansen - laura.sch.hansen@gmail.com

Menu <ul style="list-style-type: none">• Søg• Om denne side• Alle forespørgsler• Alle grupper• Mine forespørgsler• Mine grupper• Mine uploads• Ret brugeroplysninger• Log af	Opret forespørgsel <p>Ord der ønskes forslag til: <input type="text" value="Yemen"/></p> <p>Beskrivelse af ordet: <input type="text" value="Arabisk land."/></p> <p><input type="button" value="Gem"/> <input type="button" value="Gem og tilknyt grupper"/></p>
---	---

Udviklet af Jan Schrøder Hansen - Efteråret 2011

Her kan brugeren så ønske forslag til et ord, ved at udfylde de to felter og trykke på "Gem" knappen. "Gem og tilknyt grupper" er ikke implementeret endnu. Når der gemmes så vender systemet tilbage til "Mine forespørgsler"

Tegn til tiden

Bruger: Laura Schrøder Hansen - laura.sch.hansen@gmail.com

Menu <ul style="list-style-type: none">• Søg• Om denne side• Alle forespørgsler• Alle grupper• Mine forespørgsler• Mine grupper• Mine uploads• Ret brugeroplysninger• Log af	Mine forespørgsler <p>Opret ny forespørgsel</p> <p>Mine forespørgsler</p> <div><p>Yemen</p><p>Oprettet den 11/12-2011 15:09</p><p>Grupper: Ikke tilknyttet nogen gruppe. Ret</p></div> <p><< < 1 > >></p>
---	--

Udviklet af Jan Schrøder Hansen - Efteråret 2011

På listen er der et "Ret" link som ikke er implementeret endnu. Hvis brugeren trykker på ord linket, så vises følgende side. Hvor det fremgår at der ikke er nogen forslag endnu.

Tegn til tiden

Bruger: Laura Schrøder Hansen - laura.sch.hansen@gmail.com

Menu <ul style="list-style-type: none">• Søg• Om denne side• Alle forespørgsler• Alle grupper• Mine forespørgsler• Mine grupper• Mine uploads• Ret brugeroplysninger• Log af	Ord: Yemen <div>Beskrivelse Arabisk land.</div> <div>Upload forslag</div> <div>Forslag</div> <div>• Ingen forslag er uploadet.</div> <div><< < > >></div>
---	--

Udviklet af Jan Schrøder Hansen - Efteråret 2011

Hvis man forsøger at oprette et ord som findes i forvejen, så kommer følgende fejl.

Tegn til tiden

Bruger: Laura Schrøder Hansen - laura.sch.hansen@gmail.com

Menu <ul style="list-style-type: none">• Søg• Om denne side• Alle forespørgsler• Alle grupper• Mine forespørgsler• Mine grupper• Mine uploads• Ret brugeroplysninger• Log af	Opret forespørgsel <div>Ord der ønskes forslag til: <input type="text" value="Egypten"/></div> <div>Beskrivelse af ordet: <div>Test</div></div> <div><input type="button" value="Gem"/> <input type="button" value="Gem og tilknyt grupper"/></div> <div> Ordet findes i forvejen.</div>
---	--

Udviklet af Jan Schrøder Hansen - Efteråret 2011

Til slut kan brugeren logge af ved at trykke på menulinket "Log af", og følgende side vises, hvor det fremgår at brugeren ikke er logget på mere.

Tegn til tiden

Bruger: Ikke logget på

Menu <ul style="list-style-type: none">• Søg• Om denne side• Alle forespørgsler• Alle grupper• Ny bruger• Log på	Søg <div>Søg efter: <input type="text" value="Ord"/> <input type="button" value="Søg"/></div> <div>Resultat</div> <div><< < > >></div>
--	---

Udviklet af Jan Schrøder Hansen - Efteråret 2011

9.3. UDVIKLINGSMILJØ

Under udviklingen har jeg brugt følgende værktøjer, programmer og services:

- NetBeans³⁸ 7.0.1 – Java udviklingsmiljø, i dette miljø kan man starte følgende servere, som jeg har benyttet mig af:
 - JavaDB⁴ server.
 - Apache Tomcat⁵ 7 server.
- Apache James¹⁶ som er en mailserver. Denne har jeg sat op til, at sende alle e-mails til den samme modtager, som er min arbejds e-mail.
- Microsoft Outlook, mail program, for at læse de e-mails, systemet genererer. Set op til at læse mails fra Apache James.
- Apache SubVersion³⁹, kode versionskontrol system, som jeg bruger sammen med Google's GoogleCode⁴⁰ service. Så jeg har alle mine kodefiler, dokumenter m.m. på GoogleCode. Se code.google.com/p/user-driven-sign-language-dictionary. Dvs. alt hvad jeg har lavet ligger lokalt på en pc, og på førnævnte GoggleCode side.

9.4. KODE

Koden vises pakkevis i følgende rækkefølge:

- dk.jsh.itdiplom.userdrivensignlanguagedictionary.entity
- dk.jsh.itdiplom.userdrivensignlanguagedictionary.business
- dk.jsh.itdiplom.userdrivensignlanguagedictionary.wicket
- dk.jsh.itdiplom.userdrivensignlanguagedictionary.util

Til sidst følger nogle enkelte Hibernate setup, SQL DDL og Apache Ant⁴¹ scripts filer. HTML filer findes i wicket pakken.

Pakken: dk.jsh.itdiplom.userdrvensignlanguagedictionary.entity

Filen: ApplicationUser.java

```
1 package dk.jsh.itdiplom.userdrvensignlanguagedictionary.entity;
2
3 import dk.jsh.itdiplom.userdrvensignlanguagedictionary.entity.Constants.UserRole;
4 import java.io.Serializable;
5 import java.util.Date;
6 import javax.persistence.*;
7
8 /**
9  * Application user entity class.
10  *
11  * @author Jan S. Hansen
12  */
13 @Entity
14 public class ApplicationUser implements Serializable {
15     private static final long serialVersionUID = 1L;
16
17     @Id
18     @GeneratedValue(strategy = GenerationType.IDENTITY)
19     protected Long id;
20     @Version
21     @Column(nullable = false)
22     protected Integer version;
23     @Column(length=20, nullable = false, unique=true)
24     protected String login;
25     @Column(length=20, nullable = false)
26     protected String password;
27     @Column(length=50, nullable = false)
28     protected String fullname;
29     @Enumerated(EnumType.STRING)
30     @Column(length=50, nullable = false)
31     protected String email;
32     @Column(nullable = true)
33     @Temporal(javax.persistence.TemporalType.TIMESTAMP)
34     protected Date emailVerificationSent;
35     @Column(nullable = true)
36     @Temporal(javax.persistence.TemporalType.TIMESTAMP)
37     protected Date emailVerified;
38     @Enumerated(EnumType.STRING)
39     @Column(length=10, nullable = false)
40     protected UserRole userRole;
41
42     public ApplicationUser() {
43     }
44
45     public ApplicationUser(String login, String password, String fullname,
46         String email, Date emailVerificationSent,
47         Date emailVerified, UserRole userRole) {
48         this.login = login;
49         this.password = password;
50         this.fullname = fullname;
51         this.email = email;
52         this.emailVerificationSent = emailVerificationSent;
53         this.emailVerified = emailVerified;
54         this.userRole = userRole;
55     }
56
57     public Long getId() {
58         return id;
59     }
60
61     public void setId(Long id) {
62         this.id = id;
63     }
64
65     public Integer getVersion() {
66         return version;
67     }
68
69     public void setVersion(Integer version) {
70         this.version = version;
```



```

71 }
72
73 public String getFullName() {
74     return fullname;
75 }
76
77 public void setFullName(String fullname) {
78     this.fullname = fullname;
79 }
80
81 public String getLogin() {
82     return login;
83 }
84
85 public void setLogin(String login) {
86     this.login = login;
87 }
88
89 public String getEmail() {
90     return email;
91 }
92
93 public void setEmail(String email) {
94     this.email = email;
95 }
96
97 public String getPassword() {
98     return password;
99 }
100
101 public void setPassword(String password) {
102     this.password = password;
103 }
104
105 public Date getEmailVerificationSent() {
106     return emailVerificationSent;
107 }
108
109 public void setEmailVerificationSent(Date emailVerificationSent) {
110     this.emailVerificationSent = emailVerificationSent;
111 }
112
113 public Date getEmailVerified() {
114     return emailVerified;
115 }
116
117 public void setEmailVerified(Date emailVerified) {
118     this.emailVerified = emailVerified;
119 }
120
121 public UserRole getUserRole() {
122     return userRole;
123 }
124
125 public void setUserRole(UserRole userRole) {
126     this.userRole = userRole;
127 }
128 }
129

```

File: Constants.java

```

1 package dk.jsh.itdiplom.userdrivensignlanguagedictionary.entity;
2 import java.util.ArrayList;
3 import java.util.List;
4
5 /**
6  * Constants and enums.
7  *
8  * @author Jan S. Hansen
9  */
10 public class Constants {
11     private Constants() {}

```

```

12
13 /**
14  * BaRI user role: ADMIN, DEVELOPER, NORMAL.
15  */
16 public enum UserRole {
17     ADMIN("Administrator"),
18     NORMAL("Alm. bruger");
19
20     private String name;
21
22     UserRole(String name) {
23         this.name = name;
24     }
25
26     public String getName() {
27         return name;
28     }
29
30     public static List<String> getNames() {
31         List<String> names = new ArrayList<String>();
32         names.add(ADMIN.name);
33         names.add(NORMAL.name);
34         return names;
35     }
36
37     public static UserRole getName(String name) {
38         if (ADMIN.name.equals(name)) {
39             return ADMIN;
40         }
41         return NORMAL;
42     }
43 }
44 }
45

```

Filen: Video.java

```

1 package dk.jsh.itdiplom.userdrivensignlanguagedictionary.entity;
2
3 import java.io.Serializable;
4 import java.util.Date;
5 import javax.persistence.*;
6
7 /**
8  * File entity class.
9  *
10  * @author Jan S. Hansen
11  */
12 @Entity
13 public class VideoFile implements Serializable {
14     private static final long serialVersionUID = 1L;
15
16     @Id
17     @GeneratedValue(strategy = GenerationType.IDENTITY)
18     protected Long id;
19     @Version
20     @Column(nullable = false)
21     protected Integer version;
22     @Column(length=100, nullable = false)
23     protected String fileName;
24     @Column(length=250, nullable = true)
25     protected String description;
26     @Column(length=50, nullable = false)
27     protected String resourceName;
28     @Column(nullable = false)
29     @Temporal(javax.persistence.TemporalType.TIMESTAMP)
30     protected Date uploadedDateTime;
31     @ManyToOne(optional=false)
32     @org.hibernate.annotations.ForeignKey(name="fk_file_applicationuser")
33     protected ApplicationUser uploadedBy;
34     @ManyToOne(optional=false)
35     @org.hibernate.annotations.ForeignKey(name="fk_file_word")
36     protected Word toWord;
37

```

```

38 public VideoFile() {
39 };
40
41 public VideoFile(String fileName, String description, String resourceName,
42     Date uploadedDateTime,
43     ApplicationUser uploadedBy, Word toWord) {
44     this.fileName = fileName;
45     this.description = description;
46     this.resourceName = resourceName;
47     this.uploadedDateTime = uploadedDateTime;
48     this.uploadedBy = uploadedBy;
49     this.toWord = toWord;
50 }
51
52 public Long getId() {
53     return id;
54 }
55
56 public void setId(Long id) {
57     this.id = id;
58 }
59
60 public Integer getVersion() {
61     return version;
62 }
63
64 public void setVersion(Integer version) {
65     this.version = version;
66 }
67
68 public String getFileName() {
69     return fileName;
70 }
71
72 public void setFileName(String fileName) {
73     this.fileName = fileName;
74 }
75
76 public String getDescription() {
77     return description;
78 }
79
80 public void setDescription(String description) {
81     this.description = description;
82 }
83
84 public String getResourceName() {
85     return resourceName;
86 }
87
88 public void setResourceName(String resourceName) {
89     this.resourceName = resourceName;
90 }
91
92 public Word getToWord() {
93     return toWord;
94 }
95
96 public void setToWord(Word toWord) {
97     this.toWord = toWord;
98 }
99
100 public ApplicationUser getUploadedBy() {
101     return uploadedBy;
102 }
103
104 public void setUploadedBy(ApplicationUser uploadedBy) {
105     this.uploadedBy = uploadedBy;
106 }
107
108 public Date getUploadedDateTime() {
109     return uploadedDateTime;
110 }
111
112 public void setUploadedDateTime(Date uploadedDateTime) {

```

```

113     this.uploadedDateTime = uploadedDateTime;
114 }
115 }
116

```

Filen: Word.java

```

1 package dk.jsh.itdiplom.userdrivensignlanguagedictionary.entity;
2
3 import java.io.Serializable;
4 import java.util.ArrayList;
5 import java.util.Collections;
6 import java.util.Date;
7 import java.util.List;
8 import javax.persistence.*;
9
10 /**
11  * Word entity class.
12  *
13  * @author Jan S. Hansen
14  */
15 @Entity
16 public class Word implements Serializable {
17     private static final long serialVersionUID = 1L;
18
19     @Id
20     @GeneratedValue(strategy = GenerationType.IDENTITY)
21     protected Long id;
22     @Version
23     @Column(nullable = false)
24     protected Integer version;
25     @Column(length=50, nullable = false, unique=true)
26     protected String word;
27     @Column(length=250, nullable = true)
28     protected String description;
29     @Column(nullable = false)
30     @Temporal(javax.persistence.TemporalType.TIMESTAMP)
31     protected Date createdDateTime;
32     @ManyToOne(optional=false)
33     @org.hibernate.annotations.ForeignKey(name="fk_word_applicationuser")
34     protected ApplicationUser requestCreatedBy;
35     @ManyToMany(fetch = FetchType.EAGER)
36     @OrderBy("name")
37     @JoinTable(
38         name="WordGroupWordRelation",
39         joinColumns={ @JoinColumn(name="WORD_ID")},
40         inverseJoinColumns={ @JoinColumn(name="WORDGROUP_ID")}
41     )
42     protected List<WordGroup> wordGroups;
43
44     public Word() {
45     }
46
47     public Word(String word, String description, Date createdDateTime,
48         ApplicationUser requestCreatedBy) {
49         this.word = word;
50         this.description = description;
51         this.createdDateTime = createdDateTime;
52         this.requestCreatedBy = requestCreatedBy;
53     }
54
55     public Long getId() {
56         return id;
57     }
58
59     public void setId(Long id) {
60         this.id = id;
61     }
62
63     public Integer getVersion() {
64         return version;
65     }
66

```

```

67 public void setVersion(Integer version) {
68     this.version = version;
69 }
70
71 public String getWord() {
72     return word;
73 }
74
75 public void setWord(String word) {
76     this.word = word;
77 }
78
79 public String getDescription() {
80     return description;
81 }
82
83 public void setDescription(String description) {
84     this.description = description;
85 }
86
87 public ApplicationUser getRequestCreatedBy() {
88     return requestCreatedBy;
89 }
90
91 public void setRequestCreatedBy(ApplicationUser requestCreatedBy) {
92     this.requestCreatedBy = requestCreatedBy;
93 }
94
95 public Date getCreatedDateTime() {
96     return createdDateTime;
97 }
98
99 public void setCreatedDateTime(Date createdDateTime) {
100     this.createdDateTime = createdDateTime;
101 }
102
103 public List<WordGroup> getWordGroups() {
104     return wordGroups;
105 }
106
107 public List<String> getSortedWordGroups() {
108     List<String> list = new ArrayList<String>();
109     for (WordGroup wordGroup: wordGroups) {
110         list.add(wordGroup.name);
111     }
112     Collections.sort(list);
113     return list;
114 }
115 }
116

```

File: WordGroup.java

```

1 package dk.jsh.itdiplom.userdrivensignlanguagedictionary.entity;
2
3 import java.io.Serializable;
4 import java.util.Date;
5 import javax.persistence.*;
6
7 /**
8  * Word group entity class.
9  *
10  * @author Jan S. Hansen
11  */
12 @Entity
13 public class WordGroup implements Serializable{
14     private static final long serialVersionUID = 1L;
15
16     @Id
17     @GeneratedValue(strategy = GenerationType.IDENTITY)
18     protected Long id;
19     @Version
20     @Column(nullable = false)

```

```

21 protected Integer version;
22 @Column(length=30, nullable = false, unique=true)
23 protected String name;
24 @Column(length=250, nullable = true)
25 protected String description;
26 @Column(nullable = false)
27 @Temporal(javax.persistence.TemporalType.TIMESTAMP)
28 protected Date createdDateTime;
29 @ManyToOne(optional=false)
30 @org.hibernate.annotations.ForeignKey(name="fk_wordgroup_applicationuser")
31 protected ApplicationUser createdBy;
32
33 public WordGroup() {
34 }
35
36 public WordGroup(String name, String description, Date createdDateTime,
37     ApplicationUser createdBy) {
38     this.name = name;
39     this.description = description;
40     this.createdDateTime = createdDateTime;
41     this.createdBy = createdBy;
42 }
43
44 public Long getId() {
45     return id;
46 }
47
48 public void setId(Long id) {
49     this.id = id;
50 }
51
52 public Integer getVersion() {
53     return version;
54 }
55
56 public void setVersion(Integer version) {
57     this.version = version;
58 }
59
60 public String getName() {
61     return name;
62 }
63
64 public void setName(String name) {
65     this.name = name;
66 }
67
68 public String getDescription() {
69     return description;
70 }
71
72 public void setDescription(String description) {
73     this.description = description;
74 }
75
76 public ApplicationUser getCreatedBy() {
77     return createdBy;
78 }
79
80 public void setCreatedBy(ApplicationUser createdBy) {
81     this.createdBy = createdBy;
82 }
83
84 public Date getCreatedDateTime() {
85     return createdDateTime;
86 }
87
88 public void setCreatedDateTime(Date createdDateTime) {
89     this.createdDateTime = createdDateTime;
90 }
91 }
92

```

File: WordGroupWordRelation.java

```

1 package dk.jsh.itdiplom.userdrivensignlanguagedictionary.entity;
2
3 import java.io.Serializable;
4 import javax.persistence.*;
5
6 /**
7  * Word group/Word many to many relation.
8  *
9  * @author Jan S. Hansen
10 */
11 @Entity
12 @Table(name="WordGroupWordRelation",
13     uniqueConstraints = {
14         @UniqueConstraint(columnNames={ "wordGroup_id", "word_id"})
15     })
16 public class WordGroupWordRelation implements Serializable {
17     private static final long serialVersionUID = 1L;
18
19     @Id
20     @GeneratedValue(strategy = GenerationType.IDENTITY)
21     protected Long id;
22     @Version
23     @Column(nullable = false)
24     protected Integer version;
25     @ManyToOne(optional=false)
26     @org.hibernate.annotations.ForeignKey(name="fk_wordgroupwordrelation_wordgroup")
27     protected WordGroup wordGroup;
28     @ManyToOne(optional=false)
29     @org.hibernate.annotations.ForeignKey(name="fk_wordgroupwordrelation_word")
30     protected Word word;
31
32     public WordGroupWordRelation() {
33     }
34
35     public WordGroupWordRelation(WordGroup wordGroup, Word word) {
36         this.wordGroup = wordGroup;
37         this.word = word;
38     }
39
40     public Long getId() {
41         return id;
42     }
43
44     public void setId(Long id) {
45         this.id = id;
46     }
47
48     public Integer getVersion() {
49         return version;
50     }
51
52     public void setVersion(Integer version) {
53         this.version = version;
54     }
55
56     public WordGroup getWordGroup() {
57         return wordGroup;
58     }
59
60     public void setWordGroup(WordGroup wordGroup) {
61         this.wordGroup = wordGroup;
62     }
63
64     public Word getWord() {
65         return word;
66     }
67
68     public void setWord(Word word) {
69         this.word = word;
70     }
71 }
72

```

Pakken: dk.jsh.itdiplom.userdrivensignlanguagedictionary.business

Filen: ApplicationUserBusiness.java

```
1 package dk.jsh.itdiplom.userdrivensignlanguagedictionary.business;
2
3 import dk.jsh.itdiplom.userdrivensignlanguagedictionary.entity.ApplicationUser;
4 import dk.jsh.itdiplom.userdrivensignlanguagedictionary.util.HibernateUtil;
5 import java.util.Date;
6 import java.util.List;
7 import org.hibernate.Query;
8 import org.hibernate.Session;
9 import org.hibernate.Transaction;
10
11 /**
12  * Business methods for ApplicationUser.
13  *
14  * @author Jan S. Hansen
15  */
16 public class ApplicationUserBusiness {
17
18     private ApplicationUserBusiness();
19
20     /**
21      * Gets a applicationUser from login and password.
22      *
23      * @param login user login
24      * @param password password
25      * @return a ApplicationUser or null if login or password is wrong.
26      */
27     public static ApplicationUser isValidUser(String login, String password) {
28         ApplicationUser appUser = null;
29         Session session = HibernateUtil.getSessionFactory().openSession();
30         String hql = "select appUser from "
31             + "dk.jsh.itdiplom.userdrivensignlanguagedictionary.entity."
32             + "ApplicationUser appUser "
33             + "where appUser.login = :login "
34             + "and appUser.password = :password "
35             + "and appUser.emailVerified is not null";
36         Query query = session.createQuery(hql);
37         query.setString("login", login);
38         query.setString("password", login);
39         List<ApplicationUser> appUsers = query.list();
40         if (appUsers.size() == 1) {
41             appUser = appUsers.get(0);
42         }
43         else if (appUsers.size() > 1) {
44             throw new RuntimeException("More than one user with login " +
45                 login);
46         }
47         session.close();
48         return appUser;
49     }
50
51     /**
52      * Persist a new Application user.
53      *
54      * @param newUser a new ApplicationUser
55      */
56     public static void saveNew(ApplicationUser newUser) {
57         Session session = HibernateUtil.getSessionFactory().openSession();
58         Transaction tx = session.beginTransaction();
59         session.save(newUser);
60         tx.commit();
61         session.close();
62     }
63
64     /**
65      * Test if a user login is in use.
66      *
67      * @param login user login to test
68      * @return true is user login is in use.
69      */
70     public static boolean isUserLoginInUse(String login) {
```



```

71     boolean inUse = false;
72
73     Session session = HibernateUtil.getSessionFactory().openSession();
74     String hql = "select appUser "
75         + "from dk.jsh.itdiplom.userdrivensignlanguagedictionary.entity."
76         + "ApplicationUser appUser "
77         + "where appUser.login = :login";
78     Query query = session.createQuery(hql);
79     query.setString("login", login);
80     if (query.list().isEmpty()) {
81         inUse = false;
82     }
83     else {
84         inUse = true;
85     };
86     session.close();
87     return inUse;
88 }
89
90 /**
91  * Set e-mail verified to true.
92  *
93  * @param login user login
94  */
95 public static void setEmailVerified(String login) {
96     Session session = HibernateUtil.getSessionFactory().openSession();
97     String hql = "select appUser "
98         + "from dk.jsh.itdiplom.userdrivensignlanguagedictionary.entity."
99         + "ApplicationUser appUser "
100        + "where appUser.login = :login";
101     Query query = session.createQuery(hql);
102     query.setString("login", login);
103     ApplicationUser user = (ApplicationUser)query.list().get(0);
104     user.setEmailVerified(new Date());
105     Transaction tx = session.beginTransaction();
106     session.save(user);
107     tx.commit();
108     session.close();
109 }
110 }
111

```

File: VideoFileBusiness.java

```

1 package dk.jsh.itdiplom.userdrivensignlanguagedictionary.business;
2
3 import dk.jsh.itdiplom.userdrivensignlanguagedictionary.entity.VideoFile;
4 import dk.jsh.itdiplom.userdrivensignlanguagedictionary.entity.Word;
5 import dk.jsh.itdiplom.userdrivensignlanguagedictionary.util.HibernateUtil;
6 import java.util.ArrayList;
7 import java.util.List;
8 import org.hibernate.Query;
9 import org.hibernate.Session;
10 import org.hibernate.Transaction;
11
12 /**
13  * Business methods for VideoFile.
14  *
15  * @author Jan S. Hansen
16  */
17 public class VideoFileBusiness {
18     private VideoFileBusiness();
19
20     /**
21      * Persist a new video file
22      *
23      * @param newWord a new video file
24      */
25     public static void saveNew(VideoFile newFile) {
26         Session session = HibernateUtil.getSessionFactory().openSession();
27         Transaction tx = session.beginTransaction();
28         session.save(newFile);
29         tx.commit();

```

```

30     session.close();
31 }
32
33 /**
34  * Get all video files for a word.
35  *
36  * @return A list of vidoe files
37  */
38 public static List<VideoFile> getAllVideoFilesForAWord(Word word) {
39     List<VideoFile> videoFileList = new ArrayList<VideoFile>();
40     Session session = HibernateUtil.getSessionFactory().openSession();
41     String hql =
42         "select videofile from "
43         + "dk.jsh.itdiplom.userdrivensignlanguagedictionary.entity."
44         + "VideoFile videofile "
45         + "where videofile.toWord.id = :wordid "
46         + "order by videofile.uploadedDateTime desc";
47     Query query = session.createQuery(hql);
48     query.setLong("wordid", word.getId());
49     videoFileList = query.list();
50     session.close();
51     return videoFileList;
52 }
53 }
54

```

Filen: WordBusiness.java

```

1 package dk.jsh.itdiplom.userdrivensignlanguagedictionary.business;
2
3 import dk.jsh.itdiplom.userdrivensignlanguagedictionary.entity.ApplicationUser;
4 import dk.jsh.itdiplom.userdrivensignlanguagedictionary.entity.Word;
5 import dk.jsh.itdiplom.userdrivensignlanguagedictionary.entity.WordGroup;
6 import dk.jsh.itdiplom.userdrivensignlanguagedictionary.util.HibernateUtil;
7 import java.util.ArrayList;
8 import java.util.List;
9 import org.hibernate.Query;
10 import org.hibernate.Session;
11 import org.hibernate.Transaction;
12
13 /**
14  * Business metods for Word.
15  *
16  * @author Jan S. Hansen
17  */
18 public class WordBusiness {
19     private WordBusiness();
20
21     /**
22      * Persist a new Word.
23      *
24      * @param newWord a new Word
25      */
26     public static void saveNew(Word newWord) {
27         Session session = HibernateUtil.getSessionFactory().openSession();
28         Transaction tx = session.beginTransaction();
29         session.save(newWord);
30         tx.commit();
31         session.close();
32     }
33
34     /**
35      * Test if a word exists.
36      *
37      * @param word word to test
38      * @return true is the word exists
39      */
40     public static boolean isWordInUse(String word) {
41         boolean inUse = false;
42
43         Session session = HibernateUtil.getSessionFactory().openSession();
44         String hql = "select word "
45             + "from dk.jsh.itdiplom.userdrivensignlanguagedictionary.entity."

```

```

46         + "Word word "
47         + "where word.word = :word";
48     Query query = session.createQuery(hql);
49     query.setString("word", word);
50     if (query.list().isEmpty()) {
51         inUse = false;
52     }
53     else {
54         inUse = true;
55     };
56     session.close();
57     return inUse;
58 }
59
60 /**
61  * Get all words created by a specific user.
62  *
63  * @return A list of Word.
64  */
65 public static List<Word> getAllWordsCreatedByUser(ApplicationUser user) {
66     List<Word> wordList = new ArrayList<Word>();
67     Session session = HibernateUtil.getSessionFactory().openSession();
68     String hql =
69         "select word from "
70         + "dk.jsh.itdiplom.userdrivensignlanguagedictionary.entity."
71         + "Word word "
72         + "where word.requestCreatedBy.id = :userid "
73         + "order by word.word";
74     Query query = session.createQuery(hql);
75     query.setLong("userid", user.getId());
76     wordList = query.list();
77     session.close();
78     return wordList;
79 }
80
81 /**
82  * Search for words.
83  *
84  * @param search search string
85  * @return a list of words that match the search criteria
86  */
87 public static List<Word> search(String search) {
88     List<Word> wordList = new ArrayList<Word>();
89     search = search.toLowerCase();
90     search = search.replace(" ", "%");
91     search = search.replace("?", "_");
92     boolean useLike = false;
93     if (search.indexOf("%") != -1 || search.indexOf("_") != -1) {
94         useLike = true;
95     }
96     Session session = HibernateUtil.getSessionFactory().openSession();
97     StringBuilder hql = new StringBuilder();
98     hql.append("select word from ");
99     hql.append("dk.jsh.itdiplom.userdrivensignlanguagedictionary.entity.");
100    hql.append("Word word ");
101    if (useLike) {
102        hql.append("where lower(word.word) like :search ");
103    }
104    else {
105        hql.append("where lower(word.word) = :search ");
106    }
107    hql.append("order by word.word");
108    Query query = session.createQuery(hql.toString());
109    query.setString("search", search);
110    wordList = query.list();
111    session.close();
112    return wordList;
113 }
114
115 /**
116  * Search for words with specific groups.
117  *
118  * @param search search string
119  * @return a list of words that match the search criteria
120  */

```

```

121 public static List<Word> search(List<WordGroup> wordGroups) {
122     List<Word> wordList = new ArrayList<Word>();
123     Session session = HibernateUtil.getSessionFactory().openSession();
124     StringBuilder hql = new StringBuilder();
125     hql.append("select word from ");
126     hql.append("dk.jsh.itdiplom.userdrivesignlanguagedictionary.entity.");
127     hql.append("Word word ");
128     hql.append("join word.wordGroups wordGroup ");
129     hql.append("where wordGroup.id in (");
130     boolean first = true;
131     for (WordGroup wordGroup : wordGroups) {
132         if (first) {
133             first = false;
134         }
135         else {
136             hql.append(", ");
137         }
138         hql.append(wordGroup.getId());
139     }
140     hql.append(" ");
141     hql.append("order by word.word");
142     Query query = session.createQuery(hql.toString());
143     wordList = query.list();
144     session.close();
145     return wordList;
146 }
147
148 /**
149  * Get all words without uploads
150  *
151  * @return a list of all words without uploads
152  */
153 public static List<Word> getAllWordsWithoutUploads() {
154     List<Word> wordList = new ArrayList<Word>();
155     Session session = HibernateUtil.getSessionFactory().openSession();
156     StringBuilder hql = new StringBuilder();
157     hql.append("select word from ");
158     hql.append("dk.jsh.itdiplom.userdrivesignlanguagedictionary.entity.");
159     hql.append("Word word ");
160     hql.append("where word.id not in (");
161     hql.append("select distinct (videoFile.toWord.id) from ");
162     hql.append("dk.jsh.itdiplom.userdrivesignlanguagedictionary.entity.");
163     hql.append("VideoFile videoFile ");
164     hql.append(") ");
165     hql.append("order by word.word");
166     Query query = session.createQuery(hql.toString());
167     wordList = query.list();
168     session.close();
169     return wordList;
170 }
171 }
172

```

File: WordGroupBusiness.java

```

1 package dk.jsh.itdiplom.userdrivesignlanguagedictionary.business;
2
3 import dk.jsh.itdiplom.userdrivesignlanguagedictionary.entity.WordGroup;
4 import dk.jsh.itdiplom.userdrivesignlanguagedictionary.util.HibernateUtil;
5 import java.util.ArrayList;
6 import java.util.List;
7 import org.hibernate.Query;
8 import org.hibernate.Session;
9
10 /**
11  * Business methods for word groups.
12  *
13  * @author Jan S. Hansen
14  */
15 public class WordGroupBusiness {
16     private WordGroupBusiness() {}
17
18     /**

```

```

19  * Get all word groups.
20  *
21  * @return A list of WordGroup.
22  */
23  public static List<WordGroup> getAllWordGroups() {
24      List<WordGroup> wordGroupList = new ArrayList<WordGroup>();
25      Session session = HibernateUtil.getSessionFactory().openSession();
26      String hql =
27          "select wordGroup from "
28          + "dk.jsh.itdiplom.userdrivensignlanguagedictionary.entity."
29          + "WordGroup wordGroup "
30          + "order by wordGroup.name";
31      Query query = session.createQuery(hql);
32      wordGroupList = query.list();
33      session.close();
34      return wordGroupList;
35  }
36
37  /**
38   * Search for groups.
39   *
40   * @param search search string
41   * @return a list of word groups that match the search criteria
42   */
43  public static List<WordGroup> search(String search) {
44      List<WordGroup> wordGroupList = new ArrayList<WordGroup>();
45      search = search.toLowerCase();
46      search = search.replace(" ", "%");
47      search = search.replace("?", "_");
48      boolean useLike = false;
49      if (search.indexOf("%") != -1 || search.indexOf("_") != -1) {
50          useLike = true;
51      }
52      Session session = HibernateUtil.getSessionFactory().openSession();
53      StringBuilder hql = new StringBuilder();
54      hql.append("select wordGroup from ");
55      hql.append("dk.jsh.itdiplom.userdrivensignlanguagedictionary.entity.");
56      hql.append("WordGroup wordGroup ");
57      if (useLike) {
58          hql.append("where lower(wordGroup.name) like :search ");
59      }
60      else {
61          hql.append("where lower(wordGroup.name) = :search ");
62      }
63      hql.append("order by wordGroup.name");
64      Query query = session.createQuery(hql.toString());
65      query.setString("search", search);
66      wordGroupList = query.list();
67      session.close();
68      return wordGroupList;
69  }
70 }
71

```

Pakken: dk.jsh.itdiplom.userdrivensignlanguagedictionary.wicket

Filen: Application.java

```
1 package dk.jsh.itdiplom.userdrivensignlanguagedictionary.wicket;
2
3 import dk.jsh.itdiplom.userdrivensignlanguagedictionary.wicket.homepage.HomePage;
4 import java.util.logging.Logger;
5 import org.apache.wicket.Request;
6 import org.apache.wicket.Response;
7 import org.apache.wicket.Session;
8 import org.apache.wicket.protocol.http.WebApplication;
9
10
11 /**
12  * User driven sign language dictionary wicket application.
13  *
14  * @author Jan S. Hansen
15  */
16 public class Application extends WebApplication {
17     static final Logger logger = Logger.getLogger(Application.class.getName());
18
19     public Application() {
20         logger.info("Application started");
21     }
22
23     @Override
24     public Class getHomePage() {
25         return HomePage.class;
26     }
27
28     @Override
29     public Session newSession(Request request, Response response) {
30         return new WicketSession(request);
31     }
32 }
33
```

Filen: BasePage.html

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-
transitional.dtd">
3 <html xmlns:wicket="http://wicket.apache.org">
4 <head>
5     <meta charset="UTF-8">
6     <meta name="description" content="User driven sign language dictionary" />
7     <title>Tegn til tiden </title>
8     <wicket:head>
9         <wicket:link>
10             <link rel="stylesheet" type="text/css" href="style.css"/>
11         </wicket:link>
12     </wicket:head>
13 </head>
14 <body>
15     <table align="center" width="800px">
16         <tr>
17             <td>
18                 <header wicket:id="headerpanel" />
19             </td>
20         </tr>
21         <tr>
22             <td>
23                 <section class="content_container">
24                     <wicket:child/>
25                 </section>
26             </td>
27         </tr>
28         <tr>
29             <td>
30                 <footer wicket:id="footerpanel" />
31             </td>
32         </tr>
33     </table>
34 </body>
35 </html>
```

```

32     </tr>
33 </table>
34 </body>
35 </html>

```

File: BasePage.java

```

1 package dk.jsh.itdiplom.userdrivesignlanguagedictionary.wicket;
2
3 import dk.jsh.itdiplom.userdrivesignlanguagedictionary.entity.ApplicationUser;
4 import java.text.SimpleDateFormat;
5 import org.apache.wicket.markup.html.WebPage;
6
7 /**
8  * Abstract base page with header and footer panel.
9  *
10 * @author Jan S. Hansen
11 */
12 public abstract class BasePage extends WebPage {
13     protected final static SimpleDateFormat standardDateTimeFormat =
14         new SimpleDateFormat("dd/MM-yyyy HH:mm");
15
16     public BasePage() {
17         super();
18         WicketSession session = WicketSession.get();
19         ApplicationUser appUser = null;
20         if (session.isAuthenticated()) {
21             appUser = session.getApplicationUser();
22         }
23         add(new HeaderPanel("headerpanel", appUser));
24         add(new FooterPanel("footerpanel", "Udviklet af Jan Schrøder Hansen - "
25             + "Efteråret 2011"));
26     }
27 }

```

File: HeaderPanel.html

```

1 <?xml version="1.0" encoding="UTF-8"?>
2 <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-
transitional.dtd">
3 <html xmlns:wicket="http://wicket.apache.org">
4     <head><title>Tegn til tiden</title></head>
5     <body>
6         <wicket:panel>
7             <h1>Tegn til tiden</h1>
8             <p align="right" id="titleblock">
9                 <span wicket:id="userName">User Name</span>
10            </p>
11        </wicket:panel>
12    </body>
13 </html>

```

File: HeaderPanel.java

```

1 package dk.jsh.itdiplom.userdrivesignlanguagedictionary.wicket;
2
3 import dk.jsh.itdiplom.userdrivesignlanguagedictionary.entity.ApplicationUser;
4 import org.apache.wicket.markup.html.basic.Label;
5 import org.apache.wicket.markup.html.panel.Panel;
6
7 /**
8  * Header panel.
9  *
10 * @author Jan S. Hansen
11 */
12 public class HeaderPanel extends Panel {
13
14     /**
15      * Constructor

```

```

16  *
17  * @param appUser Application user
18  * @param exampleTitle title of the example
19  */
20  public HeaderPanel(String componentName, ApplicationUser appUser)
21  {
22      super(componentName);
23      StringBuilder text = new StringBuilder("Bruger: ");
24      if (appUser != null) {
25          text.append(appUser.getFullname());
26          text.append(" - ");
27          text.append(appUser.getEmail());
28      }
29      else {
30          text.append("Ikke logget på");
31      }
32      add(new Label("userName", text.toString()));
33  }
34 }

```

File: FooterPanel.html

```

1 <?xml version="1.0" encoding="UTF-8"?>
2 <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-
transitional.dtd">
3 <html xmlns:wicket="http://wicket.apache.org">
4 <body>
5 <wicket:panel>
6 <span wicket:id="footerpanel_text">This gets replaced</span>
7 </wicket:panel>
8 </body>
9 </html>

```

File: FooterPanel.java

```

1 package dk.jsh.itdiplom.userdrivesignlanguagedictionary.wicket;
2
3 import org.apache.wicket.markup.html.basic.Label;
4 import org.apache.wicket.markup.html.panel.Panel;
5
6 /**
7  * Footer panel.
8  *
9  * @author Jan S. Hansen
10 */
11 public final class FooterPanel extends Panel {
12
13     public FooterPanel(String id, String text) {
14         super(id);
15         add(new Label("footerpanel_text", text));
16     }
17 }

```

File: WicketSession.java

```

1 package dk.jsh.itdiplom.userdrivesignlanguagedictionary.wicket;
2
3 import dk.jsh.itdiplom.userdrivesignlanguagedictionary.entity.ApplicationUser;
4 import org.apache.wicket.Request;
5 import org.apache.wicket.Session;
6 import org.apache.wicket.protocol.http.WebSession;
7
8 /**
9  * Wicket session.
10 *
11 * @author Jan S. Hansen
12 */
13 public class WicketSession extends WebSession {
14     private ApplicationUser user;

```



```

15
16 public WicketSession(Request request) {
17     super(request);
18 }
19
20 public static WicketSession get() {
21     return (WicketSession) Session.get();
22 }
23
24 public boolean isAuthenticated() {
25     return (user != null);
26 }
27
28 public ApplicationUser getApplicationUser() {
29     return user;
30 }
31
32 public void setApplicationUser(ApplicationUser applicationUser) {
33     this.user = applicationUser;
34 }
35 }
36

```

Filen: style.css

```

1 body {
2     background-color: white;
3     background-repeat: no-repeat;
4     background-attachment: fixed;
5     background-position: 96% 96%;
6     color: #6F6F6F;
7     font-family: 'Lucida Sans', 'Helvetica', 'Sans-serif', 'sans';
8     font-size: 9pt;
9     line-height: 1.8em;
10    padding: 10px 10px 10px 10px;
11    margin: 10px 10px 10px 10px;
12 }
13
14 h1,h2,h3,h4,h5,h6,h7,h8 {
15     color: #E9601A;
16 }
17
18 #extitle {
19     font-size: 12pt;
20     font-weight: bold;
21     color: #E9601A;
22     padding: 10px 10px 10px 10px;
23 }
24
25 .feedbackPanelERROR {
26     color: red;
27     list-style: circle;
28     font-weight: bold;
29 }
30
31 .feedbackPanelINFO {
32     color: green;
33     list-style: circle;
34     font-weight: bold;
35 }
36
37 #hellomessage {
38     font-size: 30pt;
39 }
40
41 #titleblock {
42     background: #DEDEDE;
43     color: black;
44     border-top: solid #E9601A;
45     border-bottom: solid #E9601A;
46     border-width: thin;
47     padding: 2px 2px 2px 6px;
48 }

```

```

49
50 h2 {
51     font-size: 1.25em;
52 }
53
54 h3 {
55     font-size: 1em;
56 }
57
58 a {
59     color: #6F6F6F;
60     text-decoration: underline;
61 }
62
63 img {
64     border: none;
65 }
66
67 pre {
68     font-family: 'Lucida Sans', 'Helvetica', 'Sans serif', 'sans';
69 }
70
71 th {
72     background: #C3C3C3;
73     color: white;
74     font-weight: bold;
75 }
76
77 tr.b {
78     background: #F5F5F5;
79 }
80
81 tr.a {
82     background: #E6E6E5;
83 }
84
85 tr.none {
86     background: transparent;
87 }
88
89 a.none {
90     background: transparent;
91     padding-right: 0px;
92 }
93
94 a {
95     color: #E9601A;
96     font-weight: bold;
97     text-decoration: none;
98 }
99
100 a:hover {
101     text-decoration: underline;
102 }
103
104 li {
105     color: #E9601A;
106 }
107
108 em {
109     font-weight: bold
110 }
111
112 #inputForm {
113     width: 300px;
114 }
115
116 #inputForm label {
117     display: block;
118     margin-top: 5px;
119 }
120
121 #inputForm label.non {
122     display: inline !important;
123 }

```

```

124
125 #inputFormTable td {
126     vertical-align: top;
127     padding: 10px;
128 }
129
130 #nestedExampleTree {
131     width: 300px;
132 }
133
134 #feedbackPanel {
135     width: 600px;
136 }
137
138 #siteSelection {
139     width: 200px;
140 }
141
142 #error {
143     text-align: center;
144     color: red;
145 }
146
147 .even {
148     background-color: #fff;
149 }
150
151 .odd {
152     background-color: #ecec;
153 }

```

File: About.html

```

1 <?xml version="1.0" encoding="UTF-8"?>
2 <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
3   "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
4 <html xmlns:wicket="http://wicket.apache.org">
5 <head>
6   <meta http-equiv="Content-Type" content="text/html; charset=UTF-8"/>
7   <link rel="stylesheet" type="text/css" href="style.css"/>
8 </head>
9 <body>
10 <wicket:extend>
11 <span wicket:id="mainNavigation">
12   <h2>Om denne side</h2>
13   <table border="0">
14     <tr>
15       <td align="center">
16         <video wicket:id="aboutVideo" width="600">
17           Din browser understøtter ikke HTML5 Video.
18         </video>
19       </td>
20     </tr>
21     <tr>
22       <td>
23         <fieldset>
24           Denne side er tænkt som en brugerdreven dansk tegnsprogsordbog.<br/>
25           Dvs. at man som bruger, af denne side,
26           kan spørger andre tegnsprogskyndige,<br/>
27           om hjælp til ord man ikke selv kender.<br/>
28           <br/>
29           Eventuelle spørgsmål kan rettes til Jan Schrøder Hansen via <br/>
30           e-mail: <a href="mailto:jan.sch.hansen@gmail.com">jan.sch.hansen@gmail.com</a>
31         </fieldset>
32       </td>
33     </tr>
34   </table>
35 </span>
36 </wicket:extend>
37 </body>
38 </html>

```

File: About.java

```
1 package dk.jsh.itdiplom.userdrivesignlanguagedictionary.wicket.about;
2
3 import dk.jsh.itdiplom.userdrivesignlanguagedictionary.wicket.BasePage;
4 import dk.jsh.itdiplom.userdrivesignlanguagedictionary.wicket.homepage.MenuBorder;
5 import dk.jsh.itdiplom.userdrivesignlanguagedictionary.wicket.html5.Html5Video;
6 import dk.jsh.itdiplom.userdrivesignlanguagedictionary.wicket.html5.VideoSource;
7 import java.util.ArrayList;
8 import java.util.List;
9 import org.apache.wicket.ResourceReference;
10 import org.apache.wicket.markup.html.border.Border.BorderBodyContainer;
11 import org.apache.wicket.model.AbstractReadOnlyModel;
12 import org.apache.wicket.model.IModel;
13
14 /**
15  * About page.
16  *
17  * @author Jan S. Hansen
18  */
19 public final class About extends BasePage {
20
21     public About() {
22         MenuBorder menuBorder = new MenuBorder("mainNavigation");
23         add(menuBorder);
24         BorderBodyContainer borderBodyContainer = menuBorder.getBodyContainer();
25
26         final List<VideoSource> videoSources = new ArrayList<VideoSource>();
27         videoSources.add(new VideoSource(new ResourceReference(About.class, "About.ogv"),
28             VideoSource.VideoType.OGG));
29
30         IModel<List<VideoSource>> videoSourceList =
31             new AbstractReadOnlyModel<List<VideoSource>>() {
32                 @Override
33                 public List<VideoSource> getObject() {
34                     return videoSources;
35                 }
36             };
37         Html5Video html5Video = new Html5Video("aboutVideo", videoSourceList);
38
39         borderBodyContainer.add(html5Video);
40     }
41 }
```

File: HomePage.html

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-
transitional.dtd">
3 <html xmlns:wicket="http://wicket.apache.org">
4 <head>
5     <meta http-equiv="Content-Type" content="text/html; charset=UTF-8"/>
6     <link rel="stylesheet" type="text/css" href="style.css"/>
7 </head>
8 <body>
9 <wicket:extend>
10 <span wicket:id="mainNavigation">
11     <h2>S&oslash;g</h2>
12     <form action="#" wicket:id="form">
13         <table border="0">
14             <tr>
15                 <td>S&oslash;g efter:</td>
16                 <td>
17                     <select wicket:id="searchFor" name="" >
18                         <option>Ord</option>
19                         <option>Gruppe</option>
20                     </select>
21                 </td>
22                 <td>
23                     <input wicket:id="searchText" type="text"
24                         name="" value="" size="20"/>
```

```

25         title="Ord der skal søges efter. Brug * hvis du vil søge efter ord der ligner."
26         tabindex="1"/>
27     </td>
28     <td>
29         <input wicket:id="search" type="submit"
30             value="S&oslash;g" tabindex="2"/>
31     </td>
32 </tr>
33 </table>
34 </form>
35 <fieldset>
36     <legend>Resultat</legend>
37     <div wicket:id="feedback"></div>
38     <div wicket:id="pageable">
39         <table border="0" width="100%">
40             <tr>
41                 <td colspan="2">
42                     <a href="#" wicket:id="wordLink" title="Vis forslag">
43                         <span wicket:id="word">Ord</span>
44                     </a>
45                 </td>
46             </tr>
47             <tr>
48                 <td>
49                     &nbsp;
50                 </td>
51                 <td>
52                     Oprettet den <span wicket:id="created"></span>
53                 </td>
54             </tr>
55             <tr>
56                 <td>
57                     &nbsp;
58                 </td>
59                 <td>
60                     Grupper: <span wicket:id="groups"></span>
61                 </td>
62             </tr>
63         </table>
64     </div>
65 </fieldset>
66 <div style="float:right">
67     <span wicket:id="navigator">[dataview navigator]</span>
68 </div>
69 <div id="error">
70     <br/>
71     <table align="center">
72         <tr>
73             <td><img wicket:id="erroricon"/></td>
74             <td><span wicket:id="error">Error message goes here</span></td>
75         </tr>
76     </table>
77 </div>
78 </span>
79 </wicket:extend>
80 </body>
81 </html>

```

File: HomePage.java

```

1 package dk.jsh.itdiplom.userdrivesignlanguagedictionary.wicket.homepage;
2
3 import dk.jsh.itdiplom.userdrivesignlanguagedictionary.business.WordBusiness;
4 import dk.jsh.itdiplom.userdrivesignlanguagedictionary.business.WordGroupBusiness;
5 import dk.jsh.itdiplom.userdrivesignlanguagedictionary.entity.Word;
6 import dk.jsh.itdiplom.userdrivesignlanguagedictionary.entity.WordGroup;
7 import dk.jsh.itdiplom.userdrivesignlanguagedictionary.util.Text;
8 import dk.jsh.itdiplom.userdrivesignlanguagedictionary.wicket.BasePage;
9 import dk.jsh.itdiplom.userdrivesignlanguagedictionary.wicket.word.SelectedWord;
10 import java.util.ArrayList;
11 import java.util.List;
12 import org.apache.wicket.AttributeModifier;
13 import org.apache.wicket.Page;

```

```

14 import org.apache.wicket.ResourceReference;
15 import org.apache.wicket.markup.html.basic.Label;
16 import org.apache.wicket.markup.html.border.Border.BorderBodyContainer;
17 import org.apache.wicket.markup.html.form.Button;
18 import org.apache.wicket.markup.html.form.DropDownChoice;
19 import org.apache.wicket.markup.html.form.Form;
20 import org.apache.wicket.markup.html.form.TextField;
21 import org.apache.wicket.markup.html.image.Image;
22 import org.apache.wicket.markup.html.link.Link;
23 import org.apache.wicket.markup.html.list.ListItem;
24 import org.apache.wicket.markup.html.list.PageableListView;
25 import org.apache.wicket.markup.html.navigation.paging.PagingNavigator;
26 import org.apache.wicket.markup.html.panel.FeedbackPanel;
27 import org.apache.wicket.model.AbstractReadOnlyModel;
28 import org.apache.wicket.model.Model;
29 import org.apache.wicket.model.PropertyModel;
30
31 /**
32  * Home/search page.
33  *
34  * @author Jan S. Hansen
35  */
36 public class HomePage extends BasePage {
37     private String errorMessage = "";
38     private DropDownChoice<String> searchFor;
39     private TextField<String> searchText;
40     private Image errorIconImage = new Image("erroricon",
41         new ResourceReference(BasePage.class, "icons/attention.png"));
42     private List<Word> wordsFound = new ArrayList<Word>();
43     private PageableListView pageableListView;
44     private FeedbackPanel feedbackPanel;
45
46     public HomePage() {
47         MenuBorder menuBorder = new MenuBorder("mainNavigation");
48         add(menuBorder);
49         BorderBodyContainer borderBodyContainer = menuBorder.getBodyContainer();
50         PropertyModel errorMessageModel =
51             new PropertyModel(this, "errorMessage");
52         borderBodyContainer.add(new Label("error", errorMessageModel));
53         errorIconImage.setVisible(false);
54         borderBodyContainer.add(errorIconImage);
55
56         Form form = new Form("form") {
57             //Handles required fields error.
58             @Override
59             protected void onError() {
60                 if (!searchText.checkRequired()) {
61                     feedbackPanel.setVisible(false);
62                     setErrorMessage("Søgefeltet skal udfyldes.");
63                 }
64             }
65         };
66         borderBodyContainer.add(form);
67
68         searchFor = new DropDownChoice("searchFor",
69             new Model(SearchType.WORD.getDescription()),
70             SearchType.getDescriptions());
71         form.add(searchFor);
72         searchText = new TextField("searchText", new Model(""));
73         searchText.setRequired(true);
74         form.add(searchText);
75
76         //Add button to the form.
77         form.add(new Button("search") {
78             @Override
79             public void onSubmit() {
80                 removeErrorMessage();
81                 if (SearchType.getSearchType(searchFor.getModelObject())
82                     == SearchType.WORD) {
83                     wordsFound = WordBusiness.search(searchText.getModelObject());
84                 }
85                 else {
86                     List<WordGroup> wordGroups =
87                         WordGroupBusiness.search(searchText.getModelObject());
88                     if (!wordGroups.isEmpty()) {

```

```

89         wordsFound = WordBusiness.search(wordGroups);
90     }
91 }
92 if (wordsFound.isEmpty()) {
93     feedbackPanel.setVisible(true);
94     info("Ingen ord fundet.");
95 }
96 else {
97     feedbackPanel.setVisible(false);
98 }
99 pageableListView.setList(wordsFound);
100 pageableListView.setCurrentPage(0);
101 }
102 });
103
104 //Search results
105 feedbackPanel = new FeedbackPanel("feedback");
106 feedbackPanel.setVisible(false);
107 borderBodyContainer.add(feedbackPanel);
108 pageableListView =
109     new PageableListView("pageable", wordsFound, 4) {
110         @Override
111         protected void populateItem(final ListItem item) {
112             final Word word = (Word)item.getModelObject();
113             Label wordLabel = new Label("word", word.getWord());
114             Link wordLink = new Link("wordLink") {
115                 @Override
116                 public void onClick() {
117                     Page page = new SelectedWord(word);
118                     setResponsePage(page);
119                 }
120             };
121             wordLink.add(new AttributeModifier("title", true,
122                 new Model(word.getDescription())));
123             wordLink.add(wordLabel);
124             item.add(wordLink);
125
126             item.add(new Label("created",
127                 standardDateTimeFormat.format(word.getCreatedDateTime())));
128             List<String> wordGroupList = word.getSortedWordGroups();
129             item.add(new Label("groups", Text.makeWordGroupString(wordGroupList)));
130             item.add(new AttributeModifier("class",
131                 true, new AbstractReadOnlyModel<String>() {
132                 @Override
133                 public String getObject()
134                 {
135                     return (item.getIndex() % 2 == 1) ? "even" : "odd";
136                 }
137             }));
138         }
139     };
140
141 borderBodyContainer.add(pageableListView);
142 borderBodyContainer.add(new PagingNavigator("navigator", pageableListView));
143 }
144
145 /**
146  * Set error message.
147  */
148 public void setErrorMessage(String errorMessage) {
149     this.errorMessage = errorMessage;
150     errorIconImage.setVisible(true);
151 }
152
153 /**
154  * Remove error message.
155  */
156 public void removeErrorMessage() {
157     this.errorMessage = "";
158     errorIconImage.setVisible(false);
159 }
160
161 public enum SearchType {
162     WORD("Ord"),
163     GROUP("Gruppe");

```

```

164
165     private String description;
166
167     SearchType(String description) {
168         this.description = description;
169     }
170
171     public String getDescription() {
172         return description;
173     }
174
175     public static List<String> getDescriptions() {
176         List<String> descriptions = new ArrayList<String>();
177         descriptions.add(WORD.description);
178         descriptions.add(GROUP.description);
179         return descriptions;
180     }
181
182     public static SearchType getSearchType(String description) {
183         if (WORD.description.equals(description)) return WORD;
184         return GROUP;
185     }
186 }
187 }

```

Filen: MenuBorder.html

```

1 <?xml version="1.0" encoding="UTF-8"?>
2 <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
3 <html xmlns:wicket="http://wicket.apache.org/">
4 <head>
5   <link rel="stylesheet" type="text/css" href="style.css"/>
6 </head>
7 <body>
8 <wicket:border>
9 <table height = "100%">
10   <tr>
11     <td style="height: 100% valign = "top" >
12       <div wicket:id = "navigationBorder">
13         <h2>Menu</h2>
14         <ul>
15           <li wicket:id="menuItems">
16             <a href="#" wicket:id="menuItemLink">
17               <span wicket:id="menuItemText">Menu text</span>
18             </a>&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&~
19             </li>
20           </ul>
21           <ul>
22             <li>
23               <a href="#" wicket:id="loginLogOffMenuItemLink">
24                 <span wicket:id="loginLogoffText">Menu text</span>
25               </a>
26             </li>
27           </ul>
28         </div>
29       </td>
30     <td valign = "top">
31       <span wicket:id = "bodyBorder">
32         <wicket:body/>
33       </span>
34     </td>
35   </tr>
36 </table>
37 </wicket:border>
38 </body>
39 </html>
```


File: MenuBorder.java

```

1 package dk.jsh.itdiplom.userdrivensignlanguagedictionary.wicket.homepage;
2
3 import dk.jsh.itdiplom.userdrivensignlanguagedictionary.wicket.WicketSession;
4 import dk.jsh.itdiplom.userdrivensignlanguagedictionary.wicket.about.About;
5 import dk.jsh.itdiplom.userdrivensignlanguagedictionary.wicket.user.CreateUser;
6 import dk.jsh.itdiplom.userdrivensignlanguagedictionary.wicket.group.AllGroups;
7 import dk.jsh.itdiplom.userdrivensignlanguagedictionary.wicket.group.Groups;
8 import dk.jsh.itdiplom.userdrivensignlanguagedictionary.wicket.login.Login;
9 import dk.jsh.itdiplom.userdrivensignlanguagedictionary.wicket.request.AllRequest;
10 import dk.jsh.itdiplom.userdrivensignlanguagedictionary.wicket.request.Request;
11 import dk.jsh.itdiplom.userdrivensignlanguagedictionary.wicket.upload.Uploads;
12 import dk.jsh.itdiplom.userdrivensignlanguagedictionary.wicket.user.ChangeUser;
13 import org.apache.wicket.AttributeModifier;
14 import org.apache.wicket.markup.html.WebMarkupContainer;
15 import org.apache.wicket.markup.html.basic.Label;
16 import org.apache.wicket.markup.html.border.Border;
17 import org.apache.wicket.markup.html.border.BoxBorder;
18 import org.apache.wicket.markup.html.link.BookmarkablePageLink;
19 import org.apache.wicket.markup.html.link.Link;
20 import org.apache.wicket.markup.repeater.RepeatingView;
21 import org.apache.wicket.model.Model;
22
23 /**
24  * Menu border component.
25  *
26  * @author Jan S. Hansen
27  */
28 public class MenuBorder extends Border
29 {
30     /**
31      * Constructor
32      *
33      * @param id
34      *     The id of this component
35      */
36     public MenuBorder(final String id)
37     {
38         super(id);
39         final WicketSession session = WicketSession.get();
40
41         BoxBorder navigationBorder = new BoxBorder("navigationBorder");
42
43         RepeatingView repeatingView = new RepeatingView("menuItems");
44         navigationBorder.add(repeatingView);
45
46         addMenuLink(repeatingView, HomePage.class, "Søg", "Søg efter ord");
47         addMenuLink(repeatingView, About.class, "Om denne side",
48             "Information om denne side");
49         if (session.isAuthenticated()) {
50             addMenuLink(repeatingView, AllRequest.class, "Alle forespørgsler",
51                 "Vis oversigt over alle forespørgsler, som mangler forslag.");
52             addMenuLink(repeatingView, AllGroups.class, "Alle grupper",
53                 "Vis oversigt over alle grupper");
54             addMenuLink(repeatingView, Request.class, "Mine forespørgsler",
55                 "Vis oversigt over egne forespørgsler");
56             addMenuLink(repeatingView, Groups.class, "Mine grupper",
57                 "Vis oversigt over egne grupper");
58             addMenuLink(repeatingView, Uploads.class, "Mine uploads",
59                 "Vis oversigt over egne uploads");
60             addMenuLink(repeatingView, ChangeUser.class, "Ret brugeroplysninger",
61                 "Vis/ret egne brugeroplysninger");
62             addLogoutMenuLink(navigationBorder, session);
63         }
64         else {
65             addMenuLink(repeatingView, AllRequest.class, "Alle forespørgsler",
66                 "Vis oversigt over alle forespørgsler, som mangler forslag.");
67             addMenuLink(repeatingView, AllGroups.class, "Alle grupper",
68                 "Vis oversigt over alle grupper");
69             addMenuLink(repeatingView, CreateUser.class, "Ny bruger",
70                 "Opret ny bruger");
71             addLoginMenuLink(navigationBorder);
72         }
73     }

```

```

74     add(navigationBorder);
75     add(new BoxBorder("bodyBorder"));
76 }
77
78 private void addMenuLink(RepeatingView repeatingView, Class pageClass,
79     String text, String title) {
80     WebMarkupContainer parent =
81         new WebMarkupContainer(repeatingView.newChildId());
82     repeatingView.add(parent);
83     BookmarkablePageLink link = new BookmarkablePageLink("menuItemLink",
84         pageClass);
85     link.add(new AttributeModifier("title", true,
86         new Model(title)));
87     parent.add(link);
88     link.add(new Label("menuItemText", text));
89 }
90
91 private void addLoginMenuLink(BoxBorder navigationBorder) {
92     BookmarkablePageLink loginLink =
93         new BookmarkablePageLink("loginLogOffMenuItemLink",
94             Login.class);
95     navigationBorder.add(loginLink);
96     loginLink.add(new AttributeModifier("title", true,
97         new Model("Log på systemet.")));
98     loginLink.add(new Label("loginLogoffText", "Log på"));
99 }
100
101 private void addLogoffMenuLink(BoxBorder navigationBorder,
102     final WicketSession session) {
103     Link logoffLink = new Link("loginLogOffMenuItemLink") {
104         @Override
105         public void onClick() {
106             session.setApplicationUser(null);
107             setResponsePage(HomePage.class);
108         }
109     };
110     navigationBorder.add(logoffLink);
111     logoffLink.add(new AttributeModifier("title", true,
112         new Model("Log af systemet.")));
113     logoffLink.add(new Label("loginLogoffText", "Log af"));
114 }
115 }

```

File: Html5Video.java

```

1 package dk.jsh.itdiplom.userdrivensignlanguagedictionary.wicket.html5;
2
3 import java.util.List;
4 import org.apache.wicket.ResourceReference;
5 import org.apache.wicket.markup.ComponentTag;
6 import org.apache.wicket.markup.MarkupStream;
7 import org.apache.wicket.markup.html.WebMarkupContainer;
8 import org.apache.wicket.model.IModel;
9 import org.apache.wicket.util.string.AppendingStringBuffer;
10
11 /**
12  * HTML 5 Video tag.
13  *
14  * This is created with "Integrating HTML5 and Wicket" as a inspiration.
15  * see http://wicketbyexample.com/integrating-html5-and-wicket/
16  *
17  * @author Jan S. Hansen
18  */
19 public class Html5Video extends WebMarkupContainer {
20
21     private IModel<List<VideoSource>> sources;
22
23     public Html5Video(String id, final IModel<List<VideoSource>> model) {
24         super(id, model);
25         this.sources = wrap(model);
26     }
27
28

```

```

29 @Override
30 protected void onComponentTag(ComponentTag tag) {
31     checkComponentTag(tag, "video");
32     tag.put("autobuffer", true);
33     //tag.put("autoplay", false);
34     //tag.put("loop", false);
35     tag.put("controls", true);
36     super.onComponentTag(tag);
37 }
38
39 @Override
40 protected void onComponentTagBody(MarkupStream markupStream,
41     ComponentTag openTag) {
42
43     if (sources != null) {
44         final AppendingStringBuffer buffer = new AppendingStringBuffer();
45         List<VideoSource> videoSources = sources.getObject();
46         for (VideoSource videoSource : videoSources) {
47             buffer.append("\n<source ");
48             buffer.append("src=");
49             ResourceReference resourceReference = videoSource.getSource();
50             buffer.append(urlFor(resourceReference));
51             buffer.append("");
52             String videoType = videoSource.getType().getVideoType();
53             if (videoType != null) {
54                 buffer.append(" type=");
55                 buffer.append(videoType);
56                 buffer.append("");
57             }
58
59             buffer.append(" />");
60         }
61
62         buffer.append("\n");
63
64         getResponse().write(buffer.toString());
65
66     }
67     super.onComponentTagBody(markupStream, openTag);
68 }
69 }

```

File: VideoSource.java

```

1 package dk.jsh.itdiplom.userdrivensignlanguagedictionary.wicket.html5;
2
3 import org.apache.wicket.ResourceReference;
4
5 /**
6  * Video source.
7  *
8  * @author Jan S. Hansen
9  */
10 public class VideoSource {
11     public enum VideoType {
12         OGG("video/ogg"),
13         MP4("video/mp4");
14
15         private String type;
16
17         VideoType(String type) {
18             this.type = type;
19         }
20
21         public String getVideoType() {
22             return type;
23         }
24     };
25
26     private ResourceReference source;
27     private VideoType type;
28
29     public VideoSource(ResourceReference source, VideoType type) {

```

```

30     this.source = source;
31     this.type = type;
32 }
33
34 public ResourceReference getSource() {
35     return source;
36 }
37
38 public void setSource(ResourceReference source) {
39     this.source = source;
40 }
41
42 public VideoType getType() {
43     return type;
44 }
45
46 public void setType(VideoType type) {
47     this.type = type;
48 }
49 }

```

Filen: Login.html

```

1 <?xml version="1.0" encoding="UTF-8"?>
2 <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-
transitional.dtd">
3 <html xmlns:wicket="http://wicket.apache.org">
4 <head>
5     <meta http-equiv="Content-Type" content="text/html; charset=UTF-8"/>
6     <link rel="stylesheet" type="text/css" href="style.css"/>
7 </head>
8 <body>
9 <wicket:extend>
10 <span wicket:id="mainNavigation">
11     <div style="width: 300px">
12         <h2>Log på</h2>
13         <form action="#" wicket:id="form">
14             <table border="0">
15                 <tbody>
16                     <tr>
17                         <td style="width:120px">Brugerkode:</td>
18                         <td>
19                             <input wicket:id="userlogin" type="text"
20                                 name="" value="" size="20" tabindex="1"/>
21                         </td>
22                     </tr>
23                     <tr>
24                         <td style="width:120px">Password:</td>
25                         <td>
26                             <input wicket:id="password" type="password"
27                                 name="" value="" size="20" tabindex="2"/>
28                         </td>
29                     </tr>
30                     <tr>
31                         <td colspan="2">&nbsp;  </td>
32                     </tr>
33                     <tr>
34                         <td colspan="2">
35                             <div style="float:right">
36                                 <input wicket:id="login" type="submit"
37                                     value="Login" tabindex="3"/>
38                             </div>
39                         </td>
40                     </tr>
41                 </tbody>
42             </table>
43         </form>
44         <div id="error">
45             <table align="center">
46                 <tr>
47                     <td><img wicket:id="erroricon"/></td>
48                     <td><span wicket:id="error">Error message goes here</span></td>
49                 </tr>
50             </table>

```

```

51     </div>
52 </div>
53 </span>
54 </wicket:extend>
55 </body>
56 </html>

```

Filen: Login.java

```

1 package dk.jsh.itdiplom.userdrivensignlanguagedictionary.wicket.login;
2
3 import dk.jsh.itdiplom.userdrivensignlanguagedictionary.business.ApplicationUserBusiness;
4 import dk.jsh.itdiplom.userdrivensignlanguagedictionary.entity.ApplicationUser;
5 import dk.jsh.itdiplom.userdrivensignlanguagedictionary.wicket.BasePage;
6 import dk.jsh.itdiplom.userdrivensignlanguagedictionary.wicket.WicketSession;
7 import dk.jsh.itdiplom.userdrivensignlanguagedictionary.wicket.homepage.HomePage;
8 import dk.jsh.itdiplom.userdrivensignlanguagedictionary.wicket.homepage.MenuBorder;
9 import org.apache.wicket.AttributeModifier;
10 import org.apache.wicket.ResourceReference;
11 import org.apache.wicket.markup.html.basic.Label;
12 import org.apache.wicket.markup.html.border.Border.BorderBodyContainer;
13 import org.apache.wicket.markup.html.form.Button;
14 import org.apache.wicket.markup.html.form.Form;
15 import org.apache.wicket.markup.html.form.PasswordTextField;
16 import org.apache.wicket.markup.html.form.TextField;
17 import org.apache.wicket.markup.html.image.Image;
18 import org.apache.wicket.model.Model;
19 import org.apache.wicket.model.PropertyModel;
20
21 /**
22  * Login page.
23  *
24  * @author Jan S. Hansen
25  */
26 public final class Login extends BasePage {
27     private String errorMessage = "";
28     private TextField<String> userLogin;
29     private TextField<String> password;
30     private Image errorIconImage = new Image("erroricon",
31         new ResourceReference(BasePage.class, "icons/attention.png"));
32
33     /**
34      * Constructor.
35      */
36     public Login() {
37         MenuBorder menuBorder = new MenuBorder("mainNavigation");
38         add(menuBorder);
39         BorderBodyContainer borderBodyContainer = menuBorder.getBodyContainer();
40
41         PropertyModel errorMessageModel =
42             new PropertyModel(this, "errorMessage");
43         borderBodyContainer.add(new Label("error", errorMessageModel));
44         //Add a form as an inner class.
45         Form form = new Form("form") {
46             //Handles required fields error.
47             @Override
48             protected void onError() {
49                 boolean emptyFields = false;
50                 if (!userLogin.checkRequired()) {
51                     emptyFields = true;
52                     userLogin.add(new AttributeModifier("style", true,
53                         new Model("border-color:red;")));
54                 }
55                 else {
56                     userLogin.add(new AttributeModifier("style", true,
57                         new Model("border-color:default;")));
58                 }
59                 if (!password.checkRequired()) {
60                     emptyFields = true;
61                     password.add(new AttributeModifier("style", true,
62                         new Model("border-color:red;")));
63                 }
64                 else {
65                     password.add(new AttributeModifier("style", true,

```

```

66         new Model("border-color:default;"));
67     }
68     if (emptyFields) {
69         setErrorMessage("Begge felter skal udfyldes.");
70     }
71 }
72 };
73 borderBodyContainer.add(form);
74
75 //Add fields to the form.
76 userLogin = new TextField("userlogin", new Model(""));
77 userLogin.setRequired(true);
78 form.add(userLogin);
79
80 password = new PasswordTextField("password", new Model(""));
81 password.setRequired(true);
82 form.add(password);
83
84 errorIconImage.setVisible(false);
85 borderBodyContainer.add(errorIconImage);
86
87 //Add button to the form.
88 form.add(new Button("login") {
89     @Override
90     public void onSubmit() {
91         ApplicationUser appUser =
92             ApplicationUserBusiness.isValidUser(userLogin.getModelObject(),
93             password.getModelObject());
94         if (appUser != null) {
95             WicketSession.get().setApplicationUser(appUser);
96             setResponsePage(HomePage.class);
97         }
98         else {
99             setErrorMessage("Fejl i login eller password.");
100         }
101     }
102 });
103 }
104
105 /**
106  * Set error message.
107  */
108 public void setErrorMessage(String errorMessage) {
109     this.errorMessage = errorMessage;
110     errorIconImage.setVisible(true);
111 }
112 }

```

File: AllRequest.html

```

1 <?xml version="1.0" encoding="UTF-8"?>
2 <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-
transitional.dtd">
3 <html xmlns:wicket="http://wicket.apache.org">
4 <head>
5 <meta http-equiv="Content-Type" content="text/html; charset=UTF-8"/>
6 <link rel="stylesheet" type="text/css" href="style.css"/>
7 </head>
8 <body>
9 <wicket:extend>
10 <span wicket:id="mainNavigation">
11 <h2>Alle foresp&oslash;rsler</h2>
12 <fieldset>
13 <legend>Alle foresp&oslash;rsler uden uploads</legend>
14 <div wicket:id="feedback"></div>
15 <div wicket:id="pageable">
16 <table border="0" width="100%">
17 <tr>
18 <td colspan="3">
19 <a href="#" wicket:id="wordLink">
20 <span wicket:id="word">Ord</span>
21 </a>
22 </td>
23 </tr>

```

```

24         <tr>
25             <td>&nbsp;</td>
26             <td>
27                 Oprettet den <span wicket:id="created"></span>
28             </td>
29             <td>&nbsp;</td>
30         </tr>
31         <tr>
32             <td>&nbsp;</td>
33             <td>
34                 Grupper: <span wicket:id="groups"></span>
35             </td>
36             <td align="right">
37                 <a href="#" wicket:id="uploadLink" title="Upload video forslag">
38                     Upload forslag
39                 </a>
40             </td>
41         </tr>
42     </table>
43 </div>
44 </fieldset>
45 <div style="float:right">
46     <span wicket:id="navigator">[dataview navigator]</span>
47 </div>
48 </span>
49 </wicket:extend>
50 </body>
51 </html>

```

Filen: AllRequest.java

```

1 package dk.jsh.itdiplom.userdrivensignlanguagedictionary.wicket.request;
2
3 import dk.jsh.itdiplom.userdrivensignlanguagedictionary.business.WordBusiness;
4 import dk.jsh.itdiplom.userdrivensignlanguagedictionary.entity.Word;
5 import dk.jsh.itdiplom.userdrivensignlanguagedictionary.util.Text;
6 import dk.jsh.itdiplom.userdrivensignlanguagedictionary.wicket.BasePage;
7 import dk.jsh.itdiplom.userdrivensignlanguagedictionary.wicket.WicketSession;
8 import dk.jsh.itdiplom.userdrivensignlanguagedictionary.wicket.homepage.MenuBorder;
9 import dk.jsh.itdiplom.userdrivensignlanguagedictionary.wicket.upload.Upload;
10 import dk.jsh.itdiplom.userdrivensignlanguagedictionary.wicket.word.SelectedWord;
11 import java.util.List;
12 import org.apache.wicket.AttributeModifier;
13 import org.apache.wicket.Page;
14 import org.apache.wicket.markup.html.basic.Label;
15 import org.apache.wicket.markup.html.border.Border.BorderBodyContainer;
16 import org.apache.wicket.markup.html.link.Link;
17 import org.apache.wicket.markup.html.list.ListItem;
18 import org.apache.wicket.markup.html.list.PageableListView;
19 import org.apache.wicket.markup.html.navigation.paging.PagingNavigator;
20 import org.apache.wicket.markup.html.panel.FeedbackPanel;
21 import org.apache.wicket.model.AbstractReadOnlyModel;
22 import org.apache.wicket.model.Model;
23
24 /**
25  * All request page.
26  *
27  * @author Jan S. Hansen
28  */
29 public final class AllRequest extends BasePage {
30
31     public AllRequest() {
32         final WicketSession wicketSession = WicketSession.get();
33
34         MenuBorder menuBorder = new MenuBorder("mainNavigation");
35         add(menuBorder);
36         BorderBodyContainer borderBodyContainer = menuBorder.getBodyContainer();
37         borderBodyContainer.add(new FeedbackPanel("feedback"));
38
39         List<Word> wordsWithoutUploads =
40             WordBusiness.getAllWordsWithoutUploads();
41         if (wordsWithoutUploads.size() == 0) {
42             info("Ingen forspørgelser uden forslag.");
43         }
44     }
45 }

```

```

43     }
44     PageableListView pageableListView =
45         new PageableListView("pageable", wordsWithoutUploads, 6) {
46         @Override
47         protected void populateItem(final ListItem item) {
48             final Word word = (Word)item.getModelObject();
49             Label wordLabel = new Label("word", word.getWord());
50             Link wordLink = new Link("wordLink") {
51                 @Override
52                 public void onClick() {
53                     Page page = new SelectedWord(word);
54                     setResponsePage(page);
55                 }
56             };
57             wordLink.add(new AttributeModifier("title", true,
58                 new Model(word.getDescription())));
59             wordLink.add(wordLabel);
60             item.add(wordLink);
61
62             item.add(new Label("created",
63                 standardDateFormat.format(word.getCreatedDateTime())));
64             List<String> wordGroupList = word.getSortedWordGroups();
65             item.add(new Label("groups", Text.makeWordGroupString(wordGroupList)));
66
67             Link uploadLink = new Link("uploadLink") {
68                 @Override
69                 public void onClick() {
70                     Page page = new Upload(word);
71                     setResponsePage(page);
72                 }
73             };
74             if (!wicketSession.isAuthenticated()) {
75                 uploadLink.setEnabled(false);
76             }
77             item.add(uploadLink);
78
79
80             item.add(new AttributeModifier("class",
81                 true, new AbstractReadOnlyModel<String>() {
82                 @Override
83                 public String getObject()
84                 {
85                     return (item.getIndex() % 2 == 1) ? "even" : "odd";
86                 }
87             }));
88         }
89     };
90
91     borderBodyContainer.add(pageableListView);
92     borderBodyContainer.add(new PagingNavigator("navigator", pageableListView));
93 }
94 }

```

File: NewRequest.html

```

1 <?xml version="1.0" encoding="UTF-8"?>
2 <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-
3 <html xmlns:wicket="http://wicket.apache.org">
4 <head>
5     <meta http-equiv="Content-Type" content="text/html; charset=UTF-8"/>
6     <link rel="stylesheet" type="text/css" href="style.css"/>
7 </head>
8 <body>
9 <wicket:extend>
10 <span wicket:id="mainNavigation">
11     <h2>Opret foresp&oslash;rgsel</h2>
12     <form action="#" wicket:id="form">
13         <table border="0" width="100%">
14             <tbody>
15                 <tr>
16                     <td>Ord der ønskes forslag til:</td>
17                     <td>

```



```

27 private String errorMessage = "";
28 private TextField<String> word;
29 private TextArea<String> description;
30 private Image errorIconImage = new Image("erroricon",
31     new ResourceReference(BasePage.class, "icons/attention.png"));
32
33 public NewRequest() {
34     MenuBorder menuBorder = new MenuBorder("mainNavigation");
35     add(menuBorder);
36     BorderBodyContainer borderBodyContainer = menuBorder.getBodyContainer();
37
38     Form form = new Form("form") {
39         //Handles required fields error.
40         @Override
41         protected void onError() {
42             if (emptyRequiredFields()) {
43                 setErrorMessage("Alle felter skal udfyldes.");
44                 return;
45             }
46         }
47
48         private boolean emptyRequiredFields() {
49             //Test for empty/required fields
50             boolean emptyFields = false;
51             if (!word.checkRequired()) {
52                 emptyFields = true;
53                 word.add(new AttributeModifier("style", true,
54                     new Model("border-color:red;")));
55             }
56             else {
57                 word.add(new AttributeModifier("style", true,
58                     new Model("border-color:default;")));
59             }
60             if (!description.checkRequired()) {
61                 emptyFields = true;
62                 description.add(new AttributeModifier("style", true,
63                     new Model("border-color:red;")));
64             }
65             else {
66                 description.add(new AttributeModifier("style", true,
67                     new Model("border-color:default;")));
68             }
69             return emptyFields;
70         }
71     };
72
73     borderBodyContainer.add(form);
74
75     //Add fields to the form.
76     word = new TextField("word", new Model(""));
77     word.setRequired(true);
78     form.add(word);
79
80     description = new TextArea("description", new Model(""));
81     description.setRequired(true);
82     form.add(description);
83
84     //Add buttons to the form.
85     form.add(new Button("save") {
86         @Override
87         public void onSubmit() {
88             //Test if word exists
89             if (WordBusiness.isWordInUse(word.getModelObject())) {
90                 setErrorMessage("Ordet findes i forvejen.");
91                 word.add(new AttributeModifier("style", true,
92                     new Model("border-color:red;")));
93                 return;
94             }
95             else {
96                 word.add(new AttributeModifier("style", true,
97                     new Model("border-color:default;")));
98             }
99
100             //Save
101             WicketSession session = WicketSession.get();

```

```

102         Word newWord = new Word(word.getModelObject(),
103             description.getModelObject(), new Date(),
104             session.getApplicationUser());
105
106         WordBusiness.saveNew(newWord);
107         setResponsePage(Request.class);
108     }
109 });
110
111
112 //Add error items
113 PropertyModel errorMessageModel =
114     new PropertyModel(this, "errorMessage");
115 borderBodyContainer.add(new Label("error", errorMessageModel));
116 errorIconImage.setVisible(false);
117 borderBodyContainer.add(errorIconImage);
118 }
119
120 /**
121  * Set error message.
122  */
123 public void setErrorMessage(String errorMessage) {
124     this.errorMessage = errorMessage;
125     errorIconImage.setVisible(true);
126 }
127 }

```

File: Request.html

```

1 <?xml version="1.0" encoding="UTF-8"?>
2 <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-
transitional.dtd">
3 <html xmlns:wicket="http://wicket.apache.org">
4 <head>
5     <meta http-equiv="Content-Type" content="text/html; charset=UTF-8"/>
6     <link rel="stylesheet" type="text/css" href="style.css"/>
7 </head>
8 <body>
9 <wicket:extend>
10 <span wicket:id="mainNavigation">
11     <h2>Mine foresp&oslash;rgsler</h2>
12     <table border="0" width="100%">
13         <tr>
14             <td>
15                 <div style="float:right">
16                     <a href="#" wicket:id="createLink">
17                         Opret ny foresp&oslash;rgsel
18                     </a>
19                 </div>
20             </td>
21         </tr>
22         <tr>
23             <td>
24                 <fieldset>
25                     <legend>Mine foresp&oslash;rgsler</legend>
26                     <div wicket:id="feedback"></div>
27                     <div wicket:id="pageable">
28                         <table border="0" width="100%">
29                             <tr>
30                                 <td colspan="3">
31                                     <a href="#" wicket:id="wordLink" title="Vis forslag">
32                                         <span wicket:id="word">Ord</span>
33                                     </a>
34                                 </td>
35                             </tr>
36                             <tr>
37                                 <td>&nbsp;&nbsp;&nbsp;</td>
38                                 <td>
39                                     Oprettet den <span wicket:id="created"></span>
40                                 </td>
41                                 <td>&nbsp;&nbsp;&nbsp;</td>
42                             </tr>
43                             <tr>

```

```

44         <td>&nbsp;</td>
45     </td>
46     Grupper: <span wicket.id="groups"></span>
47 </td>
48 <td align="right">
49     <a href="#" title="Ikke implementeret">Ret</a>
50 </td>
51 </tr>
52 </table>
53 </div>
54 </fieldset>
55 </td>
56 </tr>
57 </table>
58 <div style="float:right">
59     <span wicket.id="navigator">[dataview navigator]</span>
60 </div>
61 </span>
62 </wicket:extend>
63 </body>
64 </html>

```

File: Request.java

```

1 package dk.jsh.itdiplom.userdrivesignlanguagedictionary.wicket.request;
2
3 import dk.jsh.itdiplom.userdrivesignlanguagedictionary.business.WordBusiness;
4 import dk.jsh.itdiplom.userdrivesignlanguagedictionary.entity.Word;
5 import dk.jsh.itdiplom.userdrivesignlanguagedictionary.util.Text;
6 import dk.jsh.itdiplom.userdrivesignlanguagedictionary.wicket.BasePage;
7 import dk.jsh.itdiplom.userdrivesignlanguagedictionary.wicket.WicketSession;
8 import dk.jsh.itdiplom.userdrivesignlanguagedictionary.wicket.homepage.MenuBorder;
9
10 import dk.jsh.itdiplom.userdrivesignlanguagedictionary.wicket.word.SelectedWord;
11 import java.util.List;
12 import org.apache.wicket.AttributeModifier;
13 import org.apache.wicket.Page;
14 import org.apache.wicket.markup.html.basic.Label;
15 import org.apache.wicket.markup.html.border.Border.BorderBodyContainer;
16 import org.apache.wicket.markup.html.link.BookmarkablePageLink;
17 import org.apache.wicket.markup.html.link.Link;
18 import org.apache.wicket.markup.html.list.ListItem;
19 import org.apache.wicket.markup.html.list.PageableListView;
20 import org.apache.wicket.markup.html.navigation.paging.PagingNavigator;
21 import org.apache.wicket.markup.html.panel.FeedbackPanel;
22 import org.apache.wicket.model.AbstractReadOnlyModel;
23 import org.apache.wicket.model.Model;
24
25 /**
26  * My request page.
27  *
28  * @author Jan S. Hansen
29  */
30 public final class Request extends BasePage {
31
32     public Request() {
33         MenuBorder menuBorder = new MenuBorder("mainNavigation");
34         add(menuBorder);
35         BorderBodyContainer borderBodyContainer = menuBorder.getBodyContainer();
36
37         //Link to create new request
38         BookmarkablePageLink createNewLink =
39             new BookmarkablePageLink("createLink",
40                 NewRequest.class);
41         borderBodyContainer.add(createNewLink);
42         borderBodyContainer.add(new FeedbackPanel("feedback"));
43
44         WicketSession wicketSession = WicketSession.get();
45         List<Word> allWords =
46             WordBusiness.getAllWordsCreatedByUser(wicketSession.getApplicationUser());
47         if (allWords.size() == 0) {
48             info("Ingen forespørgelser.");
49         }
50     }
51 }

```

```

50
51 PageableListView pageableListView =
52     new PageableListView("pageable", allWords, 4) {
53         @Override
54         protected void populateItem(final ListItem item) {
55             final Word word = (Word)item.getModelObject();
56             Label wordLabel = new Label("word", word.getWord());
57             Link wordLink = new Link("wordLink") {
58                 @Override
59                 public void onClick() {
60                     Page page = new SelectedWord(word);
61                     setResponsePage(page);
62                 }
63             };
64             wordLink.add(new AttributeModifier("title", true,
65                 new Model(word.getDescription())));
66             wordLink.add(wordLabel);
67             item.add(wordLink);
68
69             item.add(new Label("created",
70                 standardDateFormat.format(word.getCreatedDateTime())));
71             List<String> wordGroupList = word.getSortedWordGroups();
72             item.add(new Label("groups", Text.makeWordGroupString(wordGroupList)));
73             item.add(new AttributeModifier("class",
74                 true, new AbstractReadOnlyModel<String>() {
75                 @Override
76                 public String getObject()
77                 {
78                     return (item.getIndex() % 2 == 1) ? "even" : "odd";
79                 }
80             }));
81         }
82     };
83
84     borderBodyContainer.add(pageableListView);
85     borderBodyContainer.add(new PagingNavigator("navigator", pageableListView));
86 }
87 }

```

File: Upload.html

```

1 <?xml version="1.0" encoding="UTF-8"?>
2 <!DOCTYPE html PUBLIC "-//W3C/DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-
3 transitional.dtd">
4 <html xmlns:wicket="http://wicket.apache.org">
5 <head>
6 <meta http-equiv="Content-Type" content="text/html; charset=UTF-8"/>
7 <link rel="stylesheet" type="text/css" href="style.css"/>
8 </head>
9 <body>
10 <wicket:extend>
11 <span wicket:id = "mainNavigation">
12 <h2>Upload forslag til ord: <span wicket:id="word">[word]</span></h2>
13 <table border="0">
14 <tr>
15 <td>
16 <div style="float:right">
17 <a href="#" wicket:id="back">Tilbage</a>
18 </div>
19 </td>
20 <tr>
21 <td>&nbsp;</td>
22 </tr>
23 <tr>
24 <td>
25 <form wicket:id="form">
26 <table border="0">
27 <tr>
28 <td>
29 <label>Vælg video fil:</label>&nbsp;<input wicket:id="fileUpload" size="50" type="file"/>
30 </td>

```

```

32         </tr>
33     </tr>
34     <td>
35         <label>Beskrivelse af video:</label>
36     </td>
37 </tr>
38 <tr>
39     <td>
40         <textarea wicket:id="description" rows="5" cols="60">
41         </textarea>
42     </td>
43 </tr>
44 <tr>
45     <td align="right">
46         <input type="submit" value="Upload og gem"/>
47     </td>
48 </tr>
49 </table>
50 </form>
51 </td>
52 </tr>
53 </table>
54 <div id="error">
55     <table align="center">
56     <tr>
57         <td><img wicket:id="erroricon"/></td>
58         <td><span wicket:id="error">Error message goes here</span></td>
59     </tr>
60 </table>
61 </div>
62 </span>
63 </wicket:extend>
64 </body>
65 </html>

```

Filen: Upload.java

```

1 package dk.jsh.itdiplom.userdrivensignlanguagedictionary.wicket.upload;
2
3 import dk.jsh.itdiplom.userdrivensignlanguagedictionary.business.VideoFileBusiness;
4 import dk.jsh.itdiplom.userdrivensignlanguagedictionary.entity.ApplicationUser;
5 import dk.jsh.itdiplom.userdrivensignlanguagedictionary.entity.VideoFile;
6 import dk.jsh.itdiplom.userdrivensignlanguagedictionary.entity.Word;
7 import dk.jsh.itdiplom.userdrivensignlanguagedictionary.util.ConvertVideo;
8 import dk.jsh.itdiplom.userdrivensignlanguagedictionary.util.EMailSender;
9 import dk.jsh.itdiplom.userdrivensignlanguagedictionary.wicket.BasePage;
10 import dk.jsh.itdiplom.userdrivensignlanguagedictionary.wicket.WicketSession;
11 import dk.jsh.itdiplom.userdrivensignlanguagedictionary.wicket.homepage.MenuBorder;
12 import dk.jsh.itdiplom.userdrivensignlanguagedictionary.wicket.word.SelectedWord;
13 import java.io.File;
14 import java.util.Date;
15 import java.util.logging.Level;
16 import java.util.logging.Logger;
17 import org.apache.wicket.AttributeModifier;
18 import org.apache.wicket.Page;
19 import org.apache.wicket.ResourceReference;
20 import org.apache.wicket.markup.html.basic.Label;
21 import org.apache.wicket.markup.html.border.Border.BorderBodyContainer;
22 import org.apache.wicket.markup.html.form.Form;
23 import org.apache.wicket.markup.html.form.TextArea;
24 import org.apache.wicket.markup.html.form.upload.FileUpload;
25 import org.apache.wicket.markup.html.form.upload.FileUploadField;
26 import org.apache.wicket.markup.html.image.Image;
27 import org.apache.wicket.markup.html.link.Link;
28 import org.apache.wicket.model.Model;
29 import org.apache.wicket.model.PropertyModel;
30
31 /**
32  * Upload page.
33  *
34  * @author Jan S. Hansen
35  */
36 public final class Upload extends BasePage {
37     static final Logger logger = Logger.getLogger(Upload.class.getName());

```

```

38
39 private FileUploadField fileUpload;
40 private TextArea<String> description;
41 private String UPLOAD_FOLDER = "C:\\Temp\\Upload\\";
42 private String errorMessage = "";
43 private Image errorIconImage = new Image("erroricon",
44     new ResourceReference(BasePage.class, "icons/attention.png"));
45
46 public Upload(final Word word) {
47     MenuBorder menuBorder = new MenuBorder("mainNavigation");
48     add(menuBorder);
49     BorderBodyContainer borderBodyContainer = menuBorder.getBodyContainer();
50     borderBodyContainer.add(new Label("word", word.getWord()));
51     Link back = new Link("back") {
52         @Override
53         public void onClick() {
54             Page page = new SelectedWord(word);
55             setResponsePage(page);
56         }
57     };
58     borderBodyContainer.add(back);
59
60     Form form = new Form("form") {
61         @Override
62         protected void onError() {
63             if (emptyRequiredFields()) {
64                 setErrorMessage("Vælg video fil' skal udfyldes.");
65                 return;
66             }
67         }
68
69         private boolean emptyRequiredFields() {
70             //Test for empty/required fields
71             boolean emptyFields = false;
72             if (!fileUpload.checkRequired()) {
73                 emptyFields = true;
74                 fileUpload.add(new AttributeModifier("style", true,
75                     new Model("border-color:red;")));
76             }
77             else {
78                 fileUpload.add(new AttributeModifier("style", true,
79                     new Model("border-color:default;")));
80             }
81             return emptyFields;
82         }
83
84         @Override
85         protected void onSubmit() {
86             errorIconImage.setVisible(false);
87             final FileUpload uploadedFile = fileUpload.getFileUpload();
88             if (uploadedFile != null) {
89                 WicketSession wicketSession = WicketSession.get();
90                 ApplicationUser user = wicketSession.getApplicationUser();
91                 String userId = user.getId().toString();
92                 String fileName = uploadedFile.getClientFileName();
93
94                 // write to a new file
95                 File newFile = new File(UPLOAD_FOLDER
96                     + "UserId_" + userId + "_" + fileName);
97                 if (newFile.exists()) {
98                     newFile.delete();
99                 }
100                 try {
101                     newFile.createNewFile();
102                     uploadedFile.writeTo(newFile);
103
104                     ConvertVideo cv = new ConvertVideo();
105                     String destVideoReferenceName =
106                         cv.createOgvResourceName(
107                             userId,
108                             word.getId().toString());
109                     String destVideoPath = cv.createOgvFilename(
110                         destVideoReferenceName);
111                     cv.convert(newFile.getAbsolutePath(), destVideoPath);
112

```

```

113         File convertedFile = new File(destVideoPath);
114         if (convertedFile.exists()) {
115
116             VideoFile videoFile = new VideoFile(fileName,
117                 description.getModelObject(),
118                 destVideoReferenceName, new Date(), user, word);
119             VideoFileBusiness.saveNew(videoFile);
120             emailToRequester(word, videoFile);
121             Page page = new SelectedWord(word);
122             setResponsePage(page);
123         }
124         else {
125             setErrorMessage("Fejl ved konvertering af filen.");
126         }
127     }
128     catch (Exception exception) {
129         logger.log(Level.SEVERE, "Error converting video", exception);
130         setErrorMessage("Fejl under upload.");
131     }
132 }
133 }
134 };
135 // Enable multipart mode (need for uploads file)
136 form.setMultiPart(true);
137
138 // max upload size, 10k
139 //form.setMaxSize(Bytes.kilobytes(10));
140 fileUpload = new FileUploadField("fileUpload");
141 form.add(fileUpload);
142 fileUpload.setRequired(true);
143 description = new TextArea("description", new Model(""));
144 form.add(description);
145
146 borderBodyContainer.add(form);
147
148 //Add error items
149 PropertyModel errorMessageModel =
150     new PropertyModel(this, "errorMessage");
151 borderBodyContainer.add(new Label("error", errorMessageModel));
152 errorIconImage.setVisible(false);
153 borderBodyContainer.add(errorIconImage);
154 }
155
156 /**
157  * Set error message.
158  */
159 public void setErrorMessage(String errorMessage) {
160     this.errorMessage = errorMessage;
161     errorIconImage.setVisible(true);
162 }
163
164 /**
165  * Demo of mail to request user.
166  * TODO: Move this to an application thread.
167  *
168  * @param requester Application user that should receive a mail.
169  */
170 private void emailToRequester(Word word, VideoFile videoFile) {
171     WicketSession wicketSession = WicketSession.get();
172     if (!word.getRequestCreatedBy().getId().equals(
173         wicketSession.getApplicationUser().getId())) {
174         EmailSender emailSender = EmailSender.getInstance();
175         emailSender.sendNoReplyEmail(word.getRequestCreatedBy().getEmail(),
176             "Nyt forslag til " + word.getWord(),
177             createMailBody(word, videoFile));
178     }
179 }
180
181 private String createMailBody(Word word, VideoFile videoFile) {
182     String email = "<a href='mailto:jan.sch.hansen@gmail.com?'"
183         + "Subject=Spørgsmål til Tegn til tiden">"
184         + "jan.sch.hansen@gmail.com<a>";
185     StringBuilder mailBody = new StringBuilder();
186     mailBody.append("Der er kommet et nyt forslag til ordet: <b>");
187     mailBody.append(word.getWord());

```



```

188 mailBody.append("</b>.<br/>");
189 mailBody.append("Med følgende beskrivelse:<br/>");
190 mailBody.append(" - ");
191 mailBody.append(videoFile.getDescription());
192 mailBody.append("<br/><br/>");
193 mailBody.append("OBS! - Denne mail kan ikke besvares.<br/>");
194 mailBody.append("Eventuelle spørgsmål kan rettes til Jan Scrhøder Hansen på ");
195 mailBody.append("e-mail: ");
196 mailBody.append(email);
197 mailBody.append("<br/><br/>");
198 mailBody.append("Med venlig hilsen<br/>");
199 mailBody.append("Tegn til tiden");
200 return mailBody.toString();
201 }
202 }

```

File: CreateUser.html

```

1 <?xml version="1.0" encoding="UTF-8"?>
2 <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-
transitional.dtd">
3 <html xmlns:wicket="http://wicket.apache.org">
4 <head>
5   <meta http-equiv="Content-Type" content="text/html; charset=UTF-8"/>
6   <link rel="stylesheet" type="text/css" href="style.css"/>
7 </head>
8 <body>
9 <wicket:extend>
10 <span wicket:id="mainNavigation">
11   <div style="width: 450px">
12     <h2>Ny bruger</h2>
13     <form action="#" wicket:id="form">
14       <table border="0">
15         <tbody>
16           <tr>
17             <td style="width:120px">Navn:</td>
18             <td>
19               <input wicket:id="fullName" type="text"
20                 name="" value="" size="50" />
21             </td>
22           </tr>
23           <tr>
24             <td>Brugerkode:</td>
25             <td>
26               <input wicket:id="userlogin" type="text"
27                 name="" value="" size="20" />
28             </td>
29           </tr>
30           <tr>
31             <td>Password:</td>
32             <td>
33               <input wicket:id="password" type="password"
34                 name="" value="" size="20" />
35             </td>
36           </tr>
37           <tr>
38             <td>Gentag password:</td>
39             <td>
40               <input wicket:id="repeatPassword" type="password"
41                 name="" value="" size="20" />
42             </td>
43           </tr>
44           <tr>
45             <td>E-mail:</td>
46             <td>
47               <input wicket:id="email" type="text"
48                 name="" value="" size="50" />
49             </td>
50           </tr>
51           <tr>
52             <td colspan="2">&nbsp;</td>
53           </tr>
54           <tr>

```

```

55         <td colspan="2">
56             <div style="float:right">
57                 <input wicket:id="save" type="submit"
58                     value="Gem" />
59             </div>
60         </td>
61     </tr>
62 </tbody>
63 </table>
64 </form>
65 <div id="error">
66     <table align="center">
67         <tr>
68             <td><img wicket:id="erroricon"/></td>
69             <td><span wicket:id="error">Error message goes here</span></td>
70         </tr>
71     </table>
72 </div>
73 </div>
74 </span>
75 </wicket:extend>
76 </body>
77 </html>

```

File: CreateUser.java

```

1 package dk.jsh.itdiplom.userdrivesignlanguagedictionary.wicket.user;
2
3 import dk.jsh.itdiplom.userdrivesignlanguagedictionary.business.ApplicationUserBusiness;
4 import dk.jsh.itdiplom.userdrivesignlanguagedictionary.entity.ApplicationUser;
5 import dk.jsh.itdiplom.userdrivesignlanguagedictionary.entity.Constants;
6 import dk.jsh.itdiplom.userdrivesignlanguagedictionary.util.EMailSender;
7 import dk.jsh.itdiplom.userdrivesignlanguagedictionary.wicket.BasePage;
8 import dk.jsh.itdiplom.userdrivesignlanguagedictionary.wicket.homepage.MenuBorder;
9 import java.util.Date;
10 import org.apache.wicket.AttributeModifier;
11 import org.apache.wicket.PageParameters;
12 import org.apache.wicket.ResourceReference;
13 import org.apache.wicket.markup.html.basic.Label;
14 import org.apache.wicket.markup.html.border.Border.BorderBodyContainer;
15 import org.apache.wicket.markup.html.form.Button;
16 import org.apache.wicket.markup.html.form.Form;
17 import org.apache.wicket.markup.html.form.PasswordTextField;
18 import org.apache.wicket.markup.html.form.TextField;
19 import org.apache.wicket.markup.html.image.Image;
20 import org.apache.wicket.model.Model;
21 import org.apache.wicket.model.PropertyModel;
22 import org.apache.wicket.validation.validator.EmailAddressValidator;
23 import org.apache.wicket.validation.validator.StringValidator;
24
25 /**
26  * Create user page.
27  *
28  * @author Jan S. Hansen
29  */
30 public final class CreateUser extends BasePage {
31     private String errorMessage = "";
32     private TextField<String> fullName;
33     private TextField<String> userLogin;
34     private TextField<String> password;
35     private TextField<String> repeatPassword;
36     private TextField<String> email;
37     private Image errorIconImage = new Image("erroricon",
38         new ResourceReference(BasePage.class, "icons/attention.png"));
39
40     public CreateUser() {
41         MenuBorder menuBorder = new MenuBorder("mainNavigation");
42         add(menuBorder);
43         BorderBodyContainer borderBodyContainer = menuBorder.getBodyContainer();
44
45         //Add a form as an inner class.
46         Form form = new Form("form") {
47             //Handles required fields error.

```

```

48 @Override
49 protected void onError() {
50     if (emptyRequiredFields()) {
51         setErrorMessage("Alle felter skal udfyldes.");
52         return;
53     }
54
55     //Test for to short login and password
56     if (!userLogin.isValid()) {
57         setErrorMessage("Bruger kode skal være mindst 3 tegn langt");
58         userLogin.add(new AttributeModifier("style", true,
59             new Model("border-color:red;")));
60         return;
61     }
62     else {
63         userLogin.add(new AttributeModifier("style", true,
64             new Model("border-color:default;")));
65     }
66     if (!password.isValid()) {
67         setErrorMessage("Password skal være mindst 3 tegn langt");
68         password.add(new AttributeModifier("style", true,
69             new Model("border-color:red;")));
70         return;
71     }
72     else {
73         password.add(new AttributeModifier("style", true,
74             new Model("border-color:default;")));
75     }
76
77     //Test if it is a valid email
78     if (!email.isValid()) {
79         setErrorMessage("Email er ikke valid.");
80         email.add(new AttributeModifier("style", true,
81             new Model("border-color:red;")));
82         return;
83     }
84     else {
85         email.add(new AttributeModifier("style", true,
86             new Model("border-color:default;")));
87     }
88 }
89
90 private boolean emptyRequiredFields() {
91     //Test for empty/required fields
92     boolean emptyFields = false;
93     if (!fullName.checkRequired()) {
94         emptyFields = true;
95         fullName.add(new AttributeModifier("style", true,
96             new Model("border-color:red;")));
97     }
98     else {
99         fullName.add(new AttributeModifier("style", true,
100             new Model("border-color:default;")));
101     }
102     if (!userLogin.checkRequired()) {
103         emptyFields = true;
104         userLogin.add(new AttributeModifier("style", true,
105             new Model("border-color:red;")));
106     }
107     else {
108         userLogin.add(new AttributeModifier("style", true,
109             new Model("border-color:default;")));
110     }
111     if (!password.checkRequired()) {
112         emptyFields = true;
113         password.add(new AttributeModifier("style", true,
114             new Model("border-color:red;")));
115     }
116     else {
117         password.add(new AttributeModifier("style", true,
118             new Model("border-color:default;")));
119     }
120     if (!repeatPassword.checkRequired()) {
121         emptyFields = true;
122         repeatPassword.add(new AttributeModifier("style", true,

```

```

123         new Model("border-color:red;"));
124     }
125     else {
126         repeatPassword.add(new AttributeModifier("style", true,
127             new Model("border-color:default;")));
128     }
129     if (!email.checkRequired()) {
130         emptyFields = true;
131         email.add(new AttributeModifier("style", true,
132             new Model("border-color:red;")));
133     }
134     else {
135         email.add(new AttributeModifier("style", true,
136             new Model("border-color:default;")));
137     }
138     return emptyFields;
139 }
140 };
141 borderBodyContainer.add(form);
142
143 //Add fields to the form.
144 fullName = new TextField("fullName", new Model(""));
145 fullName.setRequired(true);
146 form.add(fullName);
147
148 userLogin = new TextField("userlogin", new Model(""));
149 userLogin.setRequired(true);
150 userLogin.add(StringValidator.minLength(3));
151 form.add(userLogin);
152
153 password = new PasswordTextField("password", new Model(""));
154 password.setRequired(true);
155 password.add(StringValidator.minLength(3));
156 form.add(password);
157
158 repeatPassword = new PasswordTextField("repeatPassword", new Model(""));
159 repeatPassword.setRequired(true);
160 repeatPassword.add(StringValidator.minLength(3));
161 form.add(repeatPassword);
162
163 email = new TextField("email", new Model(""));
164 email.add(EmailAddressValidator.getInstance());
165 email.setRequired(true);
166 form.add(email);
167
168 //Add button to the form.
169 form.add(new Button("save") {
170     @Override
171     public void onSubmit() {
172         //Test if password = repeat password
173         if (!password.getModelObject().equals(repeatPassword.getModelObject())) {
174             setErrorMessage("Password og Gentag password er ikke ens");
175             password.add(new AttributeModifier("style", true,
176                 new Model("border-color:red;")));
177             repeatPassword.add(new AttributeModifier("style", true,
178                 new Model("border-color:red;")));
179             return;
180         }
181         else {
182             password.add(new AttributeModifier("style", true,
183                 new Model("border-color:default;")));
184             repeatPassword.add(new AttributeModifier("style", true,
185                 new Model("border-color:default;")));
186         }
187
188         //Test if login is in use
189         if (ApplicationUserBusiness.isUserLoginInUse(
190             userLogin.getModelObject())) {
191             setErrorMessage("Bruger koden bruges af en anden bruger.");
192             userLogin.add(new AttributeModifier("style", true,
193                 new Model("border-color:red;")));
194             return;
195         }
196         else {
197             userLogin.add(new AttributeModifier("style", true,

```


Filen: EmailVerified.html

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-
transitional.dtd">
3 <html xmlns:wicket="http://wicket.apache.org">
4 <head>
5   <meta http-equiv="Content-Type" content="text/html; charset=UTF-8"/>
6   <link rel="stylesheet" type="text/css" href="style.css"/>
7 </head>
8 <body>
9 <wicket:extend>
10 <span wicket:id="mainNavigation">
11   <div style="width: 450px">
12     <h2>E-mail verificeret</h2>
13     <p>Velkommen til Tegn til tiden. Din e-mail adress er nu godkendt.
14       Og du kan nu logge på systemet</p>
15   </div>
16 </span>
17 </wicket:extend>
18 </body>
19 </html>
```

Filen: EmailVerified.java

```
1 package dk.jsh.itdiplom.userdrivesignlanguagedictionary.wicket.user;
2
3 import dk.jsh.itdiplom.userdrivesignlanguagedictionary.business.ApplicationUserBusiness;
4 import dk.jsh.itdiplom.userdrivesignlanguagedictionary.wicket.BasePage;
5 import dk.jsh.itdiplom.userdrivesignlanguagedictionary.wicket.homepage.MenuBorder;
6 import org.apache.wicket.PageParameters;
7
8 /**
9  * E-mail verified page
10  *
11  * @author Jan S. Hansen
12  */
13 public final class EmailVerified extends BasePage {
14
15     public EmailVerified(PageParameters params) {
16         MenuBorder menuBorder = new MenuBorder("mainNavigation");
17         add(menuBorder);
18         CharSequence charSequence = params.getCharSequence("login");
19         ApplicationUserBusiness.setEmailVerified(charSequence.toString());
20     }
21 }
```

Filen: UserCreated.html

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-
transitional.dtd">
3 <html xmlns:wicket="http://wicket.apache.org">
4 <head>
5   <meta http-equiv="Content-Type" content="text/html; charset=UTF-8"/>
6   <link rel="stylesheet" type="text/css" href="style.css"/>
7 </head>
8 <body>
9 <wicket:extend>
10 <span wicket:id="mainNavigation">
11   <div style="width: 450px">
12     <h2>Bruger oprettet</h2>
13     <p>Dine bruger oplysninger er nu gemt. Der er sendt en e-mail til den
14       e-mail adresse du har angivet. I e-mailen er der et link, til dette system,
15       som skal aktiveres før du kan logge på systemet første gang.</p>
16   </div>
17 </span>
18 </wicket:extend>
19 </body>
20 </html>
```

Filen: UserCreated.java

```

1 package dk.jsh.itdiplom.userdrivesignlanguagedictionary.wicket.user;
2
3 import dk.jsh.itdiplom.userdrivesignlanguagedictionary.wicket.BasePage;
4 import dk.jsh.itdiplom.userdrivesignlanguagedictionary.wicket.homepage.MenuBorder;
5
6 /**
7  * User created page.
8  *
9  * @author Jan S. Hansen
10 */
11 public final class UserCreated extends BasePage {
12
13     public UserCreated() {
14         MenuBorder menuBorder = new MenuBorder("mainNavigation");
15         add(menuBorder);
16     }
17 }

```

Filen: SelectedVideo.html

```

1 <?xml version="1.0" encoding="UTF-8"?>
2 <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-
transitional.dtd">
3 <html xmlns:wicket="http://wicket.apache.org">
4 <head>
5     <meta http-equiv="Content-Type" content="text/html; charset=UTF-8"/>
6     <link rel="stylesheet" type="text/css" href="style.css"/>
7 </head>
8 <body>
9 <wicket:extend>
10 <span wicket:id="mainNavigation">
11     <h2>Forslag til: <span wicket:id="word">[word]</span></h2>
12     <table border="0">
13         <tr>
14             <td>
15                 <div style="float:right">
16                     <a href="#" wicket:id="back">Tilbage</a>
17                 </div>
18             </td>
19         </tr>
20         <tr>
21             <td>
22                 <fieldset>
23                     <legend>Beskrivelse af ordet</legend>
24                     <span wicket:id="wordDescription">[WordDescription]</span>
25                 </fieldset>
26             </td>
27         </tr>
28         <tr>
29             <td align="center">
30                 <video wicket:id="video" width="600">
31                     Din browser understøtter ikke HTML5 Video.
32                 </video>
33             </td>
34         </tr>
35         <tr>
36             <td>
37                 <fieldset>
38                     <legend>Beskrivelse af video</legend>
39                     <span wicket:id="videoDescription">[VideoDescription]</span>
40                 </fieldset>
41             </td>
42         </tr>
43     </table>
44 </span>
45 </wicket:extend>
46 </body>
47 </html>

```

File: SelectedVideo.java

```

1 package dk.jsh.itdiplom.userdrivesignlanguagedictionary.wicket.video;
2
3 import dk.jsh.itdiplom.userdrivesignlanguagedictionary.entity.VideoFile;
4 import dk.jsh.itdiplom.userdrivesignlanguagedictionary.entity.Word;
5 import dk.jsh.itdiplom.userdrivesignlanguagedictionary.wicket.Application;
6 import dk.jsh.itdiplom.userdrivesignlanguagedictionary.wicket.BasePage;
7 import dk.jsh.itdiplom.userdrivesignlanguagedictionary.wicket.about.About;
8 import dk.jsh.itdiplom.userdrivesignlanguagedictionary.wicket.homepage.MenuBorder;
9 import dk.jsh.itdiplom.userdrivesignlanguagedictionary.wicket.html5.Html5Video;
10 import dk.jsh.itdiplom.userdrivesignlanguagedictionary.wicket.html5.VideoSource;
11 import dk.jsh.itdiplom.userdrivesignlanguagedictionary.wicket.word.SelectedWord;
12 import java.util.ArrayList;
13 import java.util.List;
14 import org.apache.wicket.Page;
15 import org.apache.wicket.ResourceReference;
16 import org.apache.wicket.markup.html.basic.Label;
17 import org.apache.wicket.markup.html.border.Border.BorderBodyContainer;
18 import org.apache.wicket.markup.html.link.Link;
19 import org.apache.wicket.model.AbstractReadOnlyModel;
20 import org.apache.wicket.model.IModel;
21
22
23 /**
24  * Page to show a video
25  *
26  * @author Jan S. Hansen
27  */
28 public final class SelectedVideo extends BasePage {
29
30     public SelectedVideo(final Word word, final VideoFile videoFile) {
31         MenuBorder menuBorder = new MenuBorder("mainNavigation");
32         add(menuBorder);
33         BorderBodyContainer borderBodyContainer = menuBorder.getBodyContainer();
34
35         Link back = new Link("back") {
36             @Override
37             public void onClick() {
38                 Page page = new SelectedWord(word);
39                 setResponsePage(page);
40             }
41         };
42         borderBodyContainer.add(back);
43
44         borderBodyContainer.add(new Label("word", word.getWord()));
45         borderBodyContainer.add(new Label("wordDescription", word.getDescription()));
46         borderBodyContainer.add(new Label("videoDescription", videoFile.getDescription()));
47
48         final List<VideoSource> videoSources = new ArrayList<VideoSource>();
49         videoSources.add(new VideoSource(new ResourceReference(Application.class,
50             "uploadedvideo/" + videoFile.getResourceName()),
51             VideoSource.VideoType.OGG));
52
53         IModel<List<VideoSource>> videoSourceList =
54             new AbstractReadOnlyModel<List<VideoSource>>() {
55                 @Override
56                 public List<VideoSource> getObject() {
57                     return videoSources;
58                 }
59             };
60         Html5Video html5Video = new Html5Video("video", videoSourceList);
61
62         borderBodyContainer.add(html5Video);
63     }
64 }
65

```


File: SelectedWord.html

```

1 <?xml version="1.0" encoding="UTF-8"?>
2 <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-
transitional.dtd">
3 <html xmlns:wicket="http://wicket.apache.org">
4 <head>
5   <meta http-equiv="Content-Type" content="text/html; charset=UTF-8"/>
6   <link rel="stylesheet" type="text/css" href="style.css"/>
7 </head>
8 <body>
9 <wicket:extend>
10 <span wicket:id="mainNavigation">
11   <h2>Ord: <span wicket:id="word">[word]</span></h2>
12   <table border="0">
13     <tr>
14       <td>
15         <fieldset>
16           <legend>Beskrivelse</legend>
17           <span wicket:id="description">[Description]</span>
18         </fieldset>
19       </td>
20     </tr>
21     <tr>
22       <td>
23         <div style="float:right">
24           <a href="#" wicket:id="uploadLink">Upload forslag</a>
25         </div>
26       </td>
27     </tr>
28     <tr>
29       <td>
30         <fieldset>
31           <legend>Forslag</legend>
32           <div wicket:id="feedback"></div>
33           <div wicket:id="pageable">
34             <table border="0" width="100%">
35               <tr>
36                 <td>
37                   Uploadet af
38                   <span wicket:id="byUser">[user]</span>
39                   den
40                   <span wicket:id="dateTime">[dato/tid]</span>
41                 </td>
42                 <td>&nbsp;</td>
43                 <td align="right">
44                   <a href="#" wicket:id="videoLink" title="Vis video">Vis</a>
45                 </td>
46               </tr>
47             </table>
48           </div>
49         </fieldset>
50       </td>
51     </tr>
52   </table>
53   <div style="float:right">
54     <span wicket:id="navigator">[dataview navigator]</span>
55   </div>
56 </span>
57 </wicket:extend>
58 </body>
59 </html>

```

File: SelectedWord.java

```

1 package dk.jsh.itdiplom.userdrivensignlanguagedictionary.wicket.word;
2
3 import dk.jsh.itdiplom.userdrivensignlanguagedictionary.business.VideoFileBusiness;
4 import dk.jsh.itdiplom.userdrivensignlanguagedictionary.entity.VideoFile;
5 import dk.jsh.itdiplom.userdrivensignlanguagedictionary.wicket.BasePage;
6 import dk.jsh.itdiplom.userdrivensignlanguagedictionary.entity.Word;
7 import dk.jsh.itdiplom.userdrivensignlanguagedictionary.wicket.WicketSession;
8 import dk.jsh.itdiplom.userdrivensignlanguagedictionary.wicket.homepage.MenuBorder;

```

```

9 import dk.jsh.itdiplom.userdrivesignlanguagedictionary.wicket.upload.Upload;
10 import dk.jsh.itdiplom.userdrivesignlanguagedictionary.wicket.video.SelectedVideo;
11 import java.util.List;
12 import org.apache.wicket.AttributeModifier;
13 import org.apache.wicket.Page;
14 import org.apache.wicket.markup.html.basic.Label;
15 import org.apache.wicket.markup.html.border.Border.BorderBodyContainer;
16 import org.apache.wicket.markup.html.link.Link;
17 import org.apache.wicket.markup.html.list.ListItem;
18 import org.apache.wicket.markup.html.list.PageableListView;
19 import org.apache.wicket.markup.html.navigation.paging.PagingNavigator;
20 import org.apache.wicket.markup.html.panel.FeedbackPanel;
21 import org.apache.wicket.model.AbstractReadOnlyModel;
22
23 /**
24  * Word page.
25  *
26  * @author Jan S. Hansen
27  */
28 public final class SelectedWord extends BasePage {
29
30     public SelectedWord(final Word word) {
31         MenuBorder menuBorder = new MenuBorder("mainNavigation");
32         add(menuBorder);
33         BorderBodyContainer borderBodyContainer = menuBorder.getBodyContainer();
34         borderBodyContainer.add(new Label("word", word.getWord()));
35         borderBodyContainer.add(new Label("description", word.getDescription()));
36         Link uploadLink = new Link("uploadLink") {
37             @Override
38             public void onClick() {
39                 Page page = new Upload(word);
40                 setResponsePage(page);
41             }
42         };
43         WicketSession wicketSession = WicketSession.get();
44         if (!wicketSession.isAuthenticated()) {
45             uploadLink.setVisible(false);
46         }
47         borderBodyContainer.add(uploadLink);
48         borderBodyContainer.add(new FeedbackPanel("feedback"));
49         List<VideoFile> videoFileList =
50             VideoFileBusiness.getAllVideoFilesForAWord(word);
51         if (videoFileList.size() == 0) {
52             info("Ingen forslag er uploadet.");
53         }
54         PageableListView pageableListView =
55             new PageableListView("pageable", videoFileList, 5) {
56             @Override
57             protected void populateItem(final ListItem item) {
58                 final VideoFile videoFile = (VideoFile) item.getModelObject();
59                 item.add(new Label("byUser",
60                     videoFile.getUploadedBy().getFullname()));
61                 item.add(new Label("dateTime",
62                     standardDateFormat.format(videoFile.getUploadedDateTime())));
63                 Link videoLink = new Link("videoLink") {
64                     @Override
65                     public void onClick() {
66                         Page page = new SelectedVideo(word, videoFile);
67                         setResponsePage(page);
68                     }
69                 };
70                 item.add(videoLink);
71                 item.add(new AttributeModifier("class",
72                     true, new AbstractReadOnlyModel<String>() {
73                         @Override
74                         public String getObject()
75                         {
76                             return (item.getIndex() % 2 == 1) ? "even" : "odd";
77                         }
78                     }));
79             }
80         };
81         borderBodyContainer.add(pageableListView);
82         borderBodyContainer.add(new PagingNavigator("navigator",
83             pageableListView));

```

84 }
85 }

Pakken: dk.jsh.itdiplom.userdrivensignlanguagedictionary.util

Filen: ConvertVideo.java

```
1 package dk.jsh.itdiplom.userdrivensignlanguagedictionary.util;
2
3 import java.io.BufferedReader;
4 import java.io.IOException;
5 import java.io.InputStream;
6 import java.io.InputStreamReader;
7 import java.util.Date;
8 import java.util.logging.Level;
9 import java.util.logging.Logger;
10
11 /**
12  * Convert video files to the OGG video format.
13  *
14  * @author Jan S. Hansen
15  */
16 public class ConvertVideo {
17     static final Logger logger = Logger.getLogger(ConvertVideo.class.getName());
18
19     /**
20      * Convert video file til ogg file format.
21      *
22      * @param source Source file name
23      * @param dest Destination file name
24      */
25     public boolean convert(String source, String dest) {
26         Runtime runtime = Runtime.getRuntime();
27         //TODO: location of ffmpeg2theora-0.28.exe in a properties file
28         StringBuilder cmdLine = new StringBuilder();
29         cmdLine.append("\\GoogleCode\\user-driven-sign-language-dictionary");
30         cmdLine.append("\\ffmpeg2theora-0.28.exe ");
31         cmdLine.append("-o ");
32         cmdLine.append(dest);
33         cmdLine.append(" ");
34         cmdLine.append(source);
35         try {
36             logger.info("Convert video: " + cmdLine.toString());
37             Process process = runtime.exec(cmdLine.toString());
38
39             //Start thread to read standard error
40             ProcessOutput err = new ProcessOutput(process.getErrorStream(),
41                 "ERR");
42             Thread errThread = new Thread(err);
43             errThread.start();
44
45             //Start thread to read standard output
46             ProcessOutput std = new ProcessOutput(process.getInputStream(),
47                 "STD");
48             Thread stdThread = new Thread(std);
49             stdThread.start();
50
51             process.waitFor();
52         } catch (Exception exception) {
53             logger.log(Level.SEVERE, "Error converting video", exception);
54             return false;
55         }
56         return true;
57     }
58
59     /**
60      * Create an OGV wicket resource name. Format UserId_xx_wordId_yy_mm.ogv.
61      * xx = user Id (Database PK), yy = word Id and mm = milliseconds since
62      * 1/1-1970.
63      *
64      * @param userId User id
65      * @param wordId word id
66      * @return a OGV wicket resource name.
67      */
68     public String createOgvResourceName(String userId, String wordId) {
69         StringBuilder sb = new StringBuilder("UserId_");
70         sb.append(userId);
```

```

71     sb.append("_wordId_");
72     sb.append(wordId);
73     sb.append("_");
74     Date now = new Date();
75     sb.append(now.getTime());
76     sb.append(".ogv");
77     return sb.toString();
78 }
79
80 /**
81  * Creates an OGV filename with full path.
82  *
83  * @param resourceName resource name
84  * @return filename
85  */
86 public String createOgvFilename(String resourceName) {
87     //TODO: Get full path, should not be hard coded.
88     StringBuilder sb = new StringBuilder("C:\\GoogleCode\\"
89         + "user-driven-sign-language-dictionary\\Code\\"
90         + "UserDrivenSignLanguageDictionary\\build\\web\\"
91         + "WEB-INF\\classes\\dk\\jsh\\itdiplom\\"
92         + "userdrivensignlanguagedictionary\\wicket\\uploadedvideo\\");
93     sb.append(resourceName);
94     return sb.toString();
95 }
96
97 private class ProcessOutput implements Runnable {
98     private InputStream inputStream;
99     private String type;
100
101     public ProcessOutput(InputStream inputStream, String type) {
102         this.inputStream = inputStream;
103         this.type = type;
104     }
105
106     @Override
107     public void run() {
108         BufferedReader procesOutput = new BufferedReader(new
109             InputStreamReader(inputStream));
110         String line = null;
111         try {
112             while ((line = procesOutput.readLine()) != null) {
113                 logger.info(type + ": " + line);
114             }
115         } catch (IOException ex) {
116             logger.log(Level.SEVERE, "Error reading output of type " + type,
117                 ex);
118         }
119     }
120 }
121 }
122

```

File: EMailSender.java

```

1 package dk.jsh.itdiplom.userdrivensignlanguagedictionary.util;
2
3 import java.util.Properties;
4 import javax.mail.Message;
5 import javax.mail.MessagingException;
6 import javax.mail.Session;
7 import javax.mail.Transport;
8 import javax.mail.internet.InternetAddress;
9 import javax.mail.internet.MimeMessage;
10
11 /**
12  * E-Mail sender. Singleton pattern.
13  *
14  * @author Jan S. Hansen
15  */
16 public class EMailSender {
17     private static Properties emailProperties = new Properties();
18     private static EMailSender singletonInstance;

```

```

19
20 /**
21  * Private constructor to prevent use of new keyword outside this class.
22  */
23 private EmailSender(){ };
24
25 public static EmailSender getInstance() {
26     if (singletonInstance == null) {
27         //TODO: Get host from a property file.
28         emailProperties.put("mail.smtp.host", "localhost");
29         singletonInstance = new EmailSender();
30     }
31     return singletonInstance;
32 }
33
34 /**
35  * Send an e-mail.
36  *
37  * @param fromEmailAddr from e-mail address
38  * @param toEmailAddr to e-mail address
39  * @param subject e-mail subject
40  * @param body e-mail body text
41  * @return true if no errors occurs.
42  */
43 public boolean sendEmail(String fromEmailAddr, String toEmailAddr,
44     String subject, String body) {
45     Session session = Session.getDefaultInstance(emailProperties, null);
46     MimeMessage message = new MimeMessage(session);
47     try {
48         message.setFrom(new InternetAddress(fromEmailAddr));
49         message.addRecipient(Message.RecipientType.TO, new InternetAddress(
50             toEmailAddr));
51
52         message.setSubject(subject);
53         message.setContent(body, "text/html");
54         Transport.send(message);
55         return true;
56     } catch (MessagingException ex) {
57         System.err.println("Cannot send email. " + ex);
58         return false;
59     }
60 }
61
62 /**
63  * Send a no-reply e-mail.
64  *
65  * @param toEmailAddr to e-mail address
66  * @param subject e-mail subject
67  * @param body e-mail body text
68  * @return true if no errors occurs.
69  */
70 public boolean sendNoReplyEmail(String toEmailAddr, String subject,
71     String body) {
72     return sendEmail("NoReply@TegnTilTiden.dk", toEmailAddr, subject, body);
73 }
74 }
75

```

File: HibernateUtil.java

```

1 package dk.jsh.itdiplom.userdrivensignlanguagedictionary.util;
2
3 import org.hibernate.*;
4 import org.hibernate.cfg.*;
5
6 /**
7  * Hibernate session factory.
8  *
9  * @author Jan S. Hansen
10 */
11 public class HibernateUtil {
12     private static SessionFactory sessionFactory;
13

```

```

14 static {
15     try {
16         sessionFactory =
17             new AnnotationConfiguration().configure().buildSessionFactory();
18     }
19     catch (Throwable ex) {
20         throw new ExceptionInInitializerError(ex);
21     }
22 }
23
24 /**
25  * Get a Hibernate session factory.
26  *
27  * @return a SessionFactory
28  */
29 public static SessionFactory getSessionFactory() {
30     return sessionFactory;
31 }
32
33 /**
34  * Close SessionFactory.
35  */
36 public static void shutdown() {
37     getSessionFactory().close();
38 }
39 }
40

```

Filen: Text.java

```

1 package dk.jsh.itdiplom.userdrivensignlanguagedictionary.util;
2
3 import java.util.List;
4
5 /**
6  * Text utilities.
7  *
8  * @author Jan S. Hansen
9  */
10 public class Text {
11
12     /**
13      * Make at list of word groups.
14      *
15      * @param wordGroupList a Word group list
16      * @return a string of wordGroups
17      */
18     public static String makeWordGroupString(List<String> wordGroupList) {
19         StringBuilder groups = new StringBuilder();
20         int noOfGroups = wordGroupList.size();
21         if (noOfGroups > 0) {
22             for (int i = 0; i < noOfGroups; i++) {
23                 String wordGroup = wordGroupList.get(i);
24                 if (i > 0 && i < noOfGroups - 1) {
25                     groups.append(", ");
26                 }
27                 else if (i == noOfGroups - 1) {
28                     groups.append(" og ");
29                 }
30                 groups.append(wordGroup);
31             }
32         }
33         else {
34             groups.append("Ikke tilknyttet nogen gruppe");
35         }
36         groups.append(".");
37         return groups.toString();
38     }
39 }
40

```

Diverse scripts og setup filer.

Filen: hibernate.cft.xml

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <!DOCTYPE hibernate-configuration PUBLIC "-//Hibernate/Hibernate Configuration DTD 3.0//EN"
   "http://hibernate.sourceforge.net/hibernate-configuration-3.0.dtd">
3 <hibernate-configuration>
4   <session-factory>
5     <property name="hibernate.dialect">org.hibernate.dialect.DerbyDialect</property>
6     <property name="hibernate.connection.driver_class">org.apache.derby.jdbc.ClientDriver</property>
7     <property name="hibernate.connection.url">jdbc:derby://localhost:1527/sample</property>
8     <property name="hibernate.connection.username">app</property>
9     <property name="hibernate.connection.password">app</property>
10
11
12     <property name="hibernate.dialect">org.hibernate.dialect.DerbyDialect</property>
13     <property name="hibernate.connection.driver_class">org.apache.derby.jdbc.ClientDriver</property>
14     <property name="hibernate.connection.url">jdbc:derby://localhost:1527/UDSLD</property>
15     <property name="hibernate.connection.username">app</property>
16     <property name="hibernate.connection.password">app</property>
17
18     <!-- Show and print nice SQL on stdout -->
19     <property name="hibernate.show_sql">true</property>
20     <property name="hibernate.format_sql">true</property>
21
22     <!-- List of annotated classes -->
23     <mapping class="dk.jsh.itdiplom.userdrivesignlanguagedictionary.entity.ApplicationUser" />
24     <mapping class="dk.jsh.itdiplom.userdrivesignlanguagedictionary.entity.WordGroup" />
25     <mapping class="dk.jsh.itdiplom.userdrivesignlanguagedictionary.entity.Word" />
26     <mapping class="dk.jsh.itdiplom.userdrivesignlanguagedictionary.entity.WordGroupWordRelation" />
27     <mapping class="dk.jsh.itdiplom.userdrivesignlanguagedictionary.entity.VideoFile" />
28
29   </session-factory>
30 </hibernate-configuration>
31
32
```

Filen: build.xml

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <!-- You may freely edit this file. See commented blocks below for -->
3 <!-- some examples of how to customize the build. -->
4 <!-- (If you delete it and reopen the project it will be recreated.) -->
5 <!-- By default, only the Clean and Build commands use this build script. -->
6 <!-- Commands such as Run, Debug, and Test only use this build script if -->
7 <!-- the Compile on Save feature is turned off for the project. -->
8 <!-- You can turn off the Compile on Save (or Deploy on Save) setting -->
9 <!-- in the project's Project Properties dialog box.-->
10 <project name="UserDrivenSignLanguageDictionary" default="default" basedir=".">
11   <description>Builds, tests, and runs the project UserDrivenSignLanguageDictionary.</description>
12   <import file="nbproject/build-impl.xml"/>
13   <!--
14
15   There exist several targets which are by default empty and which can be
16   used for execution of your tasks. These targets are usually executed
17   before and after some main targets. They are:
18
19   -pre-init:         called before initialization of project properties
20   -post-init:        called after initialization of project properties
21   -pre-compile:      called before javac compilation
22   -post-compile:     called after javac compilation
23   -pre-compile-single: called before javac compilation of single file
24   -post-compile-single: called after javac compilation of single file
25   -pre-compile-test:  called before javac compilation of JUnit tests
26   -post-compile-test: called after javac compilation of JUnit tests
27   -pre-compile-test-single: called before javac compilation of single JUnit test
28   -post-compile-test-single: called after javac compilation of single JUnit test
29   -pre-dist:         called before archive building
30   -post-dist:        called after archive building
31   -post-clean:       called after cleaning build products
32   -pre-run-deploy:   called before deploying
```



```

33 -post-run-deploy:      called after deploying
34
35 Example of plugging an obfuscator after the compilation could look like
36
37 <target name="-post-compile">
38   <obfuscate>
39     <fileset dir="${build.classes.dir}"/>
40   </obfuscate>
41 </target>
42
43 For list of available properties check the imported
44 nbproject/build-impl.xml file.
45
46
47 Other way how to customize the build is by overriding existing main targets.
48 The target of interest are:
49
50 init-macrodef-javac:  defines macro for javac compilation
51 init-macrodef-junit:  defines macro for junit execution
52 init-macrodef-debug:  defines macro for class debugging
53 do-dist:              archive building
54 run:                  execution of project
55 javadoc-build:        javadoc generation
56
57 Example of overriding the target for project execution could look like
58
59 <target name="run" depends="<PROJNAME>-impl.jar">
60   <exec dir="bin" executable="launcher.exe">
61     <arg file="${dist.jar}"/>
62   </exec>
63 </target>
64
65 Notice that overridden target depends on jar target and not only on
66 compile target as regular run target does. Again, for list of available
67 properties which you can use check the target you are overriding in
68 nbproject/build-impl.xml file.
69
70 -->
71
72 <target name="schemaexport">
73
74   <path id="project.classpath">
75     <fileset dir="build/web/WEB-INF/lib">
76       <include name="**/*.jar"/>
77       <include name="**/*.zip"/>
78     </fileset>
79   </path>
80
81   <taskdef name="hibernatetool"
82     classname="org.hibernate.tool.ant.HibernateToolTask"
83     classpathref="project.classpath"/>
84
85   <hibernatetool destdir=".">
86     <classpath path="build/web/WEB-INF/classes" />
87     <annotationconfiguration configurationfile="src/java/hibernate.cfg.xml" />
88     <hbm2ddl
89       drop="true"
90       create="true"
91       export="true"
92       outputfilename="ddl.sql"
93       delimiter=";"
94       format="true" />
95   </hibernatetool>
96 </target>
97 </project>
98
99

```

File: ddl.sql

```
1
2 alter table VideoFile
3     drop constraint fk_file_applicationuser;
4
5 alter table VideoFile
6     drop constraint fk_file_word;
7
8 alter table Word
9     drop constraint fk_word_applicationuser;
10
11 alter table WordGroup
12     drop constraint fk_wordgroup_applicationuser;
13
14 alter table WordGroupWordRelation
15     drop constraint fk_wordgroupwordrelation_wordgroup;
16
17 alter table WordGroupWordRelation
18     drop constraint fk_wordgroupwordrelation_word;
19
20 drop table ApplicationUser;
21
22 drop table VideoFile;
23
24 drop table Word;
25
26 drop table WordGroup;
27
28 drop table WordGroupWordRelation;
29
30 create table ApplicationUser (
31     id bigint not null generated always as identity,
32     email varchar(50) not null,
33     emailVerificationSent timestamp,
34     emailVerified timestamp,
35     fullname varchar(50) not null,
36     login varchar(20) not null unique,
37     password varchar(20) not null,
38     userRole varchar(10) not null,
39     version integer not null,
40     primary key (id)
41 );
42
43 create table VideoFile (
44     id bigint not null generated always as identity,
45     description varchar(250),
46     fileName varchar(100) not null,
47     resourceName varchar(50) not null,
48     uploadedDateTime timestamp not null,
49     version integer not null,
50     toWord_id bigint not null,
51     uploadedBy_id bigint not null,
52     primary key (id)
53 );
54
55 create table Word (
56     id bigint not null generated always as identity,
57     createdDateTime timestamp not null,
58     description varchar(250),
59     version integer not null,
60     word varchar(50) not null unique,
61     requestCreatedBy_id bigint not null,
62     primary key (id)
63 );
64
65 create table WordGroup (
66     id bigint not null generated always as identity,
67     createdDateTime timestamp not null,
68     description varchar(250),
69     name varchar(30) not null unique,
70     version integer not null,
71     createdBy_id bigint not null,
72     primary key (id)
73 );
```

```

74
75 create table WordGroupWordRelation (
76     id bigint not null generated always as identity,
77     version integer not null,
78     word_id bigint not null,
79     wordGroup_id bigint not null,
80     primary key (id),
81     unique (wordGroup_id, word_id)
82 );
83
84 alter table VideoFile
85     add constraint fk_file_applicationuser
86     foreign key (uploadedBy_id)
87     references ApplicationUser;
88
89 alter table VideoFile
90     add constraint fk_file_word
91     foreign key (toWord_id)
92     references Word;
93
94 alter table Word
95     add constraint fk_word_applicationuser
96     foreign key (requestCreatedBy_id)
97     references ApplicationUser;
98
99 alter table WordGroup
100     add constraint fk_wordgroup_applicationuser
101     foreign key (createdBy_id)
102     references ApplicationUser;
103
104 alter table WordGroupWordRelation
105     add constraint fk_wordgroupwordrelation_wordgroup
106     foreign key (wordGroup_id)
107     references WordGroup;
108
109 alter table WordGroupWordRelation
110     add constraint fk_wordgroupwordrelation_word
111     foreign key (word_id)
112     references Word;
113
114

```

File: db_init.sql

```

1 -- Extra constraints
2 alter table ApplicationUser
3 add constraint valid_user_roles
4 check (userrole in ('ADMIN', 'NORMAL'));
5
6 alter table ApplicationUser
7 add constraint password_length_ge_3
8 check (length(password) >= 3);
9
10 -- Test data
11 -----
12 -- Users
13 insert into applicationuser
14 (login, password, userrole, fullname, email, emailverificationsent,
15  emailverified, version)
16 values
17 ('jsh', 'jsh', 'NORMAL', 'Jan Schröder Hansen', 'jsh@jsh.dk',
18  '2011-10-29 10:10:10', '2011-10-29 10:10:12', 1);
19
20 insert into applicationuser
21 (login, password, userrole, fullname, email, emailverificationsent,
22  emailverified, version)
23 values
24 ('tki', 'tki', 'NORMAL', 'Tanja Kikkenborg', 'tki@tki.dk',
25  '2011-10-29 10:10:10', '2011-10-29 10:10:12', 1);
26
27

```

9.5. INDHOLD PÅ DEN VEDLAGTE CD

Indholdet på den vedlagte CD er inddelt i følgende 3 kataloger:

- Rapport – Indeholder denne rapport i Word 2007 og PDF format.
- MagicDraw – Indeholder 3 MagicDraw projekter. Et for analyse, et for Design og implementering samt et for webside brugerflade design.
- Kode – Indeholder al kode til projektet.

De fleste af mine noter, henviser (linker) til den engelske Wikipedia, da den engelske version af Wikipedia, ofte er mere beskrivende end den danske, og fordi Wikipedia som regel er neutral i sine beskrivelser.

-
- ¹ Java – Programmeringssprog, se en.wikipedia.org/wiki/Java_%28programming_language%29.
 - ² Apache Wicket - Java Web framework, se en.wikipedia.org/wiki/Apache_Wicket.
 - ³ Hibernate - Java Object til database framework, se en.wikipedia.org/wiki/Hibernate.
 - ⁴ JavaDB - Java Database, som er en del af standard java, se en.wikipedia.org/wiki/Apache_Derby.
 - ⁵ Apache Tomcat – Web server, se en.wikipedia.org/wiki/Apache_Tomcat.
 - ⁶ UML – Unified Modeling Language, se en.wikipedia.org/wiki/Unified_Modeling_Language.
 - ⁷ Use case – System/krav beskrivelser, se en.wikipedia.org/wiki/Use_case_diagram.
 - ⁸ UP – Unified Process, se en.wikipedia.org/wiki/Unified_Process.
 - ⁹ MagicDraw – UML tegneprogram, se en.wikipedia.org/wiki/MagicDraw.
 - ¹⁰ Java Annotations – mærkning af java klasser, se en.wikipedia.org/wiki/Java_annotation.
 - ¹¹ DDL – Data Definition Language, se en.wikipedia.org/wiki/Data_Definition_Language.
 - ¹² Optimistisk låsning, se http://en.wikipedia.org/wiki/Optimistic_concurrency_control.
 - ¹³ Singleton – Design mønster, som sikre at der kun kan findes en instans af klassen, se en.wikipedia.org/wiki/Singleton_pattern.
 - ¹⁴ Programtråd – en parallel program tråd, som kører på samme tid som hovedtråden, se [en.wikipedia.org/wiki/Thread_\(computer_science\)](http://en.wikipedia.org/wiki/Thread_(computer_science)).
 - ¹⁵ FFMPEG – et video format konverteringsprogram, se ffmpeg.org.
 - ¹⁶ Apache James – Email server, se james.apache.org.
 - ¹⁷ Why Wicket, se wicket.apache.org/introduction.html.
 - ¹⁸ Struts - Web framework, se struts.apache.org.
 - ¹⁹ ASP.NET – Web framework, se en.wikipedia.org/wiki/ASP.NET.
 - ²⁰ JSP – Java Server Pages, se en.wikipedia.org/wiki/JSP.
 - ²¹ HTML – Hyper Text Markup Language, se en.wikipedia.org/wiki/HTML.
 - ²² JavaScript – programmeringssprog, se en.wikipedia.org/wiki/JavaScript.
 - ²³ C# - C Sharp programmeringssprog, se en.wikipedia.org/wiki/C_Sharp_programming_language.
 - ²⁴ CSS – Cascading Style Sheets, se http://en.wikipedia.org/wiki/Cascading_Style_Sheets.
 - ²⁵ HQL - Hibernate Query Language, se docs.jboss.org/hibernate/core/3.3/reference/en/html/queryhql.html.
 - ²⁶ SQL - Structured Query Language, se en.wikipedia.org/wiki/SQL.
 - ²⁷ Wicket secure by default - se mere wicket.apache.org/meet/features.html.
 - ²⁸ Se OWASP Top 10, se www.owasp.org/index.php/Top_10_2007.
 - ²⁹ HTTPS/SSL, se en.wikipedia.org/wiki/Transport_Layer_Security.
 - ³⁰ Password Hashed – se www.owasp.org/index.php/Guide_to_Authentication#Password_Guidelines.
 - ³¹ WAR – Web application ARchive, se [en.wikipedia.org/wiki/WAR_file_format_\(Sun\)](http://en.wikipedia.org/wiki/WAR_file_format_(Sun)).
 - ³² SCRUM - Udviklingsproces, se [en.wikipedia.org/wiki/Scrum_\(development\)](http://en.wikipedia.org/wiki/Scrum_(development)).
 - ³³ XP – eXtreme Programming, se en.wikipedia.org/wiki/Extreme_programming.
 - ³⁴ CMS – Content Management System, se en.wikipedia.org/wiki/Content_management_system.
 - ³⁵ OGG – Videoformat, se en.wikipedia.org/wiki/Ogg.
 - ³⁶ MPEG-4 – Videoformat, se en.wikipedia.org/wiki/MPEG-4_Part_14.
 - ³⁷ Adobe Flash – Multimedieplatform fra Adobe, kan bl.a. vise video, se en.wikipedia.org/wiki/Adobe_Flash.
 - ³⁸ NetBeans – Java udviklingsmiljø, se en.wikipedia.org/wiki/NetBeans.
 - ³⁹ Apache SubVersion – kode versionerings system, se en.wikipedia.org/wiki/Apache_Subversion.
 - ⁴⁰ GoogleCode - Google service der gratis hoster udviklingsprojekter, mod at de udgives som open source, se en.wikipedia.org/wiki/Google_Code.
 - ⁴¹ Apache Ant – Software bygge værktøj, se en.wikipedia.org/wiki/Apache_Ant.