BACHELOROPPGAVE:

NTWEAR15

FORFATTERE:
Lars Bendik Dølvik
John Christian G. Fjeld

Dato:
15. Mai 2015

SAMMENDRAG

Tittel:	NTwear15	Dato : 15. Mai 2015
Deltakere:	Lars Bendik Dølvik John Christian Fjeld	
Veileder:	Ivar Farup	
Oppdragsgiver:	Jørn Berg Nordlund, Norsk Tipping	
Stikkord/nøkkelord (3-5 stk)	PHP, Javascript, SQL, Applikasjon, Database	
Antall sider: 81(114)	Antall vedlegg: 10	Tilgjengelighet (åpen/konfidensiell): Åpen

Kort beskrivelse av bacheloroppgaven:

Vi har laget en webapplikasjon for Norsk Tipping. Denne applikasjonen skal hjelpe Norsk Tipping i å aktivisere og sosialisere grasrotlagenes medlemmer.

Ved å bruke vår applikasjon har man tjene opp grasrotcoins ved å være aktiv og sosial. Man kan møte opp på området og støtte grasrotlaget å få grasrocoins for dette. Man kan også gå til jobbe eller trimme en tur og få grasrotcoins for det.

Som bruker kan man støtte grasrotlag og evetuelle undrgrupper av grasrotlagene. Når man tjener opp grasrotcoins kan man velge hvem av grasrotlagene/undergruppene man støtter og som skal få grasrotcoinsene i tillegg til brukeren.

Vi har brukt PHP, Javascript og CSS til å utvikle hovedfunksjonalitet i oppgave. Vi har også benyttet oss av diverse bibliotek og APIer som Ajax og jQuery, og HTMLs Geolocation API og Google Maps API. Vår applikasjon tilbyr også innlogging via Facebook.

ABSTRACT

Title:	NTwear15	Date : 15. May 2015
Participants:	Lars Bendik Dølvik John Christian G. Fjeld	
Supervisor:	Ivar Farup	
Employer:	Jørn Berg Nordlund, Norsk Tipping	
Keywords (3-5)	PHP, Javascript, SQL, Application, Database	

Number of pages: 81(114)	Number of appendix:10	Availability (open/confidential): Open

Short description of the bachelor thesis:

We have made a webapplication for Norsk Tipping. This applications purpose is to help Norsk Tipping in making the members of “grasrolag” more social and active.

By using our application users can acquire grasrotcoins when beeing active and social. Users can meet up and support their “grasrotlag” and acquire grasrotcoins by doing so. Users can also acquire grasrotcoins by beeing active, for example by walking or running.

Users can choose what “grasrotlag” to support. So, while acquiring grasrotcoins you can decide which of the “grasrotlag” you want to support and share the grasrotcoins with.

This application is written i PHP, Javascript and CSS. While developing this application we have use different liberies and APIs like Ajax and jQuery, and HTMLs Geolocation API and Google Maps API. Our application also offers a Facebook-login.

Forord

Grasrotbingo er i samarbeid med Norsk Tipping utviklet av 2 karer som studerer Bachelor i ingeniørfag – data ved Høgskolen i Gjøvik. Vi syns oppgaven har vært veldig spennende og utfordrende. Vi har fått innsikt i hvordan utvikling foregår ut i den store verden og fått bryne oss på mange reelle jobbsituasjoner.

Det har vært en veldig lærerik prosess. Vi har tatt med oss veldig mange erfaringer og mye kunnskap i sekken. Vi har vokst som utviklere i løpet av prosessen og ting som før virket skummelt er ikke lengre skummelt, men spennende utfordringer.

Vi har under denne oppgaven hatt Ivar Farup som vår veileder. Vi vil gjerne takke for samarbeidet og fine møter på hans kontor. Vi har fått mange gode tips og råd av Ivar i løpet av periode.

Vi vil også takke Jørn Berg Nordlund og Norsk Tipping for samarbeidet og tilliten. Jørn har med sin kunnskap kommet med mange ideer underveis og gitt oss en spennende bacheloroppgave. Vi vil i tillegg takke Håvard Kindem for all ekspertise og hjelp han har bistått med under utviklingen av applikasjonen.

Tilslutt vil vi takke Høgskolen i Gjøvik for tre flotte år og for å ha gitt oss den faglige kompetansen vi har den dag, idag.

15.Mai 2015

Lars Bendik Dølvik

John Christian G. Fjeld

Innholdsfortegnelse

Antall ord i oppgaven: 18875

Figurer

Kodeeksempel

1 Innledning

1.1 Introduksjon

Vi har laget en applikasjon i samarbeid med Norsk Tipping. Vi søkte på dette oppdraget da vi syns det virket som en spennende utfordring og vi ønsket å jobbe for en reel oppdragsgiver.

1.1.1 Problemområde

I idrettslag og frivillige foreninger(grasrotlag), erfarer vi at man setter stor pris på engasjement og bidrag. Da slike foreninger er ofte avhengig av sånne typer bidrag for å kunne fungere. La oss ta ett idrettslag som eksempel. Enten du selv er medlem i ett idrettslag, har barn som er medlem i idrettslaget eller du bare ønsker å støtte ditt lokale idrettslag så kan du bidra på forskjellige måter. Ofte er det snakk om bidrag i form av dugnader og salg av diverse saker. Idrettslagene arrangerer dugnader der man frivillig kan delta i og/eller sender med medlemmer lodd eller kakebokser, som må selges for å tjene inn penger på vegne av idrettslaget. Dette funker til en viss grad, men vi er alle kjent med fotball-lodd og kakebokser som blir stående og dugnader som ikke passer med jobb også videre. Tilslutt ender det med at man må gi fra egen lomme til foreningen, noe som selvfølgelig ikke er intensjonen.

Figur 1: Grasrotandelen - Norsk Tipping(1)

Norsk Tipping ønsker å hjelpe idrettslag og foreninger og lanserte i 2009 Grasrotandelen. Det Grasrotandelen gikk ut på var at spilleren hos Norsk Tipping selv kunne velge et idrettslag eller foreningen som ville få et beløp som tilsvarte 5% av spillerens innsats.[1]

Norsk Tipping ønsker nå å utforske muligheten

[1] https://www.norsk-tipping.no/grasrotandelen - Norsk Tippings - Grasrot Andelen

for å ta dette et steg videre. Rundt i den ganske land finnes det mange ildsjeler, som utgjør en stor forskjell og i visse tilfeller er grunnen til at hjulene går rundt. Ildsjeler er utrolig viktige, men det er ønskelig med en generelt større interesse og engasjement. Norsk Tippings ønske er dermed å øke engasjement og det sosiale innad, samtidig som de ønsker å forenkle inntektsformen som dugnader og lodd-salg gir i dag. I denne forbindelse har vi fått i oppdrag av Norsk Tipping å utvikle en applikasjon med navnet "Grasrotbingo".

1.1.2 Problemstilling

Hvordan engasjere, sosialisere og aktivisere grasrotmedlemmer ved bruk av en applikasjon?

1.1.3 Avgrensning og oppgavedefinisjon

I utviklingen av Grasrotbingo har vi laget en web-applikasjon som skal hjelpe til med å innfri og løse problemstillingen og ønskene(se avsnitt 2.3, Produktkø) Norsk Tipping har kommet med. Norsk Tipping har i forbindelse med vår bacheloroppgave kommet opp med en ny og spennende, digital valuta de har valgt å kalle for "grasrotcoins" og det er rundt denne valutaen applikasjonen er basert. Ved bruk av applikasjonen kan du opparbeide deg grasrotcoins og bruke dem til å potensielt kunne delta i konkurranser, innad i et grasrotlag. På bakgrunn av dette fikk vi i oppgave å lage en applikasjon som skal løse problemstillingen ved hjelp av grasrotcoins.

I denne oppgaven vil vi ikke ta for oss det økonomisk rundt å utvikle en applikasjon eller krav angående ytelse.

Dersom grasrotlaget ønsker å opprette undergrupper, som aldersbestemte trinn i idrettslaget eller diverse grupper i foreningen, vil en ansvarlig bruker fra grasrotlaget få innloggingsinformasjon til "Org Pålogging". "Org Pålogging" er en selvstendig innlogging for administratorbrukeren til organisasjonen (grasrotlaget). Inne på "Org-siden" vil grasrotlaget få muligheten til å legge inn de undergruppene de ønsker. På "Org-siden" vil også administratoren kunne endre på navnet på undergrupper, samt slette undergrupper om ønskelig.

På «Min Profil» vil du få mulighet til å velge ett eller flere grasrotlag du ønsker å støtte/delta for. Du vil også kunne velge om du vil støtte/delta for en spesifikk undergruppe i grasrotlaget, dersom administrator for grasrotlaget har lagt til undergrupper via "Org-

siden". Du står fritt til å hele tiden slette eller legge til ny grasrotlag/undergrupper. På "Profil-siden" kan du også endre kontakt- og påloggingsinformasjon

Først når man har laget bruker og valgt hvilke grasrotlag, eventuelt undergrupper, man ønsker å støtte/delta i vil man kunne begynne å opparbeide seg grasrotcoins.

Grasrotcoins kan du opparbeide deg ved å være aktiv. Vi har laget en "sporings"-funksjon som heter "Gå til jobben", som lar deg tjene deg opp grasrotcoins etter hvor langt du har gått, løpt eller syklet. Funksjonen bruker snittfart til å anta om bruker har gått, løpt eller syklet og sporer lengden ved bruk av *GPS*en, eventuelt med *telefontårn triangulering*. Er du ute og trimmer eller går til jobben og glemmer å starte turen, går det også ann å etter-registrere en tur, dette ved bruk av "Etter-registrering"-funksjonen.

Når man stiller opp og er engasjert innad i en grasrotlag burde også dette gi grasrotcoins. Derfor er det laget en funksjon som heter "Støtt Grasrotlag". Denne lar deg tjene opp grasrotcoins etter hvor lenge du befinner deg på området. Det er satt to krav som må oppfylles for at brukeren skal få grasrotcoinsene brukeren har tjent opp. Det første kravet er at brukeren må være på området i 20 minutter eller mer. Det andre kravet er at brukeren må befinne seg i en radius på 250 meter fra startposisjonen når brukeren velger å forlate området. Hvor startposisjonen er posisjonen brukerens enhet har i det "Støtt Grasrotlag"-funksjonen startes.

Når du tjener opp grasrotcoins vil du som bruker få lagret alle grasrotcoins på din bruker, men du tjener også opp grasrotcoins til et av grasrotlagene/undergruppen du har valgt å være med i. Du kan kun tjene opp grasrotcoins til ett grasrotlag/undergruppe per økt.

Alle grasrotcoins både på bruker og grasrotlag/undergruppe lagres i databasen som gjør det mulig å sjekke hvilken bruker, hvilke grasrotlag og hvilke undergrupper i et grasrotlag som har tjent inn flest grasrotcoins om ønskelig.

Det er også utviklet et spill, bingo. Med bingoen vil grasrotlag få muligheten til å spille og konkurrere innad i grasrotlaget. For eksempel kan ett grasrotlag arrangere bingo en gang i måneden for å øke den sosiale omgangen innad. I tillegg kan grasrotlaget dele ut premier til brukere og undergrupper som har klart å tjene opp flest grasrotcoins, da denne dataen er tilgjengelig.

1.2 Målgrupper

Denne rapporten og applikasjonen har flere enn en målgruppe.

Målgruppen for rapporten er først og fremst sensor og Norsk Tipping som oppdragsgiver. Det vil også være mulig for studenter/andre med tilnærmet lik fagbakgrunn som oss å lese rapporten og dra nytte av våre erfaringer. Rapporten kan være nyttig for Norsk Tipping, dersom de ønsker å videreutvikle applikasjonen. Vi håper også at rapporten også vil være et hjelpemiddel for Norsk Tipping dersom applikasjonen skal introduseres for grasrotlagene.

Applikasjonen er laget for Norsk Tipping, deres grasrotlag og lagenes medlemmer.

1.3 Prosjektmål

Prosjektet har ulike typer mål, dermed har vi valgt å dele prosjektmålene opp i underkategorier for å kunne forklare de forskjellige prosjektmålene på en konkret og oversiktlig måte. Resultatmål beskriver hva oppdragsgiver vil at applikasjonen skal levere. Effektmål er hva oppdragsgiver ønsker å oppnå med applikasjonen. Læringsmål sier noe om hva vi i prosjektgruppen har som mål å lære.

1.3.1 Resultatmål

Resultatmålet med oppdraget var å utvikle en prototype ut ifra ønsker og krav gitt av Norsk Tipping. Applikasjonen skulle gi muligheten til å tjene og lagre grasrotcoins på en aktiv måte til ett eller flere grasrotlag/undergrupper. I tillegg vil det være en spillfunksjon som skal kunne brukes i en sosial setting. Applikasjonen skal kunne fungere på Iphone og helst på flere plattformer.

1.3.2 Effektmål

Med denne applikasjonen er målet å skape ett konkurransemiljø innad i grasrotlag som vil resultere i et mer aktivt og engasjert miljø. Ved å kunne samle inn grasrotcoins til en spesifikk undergruppe, for eksempel en fotballklubb, vil klubbmedlemmer og foresatte kunne konkurrere om hvilket lag som har samlet flest grasrotcoins. Med bingoen er målet å skape en sosial spill funksjon, som for eksempel kan kjøres i pausen på en fotballkamp. Bingoen vil da bidra som pauseunderholdig, samt muligens øke oppmøte på kampen.

1.3.3 Læringsmål

Ved å jobbe med dette prosjektet har vi i prosjektgruppen tilegnet oss en hel rekke nye erfaringer. Ved å jobbe i stort prosjekt over lengre tid har vi lært viktigheten ved planlegging, strukturering og gjennomføring av både arbeid og kode. Vi har også lært hvordan det er å jobbe under en reell oppdragsgiver i en reell jobbsituasjon. Ved å være med å utvikle i en reell systemutviklingsprosess har vi fått lært hvordan utviklingsmodellen Scrum (se avnsitt 1.6, Utviklingsmodell) fungerer i arbeidslivet. I prosessen har vi brukt og lært nye språk, nye *API*er og nye *biblioteker*. Vi har også fått nye erfaringer ved å bruke nye kodeverktøy og hjelpeverktøy i prosessen, se avsnitt 5.1, Utviklingsverktøy. Sammen med alt dette, har vi endelig få sett hvordan ting vi har lært igjennom studiet passer inn i arbeidslivet.

1.4 Bakgrunn og faglig kompetanse

Gruppen besto i utgangspunktet av tre studenter hvor to studerer Dataingeniør, Lars og John, og en Programvareutvikling, Petter, ved Høgskolen I Gjøvik. Dessverre falt Petter fra prosjektgruppen underveis. Dette fordi Petter var fraværende, ikke deltok og ikke viste initiativ. De resterende gruppemedlemmene var dermed nødt til å tre frem i henhold til gruppeavtalen, (se vedlegg 9.10, Forprosjekt). Til slutt ble veileder kontaktet, og i forbindelse med dette valgte Petter selv å trekke seg fra prosjektet. Dermed ble prosjektet fullført med kun to medlemmer, Lars og John. Får å se spesifikt hva Petter har bidratt med i prosjektet se vedlegg 9.3, JIRA.

I løpet av Dataingeniørstudiet har vi fått god kjennskap til programmering og objekt-orientert tankegang, og det har vært mye fokus på programspråket C++. Vi har også tatt andre relevante fag som Datamodellering & databasesystemer og Systemutvikling. I tillegg til dette har vi begge hatt valgfag som er relevante til oppgaven. Vi har begge hatt valgfag innen web-programmering, hvor vi lærte det elementære innenfor emnet. Lars har også i tillegg tatt ett fag som omhandler mobilutvikling på android plattformen ved bruk av *Android Studio*, som kom godt med da han har vært borti kartverket Google Maps API, fra før. Mens John hadde hatt enkel innføring i PHP og CSS, da han var på utveksling.

Ingen av oss har jobbet med større prosjekter før, og det var mye vi måtte sette oss inn i. Ingen av oss har i stor grad vært innom Javascript, Jquery eller Ajax, noe vi måtte lære oss da dette har blitt anvendt flere steder i applikasjonen. Vi har heller ikke hatt noen andre store prosjekter som har gått over en slik periode og vært såpass tidskrevende. Det nærmeste vi har vært et slikt prosjekt er ett prosjekt vi hadde i systemutvikling, som gikk over ett par måneder hvor vi hadde rundt 15 arbeidstimer i uken, og ett programmeringsprosjekt i Objekt-orientert Programmering. Hvor begge disse var gruppeprosjekt på tilnærmet like store grupper.

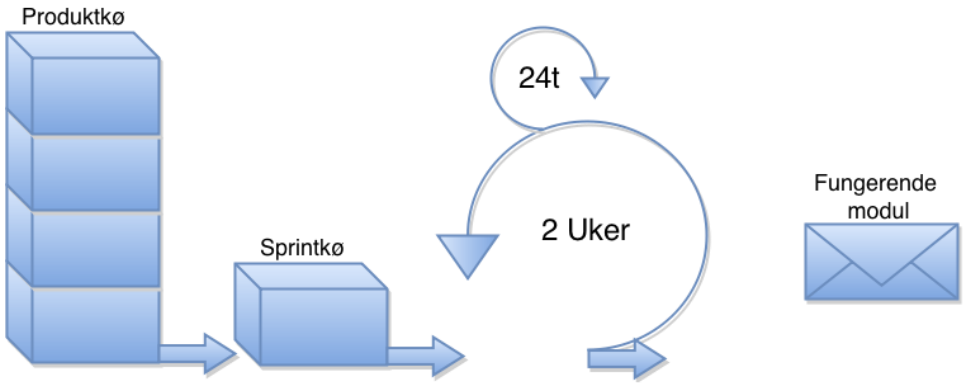
1.5 Arbeidsmetode

Det hele startet med at vi jobbet som gruppe over nett, fordelte arbeidsoppgaver oss imellom og var svært lite på skolen. Dette viste seg å fungere dårlig da vi ikke fikk skikkelige diskusjoner med hverandre og avklart ting som var uklart. Så vi bestemte oss for å møte hver ukedag på skolen i tillegg til et endel helger og jobbet som en gruppe. Hvor dette viste seg å være svært mye mer effektivt da vi kunne hjelpe hverandre og dele kunnskapen vår, slik at den andre slapp og finne ut dette på egenhånd. Det utviklet seg til å bli gjøremål som skulle være ferdig før vi dro hjem for dagen, som resulterte i noen lange dager, men vi mener dette var en god løsning.

Vi har i denne utviklingsprosessen brukt Scrum, da dette var ett sterkt ønske fra Norsk Tipping. Dette er noe de bruker selv og har god kontroll på. Scrum var også noe vi hadde brukt før i systemutviklingsprosjektet tidligere, så dette passet veldig bra.

1.6 Utviklingsmodell

Som hjelpeverktøy til Scrum, som er en smidig utviklingsmodell, valgte vi å benytte oss av prosjektplanleggingsverktøyet JIRA og fikk raskt satt opp dette med hjelp fra Høgskolen. Vi satte opp deler av produktkøen sammen med oppdragsgiver så fort oppgaven var avklart og dette lot seg gjøre. Vi fordelte oppgavene i prosjektgruppen etter hva vi var gode på og jobbet systematisk med hvilke oppgaver vi la inn i sprintkøen.



Figur 2 - Scrummodellen

Sprintene våre gikk over to uker, dette var ett ønske fra Norsk Tipping, da det var dette de brukte selv. Vi samlet oss hver morgen, så ofte det lot seg gjøre, for å holde *daily scrum meeting*. Vi forsøkte også å ha sprintmøter med arbeidsgiver annenhver uke for å vise fremgangen i prosjektet. På disse møtene fikk vi kommentarer og tilbakemeldinger på det vi hadde gjort, samt ønsker om endringer av funksjonalitet og ønsker om ny funksjonalitet. Alle tilbakemeldingene som ble gitt, fikk vi tatt notater på og dersom dette var hensiktsmessig ble de implementert i løsningen. Møtene ble gjort både i person hos Norsk Tipping på Hamar, men også over nett hvor vi brukte kommunikasjonsprogrammet Lync, se punkt 5.1.9, Skype og Microsoft Lync.

Siden vi benyttet oss av JIRA i utviklingsprosessen, så hadde vi hele tiden full oversikt over hvordan prosessen gikk fremover. Vi hadde også gitt arbeidsgiver tilgang til JIRA, så han/de hele tiden kunne følge med på prosessen og komme med tilbakemeldinger på fremdriften.

Ut over dette så hadde vi også møte med veileder annenhver onsdag kl 10.00 på veileders kontor. Her viste vi frem hva vi hadde gjort, hvordan vi lå an i forhold til prosessen og hvordan vi har planlagt å jobbe fremover. Vi fikk mye mer faglige tilbakemeldinger av veileder som viste seg å være svært bra i forhold til alle ideene som vi fikk av oppdragsgiveren.

For å hjelpe oss med å holde orden hvilken versjonen av applikasjonen som er det nyeste har vi i utviklingsprosessen benyttet oss av versjonskontrollen Git. Som *frontend* til Git har vi brukt verktøyet *SourceTree*, for å enkelt håndtere bruken av Git. (se punkt 5.1, utviklingsverktøy)

1.7 Roller

1.7.1 Arbeidsgiver:

Bedrift:

Norsk Tipping AS er ett aksjeselskap som er underlagt Kulturdepartementet.
Norsk Tipping er Norges eneste aktør innen drift av pengespill og har en viktig posisjon i å forhindre negativ spilladferd i samfunnet. Dette setter Norsk Tipping i en unik situasjon hvor størst mulig overskudd ikke er dets viktigste formål, men heller å føre pengespill i Norge i en forsvarlig sosialpolitisk retning, og sørge for at overskuddet fra pengespill skal komme resten av samfunnet til gode ved sponsing av lokale institusjoner som idrettslag og kulturskoler.(1)

Norsk tipping er under stadig utvikling og hele tiden på utkikk etter andre måter og ekspandere seg på. I 2014 kom de med *Instaspill* og lanserte nylig *Nabolaget* (13. Februar, 2015)

Kontaktperson:

Jørn Berg Nordlund har vært vår kontaktperson hos Norsk Tipping. Han er ansatt i Norsk Tipping under tittelen Virksomhetsleder, Arkitektur. Det var Jørn som vi viste frem progresjonen i prosjektet vårt til og som kom med innspill om ny funksjonalitet. Det var også Jørn som formulerte oppgaven som vi valgte og det var han som tildelte oss den etter søknaden vi sendte inn. Tidligere har Jørn jobbet for både *IBM* og Steria, han har dermed god kjennskap til utviklingsprosessen, og han har ett godt innblikk på hvordan ting skal gjøres på en god og effektiv måte.

Håvard Kindem var vår andre kontaktpersonen hos Norsk Tipping. Håvard jobber på Norsk Tipping som konseptutvikler og gjort dette siden April 2014. Håvard har også en bachelor fra Høgskolen i Gjøvik innenfor spillprogrammering i 2010. Så han har en god forståelse for hva vi har gått igjennom på skolen og vet hva vi kan og skal kunne. Det er Håvard vi har henvendt oss til da det har vært ett faglig spørsmål angående prosjektet. Han hjalp oss med å få tilgang til og forstå API'ene og funksjonene Norsk Tipping allerede bruker, så vi kunne ta de i bruk dersom vi ønsket det.

Veileder:

Ivar Farup, emneansvarlig for Dataingeniør ved Høgskolen I Gjøvik har vært til god hjelp under prosjektet. Ivar Farup har vært ansatt ved denne skolen siden 2000, og er nå professor i informatikk som han har vært siden 2012. Med hans erfaring har vært til stor hjelp for oss under prosjektet.(2)

1.8 Organisering av rapporten

1.8.1 Oppsett av rapporten

Vår rapport er delt opp i 9 kapitler. Alle kapitler i rapporten er videre delt opp i underkapitler som igjen inneholder underoverskrifter. Alle disse forskjellige nivåene er nummeret for å gjøre rapporten oversiktlig. I innholdsfortegnelsen står tittel på kapitler og underkapitler med sitt nummer og sidetall.

Vi har igjennom rapporten fulgt kravene funnet i "Retningslinjer for mastergradsoppgaver og større studentoppgaver på bachelornivå ved Høgskolen i Gjøvik".(3)

Denne rapporten inneholder faguttrykk og ord på både norsk og engelsk som kan være vanskelig å forstå dersom man ikke har tilnærmet lik faglig kompetanse som prosjektgruppen, og dermed er ikke alle ord definert. De andre faguttrykkene som vi ser på som uklare og som kan være ukjent for noen, finnes det en liste, Definisjoner, se Vedlegg 9.1, Definisjoner, som beskriver vanskelig ord i rapporten. Første gangen et ord som finnes i Definisjoner forekommer, vil ordet stå i kursiv.

1.8.2 Kapitteloppsummering

Kapittel 1: Innledning

I dette kapittelet kan man lese om problemområdet og grunnlaget for prosjektet. Avgrensninger og oppgavedefinisjon. Man kan også lese litt om hvordan vi har gått frem for å løse problemstillingene og målet med prosjektet.

Kapittel 2: Kravspesifikasjon

Her beskrives systemets brukere. Applikasjonens funksjoner illustreres ved hjelp av et Use Case diagram og funksjonenes krav spesifiseres.

Kapittel 3: Design

Under Design blir det skrevet om mål ved designet og bakgrunnen for vårt design. I tillegg vises og beskrives applikasjonens brukergrensesnitt.

Kapittel 4: Arkitektur

Her beskrives og illustreres applikasjonens logiske oppsett. Ved bruk av figurer og tekst får man ett innsyns i filstrukturen, databasedesignet og kommunikasjonen mellom klient og server.

Kapittel 5: Implementasjon

Her forklares hvordan vi har kodet applikasjonens funksjoner. Det fortelles også om språk og verktøy som har blitt brukt i prosessen og tekniske memoer.

Kapittel 6: Testing og kvalitetssikring

Kapittelet omhandler testing på forskjellige måter. Det fortelles om Whitebox og Blackbox testing og eksempler på hvordan vi har utført disse. Det er også lagt ved brukertester, som er utført av potensielle brukere for å teste applikasjonens brukergrensesnitt og funksjoner.

Kapittel 7: Avslutning

Her evalueres arbeidet. Det fortelles om avgjørelse og diskusjoner som har blitt gjort underveis, kritikk til oppgaven og mulighetene ved videre arbeid på applikasjonen. Tilslutt kan det leses om hvordan gruppen har fungert og en oppsummering av prosjektet.

Kapittel 8: Litteraturliste

Her finner man alle referanser vi har benyttet og markert i rapporten.

Kapittel 9: Vedlegg

Under dette kapittelet ligger alle vedlegg som er med i rapporten.

Definisjoner, Logg, Statusrapport, JIRA, Brukertester, Regler for bingo brett, Introduksjon i Bluemix, README.txt, Facebook, Kontrakten og Forrapport.

.

2 Kravspesifikasjon

2.1 Brukerbeskrivelser

2.1.1 Omgivelser

Dersom en bruker ønsker å ta i bruk applikasjonen må brukeren først laste ned applikasjonen eller åpne nettsiden i nettleseren. Deretter logge inn med gyldig brukernavn og passord, eller registrere ved førstegangsinnlogging. Bruker kan velge å logge inn via Facebook, om bruker heller ønsker dette. Applikasjonen må brukes på enheter som har tilgang til internett, fordi applikasjonen krever nett-tilgang for å kunne utføre tilnærmet alle dens funksjoner. Enheter med GPS vil ha mer presis distansemålinger og kart produsert via Google Maps API. Brukeren trenger ikke å laste ned noe utover applikasjonen.

2.1.2 Systemets brukere

Applikasjonen vil ha enn rekke forskjellige type aktører.

Bruker:

Det er brukeren som vil benytte applikasjonen. En bruker er ideelt sett en person som ønsker å engasjere seg innad i et Grasrotlag, og vil gjør dette igjennom å laste ned applikasjonen å bruke dens funksjoner.

Administrator:

Applikasjonen har en såkalt «superbruker», administrator, som full tilgang til alt på applikasjonen. Administratoren kan gå direkte inn i applikasjonens database og rette opp eventuelle feil eller lignende dersom det skulle være nødvendig.

Org Administrator:

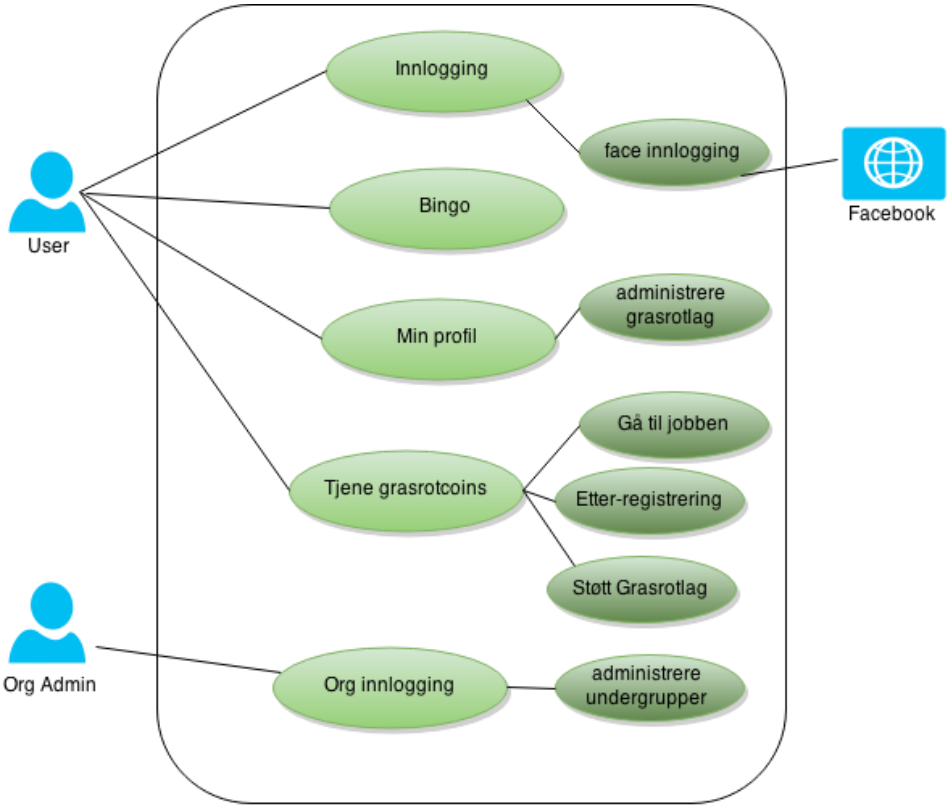
Alle Grasrotlag har mulighet til å legge til, endre og slette undergrupper innad i Grasrotlaget dersom de ønsker det. Dette er kun Ord Administrator som har rettigheter til å gjøre gjennom å logge seg inn via Org Innlogging. Det vil kun være en Ord Administratorbruker per Grasrotlag.

2.2 Systemets egenskaper

Scrum har ingen spesifikk mal når det kommer til å lage kravspesifikasjon. Derfor har vi valgt å benytte oss av use case for å fortelle krav satt til de forskjellige funksjonene. Dette er illustrert gjennom, use case diagram, high-level use case og detaljert use case.

2.2.1 Use case diagram

Dette er et use case diagram som viser hvordan funksjoner i prosjektet arbeider sammen.



Figur 3 -Use case diagram, laget i draw.io(4)

Facebook:

Fungerer som en ekstern aktør for påloggingen. Tilknytningen til Facebook gjør det mulig for brukeren og verifisere med Facebook, slik at man slipper å registrere en ny bruker om du allerede har en eksisterende Facebook-profil.

2.2.2 High-level use case beskrivelser

UseCase	**Min Profil**
Aktør	Bruker
Mål	Administrere informasjon til brukerens profil
Beskrivelse	På profilsiden så vil brukeren kunne se all informasjon om seg selv. Og brukeren vil ha tilgang til å endre personalia og passordet om dette er ønskelig. Brukernavnet vil være låst og kan kun endres om brukeren selv kontakter en administrator.

UseCase	**Administrere grasrotlag (Extend av Min Profil)**
Aktør	Bruker
Mål	Administrere organisasjonene brukeren vil støtte
Beskrivelse	På profilsiden vil det også være mulighet for å kunne legge til Grasrotlag. Du vil få mulighet til å legge til alle du ønsker å støtte. Du kan også legge til undergrupper av Grasrotlaget lagene dine, og fjerne de du ikke lengre vil støtte.

UseCase	**Gå til jobben (Tjene grasrotcoins)**
Aktør	Bruker
Mål	Aktivisere befolkningen
Beskrivelse	Det vil være en funksjon hos brukeren hvor han har mulighet til å tjene opp Grasrotcoins ved å være aktiv. Det vil bli tjent opp Grasrotcoins til dine valgte Grasrotlag. Det blir regnet ut hvor mange Grasrotcoins du skal få ut ifra hvor lang distanse du har tilbakelagt og fremgangsmetode, om du gikk, jogget eller syklet. Hvor du vil tjene mest ved å jogge og minst ved å sykle.

UseCase	**Etter registrering(Tjene grasrotcoins)**
Aktør	Bruker
Mål	Etter-registrere "Gå til jobben"-tur

Beskrivelse	Skulle du glemme å starte distansemålingen når du er ut å trimmer/på tur betyr ikke det nødvendigvis at du ikke får Grasrotcoins for innsatsen. Applikasjonen har en funksjon som heter Etter-registrering som lar deg gå inn å registrere turen du akkurat har vært på. Og vi har plukket ut tre distanser sammen med NT som brukeren kan velge mellom. Her kan du velge hvor lang turen var, hvordan du trimmet og hvem av dine registrerte Grasrotlag eller undergruppe du ønsker å gi Grasrotcoinsene til.

UseCase	**Støtt Grasrotlag (Tjene grasrotcoins)**
Aktør	Bruker
Mål	Øke engasjement rundt grasrotlagene
Beskrivelse	Det vil være en funksjon hos brukeren hvor brukeren har mulighet til å registrere at personen har vært tilstede, og støttet Grasrotlaget. Det skal være en minimum tilstedeværelse på 20 minutter, og dette blir sporet. Brukeren må også befinne seg i radius på 250 meter fra utgangspunktet i det brukeren avslutter økten dersom den skal være tellende. Ut ifra dette vil brukeren tjene Grasrotcoins etter hvor lenge brukeren befant seg på området.

UseCase	**Innlogging**
Aktør	Bruker
Mål	Autorisere bruker pålogging
Beskrivelse	Før brukeren får tilgang til noe som helst på våres applikasjon så må brukeren være innlogget. Dataen er lagret i databasen og blir dermed sjekket opp mot denne om brukeren eksisterer og informasjonen stemmer. Dersom brukeren ikke har en eksisterende profil eller ønsker å logge inn via Facebook, må brukeren registrere seg ved bruk av registreringssiden

	som er tilgjengelig gjennom "Registrer deg"-knappen på innloggingssiden.

UseCase	**Facebook innlogging**
Aktør	Bruker
Mål	Autorisere pålogging via Facebook
Beskrivelse	Skal være en alternativ måte å logge inn på. Det er en knapp ved innloggingssiden hvor brukeren får muligheten til å logge seg inn ved hjelp av Facebook. Tilknytningen til Facebook skal gjøre det mulig for brukeren og verifisere med Facebook, slik at man slipper å registrere en ny bruker om du allerede har en eksisterende Facebook profil.

UseCase	**Org innlogging**
Aktør	Org admin
Mål	Autorisere Org administrators pålogging
Beskrivelse	Dersom Grasrotlaget ønkser å ha undergrupper må dette gjøres på Org siden. Får å komme til Org siden må en ansvarlig bruker for Grasrotlaget, Org admin, logge inn via Org innlogging med innloggingsinformasjon levert av Norsk Tipping.

UseCase	**Administrere undergrupper (Extend av org innlogging)**
Aktør	Org admin
Mål	Administrere organisasjonen med undergrupper
Beskrivelse	Dette vil være en side for organisasjonene, hvor de får tilgang til organisasjonen sin, ved en organisasjon innlogging. På organisasjonssiden vil organisasjonen ha tilgang til å legge til undergrupper til sin organisasjon. Det vil også være mulighet for å endre de eksisterende og fjerne dem.

UseCase	Bingo
Aktør	Bruker
Mål	Spill som kan brukes i forbindelse med Grasrotcoins
Beskrivelse	Spillet skal være en mulighet for Grasrotlag å arrangere spill i forskjellige situasjoner. Potensielt skal brukeren kunne bruke Grasrotcoins til delta i bingo, eventuelt kjøpe seg inn.

2.2.3 Detaljert use case analyse

På den detaljerte analysen så har vi valgt å bare gå ekstra inn på to av use casene. De vi valgte ut var Facebook pålogging, og administrere undergrupper. Grunnen til at vi valgte å ta ut disse er fordi vi ser på dette som to omfattende funksjoner i vår applikasjon.

UseCase	Facebook innlogging
Aktør	Bruker
Mål	Autorisere pålogging via Facebook
Beskrivelse	Skal være en alternativ måte å logge inn på. Det skal være en knapp ved innloggingssiden hvor brukeren får muligheten til å logge seg inn ved hjelp av Facebook. Tilknytningen til Facebook skal gjøre det mulig for brukeren og verifisere med Facebook, slik at man slipper å registrere en ny bruker om du allerede har en eksisterende Facebook profil.
Betingelser	1. Må være internett tilgang og kunne koble til serveren 2. Må ha Facebook bruker 3. Godta at vår applikasjon får tilgang til profil informasjonen på Facebook 4. Ingen nettverks brudd.
Detaljert hendelsesforløp	1. Går til innloggingssiden til applikasjonen 2. Trykker på Facebook-innloggingsknappen 3. Case 1: Førstegangs pålogging a. Facebook åpner seg

	b. Trykk godta at Facebook får tilgang til brukerens profil informasjon c. Mailen finnes ikke i DB d. Skriver informasjon til DB Case 2: Har vanlig bruker, trykker Facebook-pålogging a. Facebook åpner seg b. Trykk "Godta" når Facebook ber om tilgang til brukerens profil informasjon c. Sjekker mail på Facebookbrukers profil d. Ser om samme mail finnes i DB på en bruker e. Legger til Facebook-ID til DB på brukeren Case 3: Andre gangs eller senere trykker knappen a. Kobler til DB b. Sjekker om informasjonen stemmer c. Treff på mail og facebook-ID i DB 4. Setter Sessions variable 5. Sender bruker til meny siden.
Feil situasjoner	1. Godtar ikke at applikasjonen får tilgang til Facebookinformasjonen til brukeren. 2. Får ikke koblet til DB eller Facebook
Alternativer	1. Hvis brukeren ikke vil at applikasjonen skal ha tilgang til Facebook-profilen, så må det registreres en bruker med ”Registrer”-knappen.

UseCase	**Administrere undergrupper**
Aktør	Org admin
Mål	Administrere organisasjonen med undergrupper
Beskrivelse	Dette vil være en side for organisasjonene, hvor de får tilgang til organisasjonen sin, ved en organisasjon innlogging. På organisasjonssiden vil organisasjonen ha tilgang til å legge til undergrupper til sin organisasjon. Det vil også være mulighet for å

	endre de eksisterende og fjerne dem.
Betingelser	1. Organisasjonen må ha kontaktet en administrator og fått tildelt påloggingsinformasjon 2. Koblet til internett og databasen 3. Organisasjonen er pålogget
Detaljert hendelsesforløp	1. Case 1: Legg til undergruppe a. Trykk på "add"-knappen b. Fyll inn ønsket navn på ny undergruppe c. Trykk "Lagre"-knappen d. Sjekker om noe står i feltet e. Skriver ny undergruppe til org på DB 2. Case 2: Endre undergruppe a. Trykk på "endre"-knappen b. Velg ifra dropdown meny hvilket undergruppe du vil endre navnet på c. Eksisterende navn vises i navn feltet d. Skriv over med det nye ønskede navnet e. Trykk "Lagre"-knappen f. Sjekker om undergruppe er valgt g. Sjekker at nytt navn er noe h. Skriver ny info til DB 3. Case 3: Fjern undergruppe a. Eksisterende ulag vises i listet b. Marker ved hjelp av avkrysning hvilken du vil slette. c. Trykk på "slett"-knappen d. Sjekker om noe er markert e. Fjerner markert underlag fra databasen
Feil situasjoner	1. Innloggings informasjonen stemmer ikke 2. Mulighet for duplikater av underlag i DB
Alternativer	1. Kontakt administrator for ny informasjon. 2. Sjekk at det ikke er duplikater i ditt eget ulag ved oversikten,

	på dine undergrupper.

2.3 Produktkø

Prosjektets produktkøen inneholder all funksjonalitet som det var planlagt at skulle implementeres i applikasjonen. Produktkøen inneholder både funksjonalitet som var ett krav og funksjonaliteten som var ønsket fra oppdragsgiver. For en fullstendig oversikt over hvordan vi fordelte produktkøen opp i sprintkøer, se Vedlegg 9.3, JIRA

NTWEAR-49	legge til bootstrap
NTWEAR-48	Send data til getstadioncoins.php ved "stopp"(Støtt Grasrotlag)
NTWEAR-47	Generere kart + pin + radius sirkel(Støtt Grasrotlag)
NTWEAR-46	Lag distanse måler, tidsjekk+stoppknapp(Støtt Grasrolag)
NTWEAR-45	organisere filsystem
NTWEAR-44	kode kommentering
NTWEAR-42	på profile add så velge flere ulag og skrive alle til db
NTWEAR-41	org side hover man kan lage ulag til kunn sin org
NTWEAR-40	lag admin table
NTWEAR-39	Fix duplicate of gOrg in DB on add
NTWEAR-38	add grasrotcoins til db
NTWEAR-37	start og stopp button
NTWEAR-36	Vise posisjon/rute på map
NTWEAR-35	Hente startlokasjon og tracke distanse tilbakelagt
NTWEAR-34	oppdatere profil
NTWEAR-30	Sette seg inn i phonegap
NTWEAR-29	Flytte DB
NTWEAR-28	Legg til Facebook innlogging og legge til i DB
NTWEAR-27	Rydde opp i DB og lage ny struktur
NTWEAR-26	Legge til NT stil på siden
NTWEAR-25	Lage profilside(legge til grasrotlag)
NTWEAR-24	En spiller kan registere fler grasrotmottakere
NTWEAR-22	Oppdatere bloggen
NTWEAR-21	Endre db-struktur
NTWEAR-20	Sette seg inn i node.red
NTWEAR-19	Importere grasrotlag til db
NTWEAR-18	JIRA-FIX
NTWEAR-17	connect til DB fra bluemix
NTWEAR-16	Compare som sjekker at markerte tall også står i bingo rekke, eller fler
NTWEAR-15	sette opp jquery og markere felt i bingoen, hvor de markerte legges i en array
NTWEAR-14	Relasjonsdiagram
NTWEAR-12	sette opp database
NTWEAR-11	registering
NTWEAR-10	Innlogging
NTWEAR-9	Må nå ha bingo på 2rader++(deles opp)?
NTWEAR-8	Bingomelding til alle
NTWEAR-7	Fikse sleep på tallgen(sprint 2)
NTWEAR-6	Send TallGen til client(sprint 2)
NTWEAR-5	Opprette websocket-forbindelse mellom client og server(sprint 2)
NTWEAR-4	Compare mellom markerte tall og brett, finne ut om korrekt markering.(sprint 2)
NTWEAR-3	compare mellom KORREKTEmarkerte tall og Trektetall(sprint 2)
NTWEAR-2	Motor som generer random tall, displayer og adder i array
NTWEAR-1	Lage brett som genererer tall på riktig måte.

Figur 4 - Prosjektets produktkø

2.4 Supplementær spesifikasjon

2.4.1 Hardware

For at applikasjonen skal fungere må brukerens enhet som har applikasjonen installert ha tilgang til internett. Applikasjonen bruker en sporings-funksjon som gir brukeren muligheten til å tjene opp grasrotcoins, for at denne sporingen skal bli så presis som mulig bør brukerens enhet ha GPS. Vår applikasjon er en web-applikasjon, men også en nettside som kan kjøres i en nettleser på for eksempel en PC. Man vil kunne spore PCen sin lokasjon selv om den ikke har GPS, dog vil sporingen bli mindre presis enn den ville vært med GPS.(5)

2.4.2 Software

For at brukeren skal kunne bruke sporings-funksjonen, som finnes både på distansemålingen og "Støtt Grasrotlag"-funksjonen, må enhetens nettleser støtte HTML5 sitt Geolocation API. Dersom brukerens enhet innfrir alle kriteriene må brukeren selv godta at han/hennes posisjon blir brukt til å utføre nødvendige kalkulasjoner. Denne forespørselen vil komme når brukeren går inn på en side som krever lokasjonsdata fra brukerens enhet. Dette for å ivaretatt personvern, slik at brukere som ikke ønsker å frigi sin lokasjon slipper dette, igjen, de får da ikke brukt delene av applikasjonen som krever brukerens lokasjon.

2.4.3 Dokumentasjon

Norsk tipping ønsket å sjekke ut potensialet i en ide de har, gjennom vår applikasjon. Vår applikasjon er derfor en prototype, altså ikke et ferdig produkt. Med tank på muligheten på at Norsk Tipping kanskje ønsker å videreutvikle applikasjonen er koden kommentert og vi har etter beste evne prøvd å kode pent og ryddig. Både i rapporten og i applikasjonen har vi vært nøye med systemets arkitektur, se kapittel 4.

2.4.4 Sikkerhet

Det ble også satt som ett krav at det skulle være sikkert å bruke applikasjonen. Det skulle være en sikkerhet ved dataoverføring mellom klient og server, og det skulle være en sikkerhet mellom brukerne. Blant annet ved innloggingsløsningen var det ønskelig at passord og annen sensitiv data til brukerne ble kryptert og ikke kunne misbrukes av administratorene av applikasjonen eller andre som får tilgang til databasen.

3 Design og Brukergrensesnitt

I dette kapittelet har vi beskrevet applikasjonens brukergrensesnitt. Vi har tatt for oss de fleste sidene, men valgt og ikke ta med alle da det er noen av sidene er relativt like.

3.1 Generelle mål ved design og brukergrensesnitt

Vår applikasjon har som mål å kunne bli brukt av alle medlemmer av et Grasrotlag og andre som ønsker å støtte Grasrotlaget. Dette betyr at applikasjonens målgruppe sannsynligvis er veldig spredt når det kommer til alder og datakyndighet. Som en konsekvens av dette er brukergrensesnittet og design utviklet slik at applikasjonen skal være lett å bruke. Designet er veldig simpelt og direkte. Det er enkelt å manøvrere seg mellom de ulike funksjonene, knappene er store og tydelig, teksten i applikasjonen er skrevet på en måte for å gjøre funksjoner lette å forstå. All tekst i vår applikasjon er skrevet på norsk, da vi antar at dette er morsmålet for majoriteten av målgruppen.

3.2 Bakgrunn for vårt design

Vi har brukt bootstrap sitt bibliotek til å hjelpe oss med å designe applikasjonen. Gjennom boostraps biblioteket fikk vi laget en simpel, men elegant layout. Resten av stylingen er gjort i en egen css-fil.

Under utviklingen av applikasjonen fikk vi et ønske fra Norsk Tipping. De ønsket at vi skulle bruke deres designguide dersom vi skulle benytte oss av en av deres logoer og elementer. Dette tok vi selvfølgelig stilling til siden vi, for det første ønsket å gjøre Norsk Tipping så tilfreds med produktet som mulig og for det andre, ville vi at applikasjonen skulle ligne mest mulig på et produkt levert av Norsk Tipping.

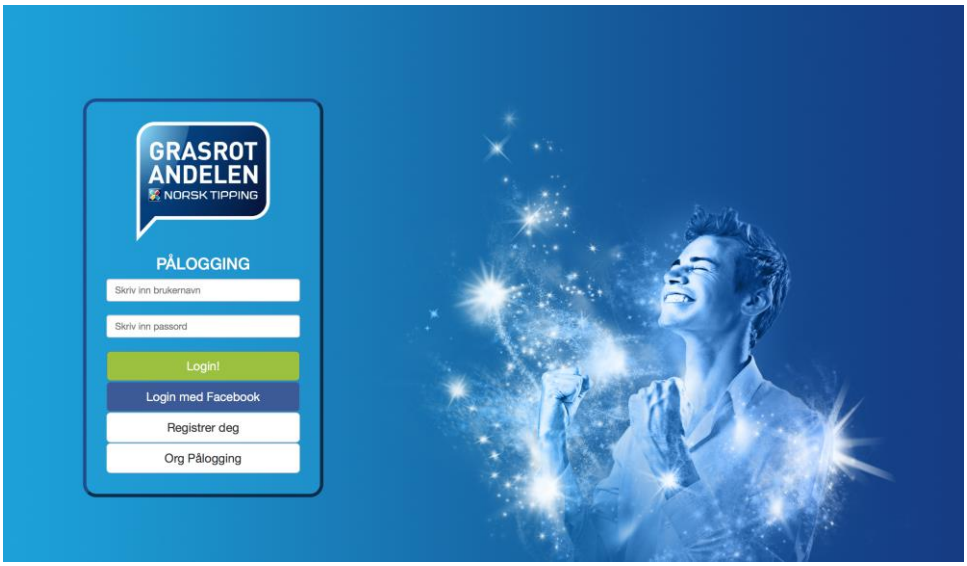
I Norsk Tippings designguide blir det tydelig presisert hvilke krav som må innfris dersom man ønsker å benytte seg av diverse elementer. På designguiden er det oppgitt hvilke kombinasjoner av farger, fonter, bakgrunner, logoer og størrelser man har lov til å ta i bruk, og på hvilken måte disse skal brukes om hverandre. Vi ønsket å bruke deres Grasrot Andelen-logo. Dette medførte at vi måtte følge visse designkrav som er en del av Grasrot Andelen-temaet. I vår applikasjon har vi derfor brukt Grasrot Andelens *fargepalett*, som

består av 2 forskjellige blåtoner, grasrot-grønn og hvit. Vi tok også i bruk andre designelementer fra guiden, blant annet fonter, og bakgrunnsbildet som vises dersom brukeren logger inn via PC.

3.3 Brukergrensesnitt

3.3.1 Pålogging

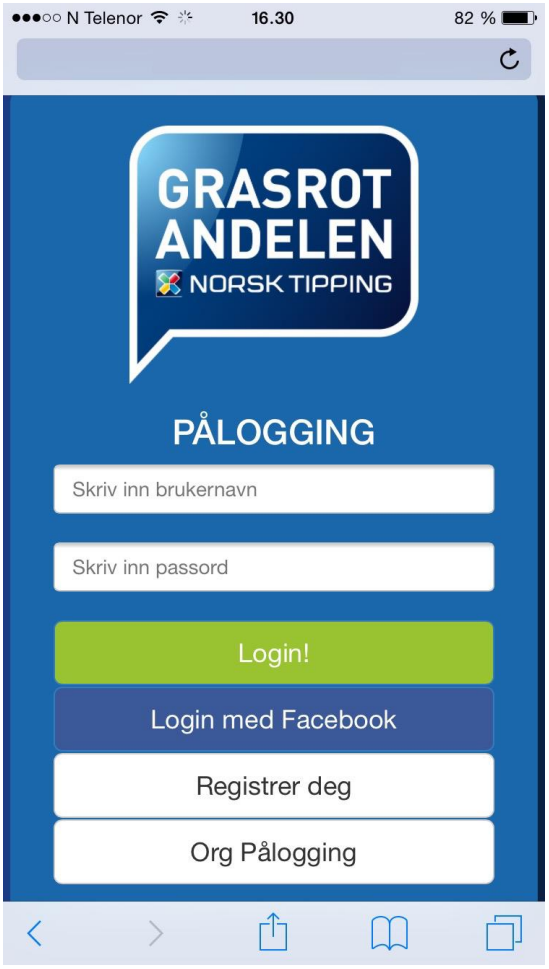
Når brukeren åpner applikasjonen for første gang vil personen først bli presentert for innloggingssiden. Første inntrykket er viktig enten personen logger inn via nettleseren på en PC eller en mobil enhet. På innloggingsiden, som man kan se på figur 5 nedenfor, har vi valgt å bruke Norsk Tippings Grasrot Andelen-logo som blikkfanger. Brukeren vil kjapt kjenne igjen Norsk Tippings logo, dette vil gi ett godt først inntrykk og skape trygghet hos brukeren. Ved å ha tittelen "Pålogging" under Grasrot Andelen-logen viser vi brukeren fort hvilken funksjon siden har. Uansett enhet vil innloggingsboksen inneholde 2 felter for inntasting av brukernavn og passord. Under feltene er det lagt 4 knapper, med tekst som beskriver de ulike funksjonene de utfører. Dersom brukeren logger inn via PC vil innloggingsboksen være venstrestilt, med et tydelig, enkelt og simpelt design, for å fremheve bakgrunnsbildet.



Figur 5 - Påloggingssiden på datamaskin

Ved innlogging via mobilapplikasjonen vil designet se noe annerledes ut. Tatt i betraktning at ved innlogging på en mobil enhet vil skjermen til brukeren sannsynligvis være mindre enn ved innlogging på PC så har vi skalert inn innloggingsboksen slik at den dekker hele

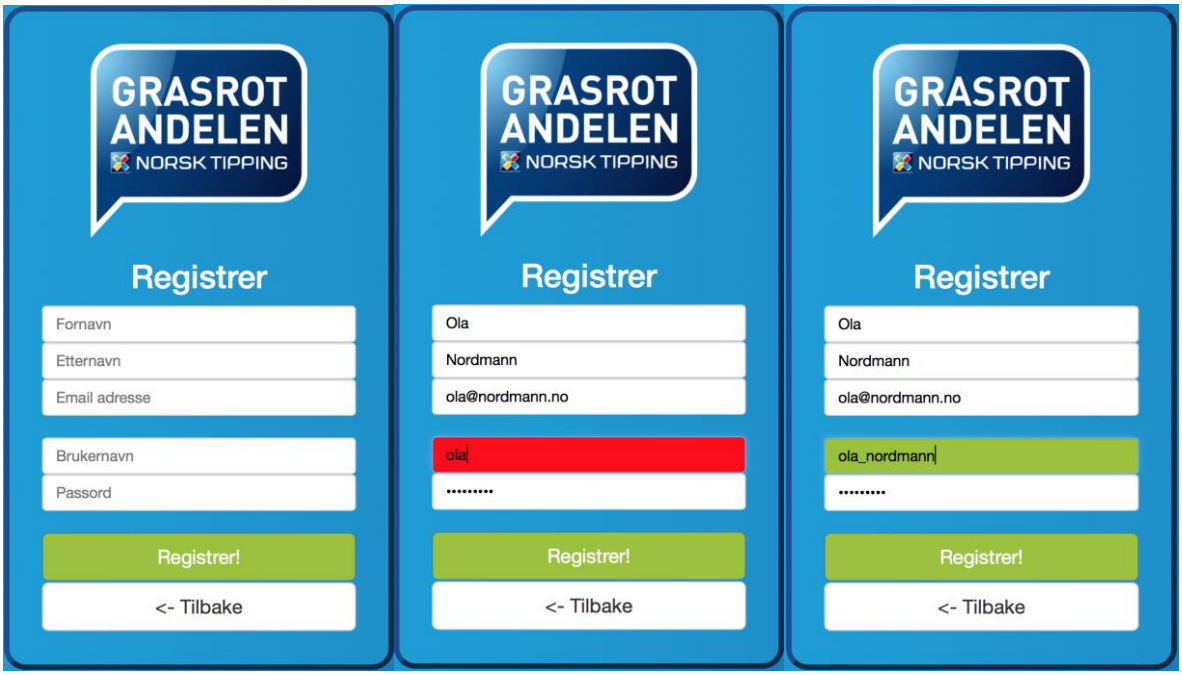
skjermen. Bakgrunnsbildet er også fjernet og byttet ut med en av de 2 blåtonene funnet i Grasrot Andelens designguide. Som illustrert på figur 6 under er feltene for inntasting av brukernavn og passord store og lett synlige. Knappen er store slik at det er lett og trykke på dem, samtidig som skriften har en passende størrelse og er lett leselig.



Figur 6 - Pålogging på smarttelefon

3.3.2 Registrer

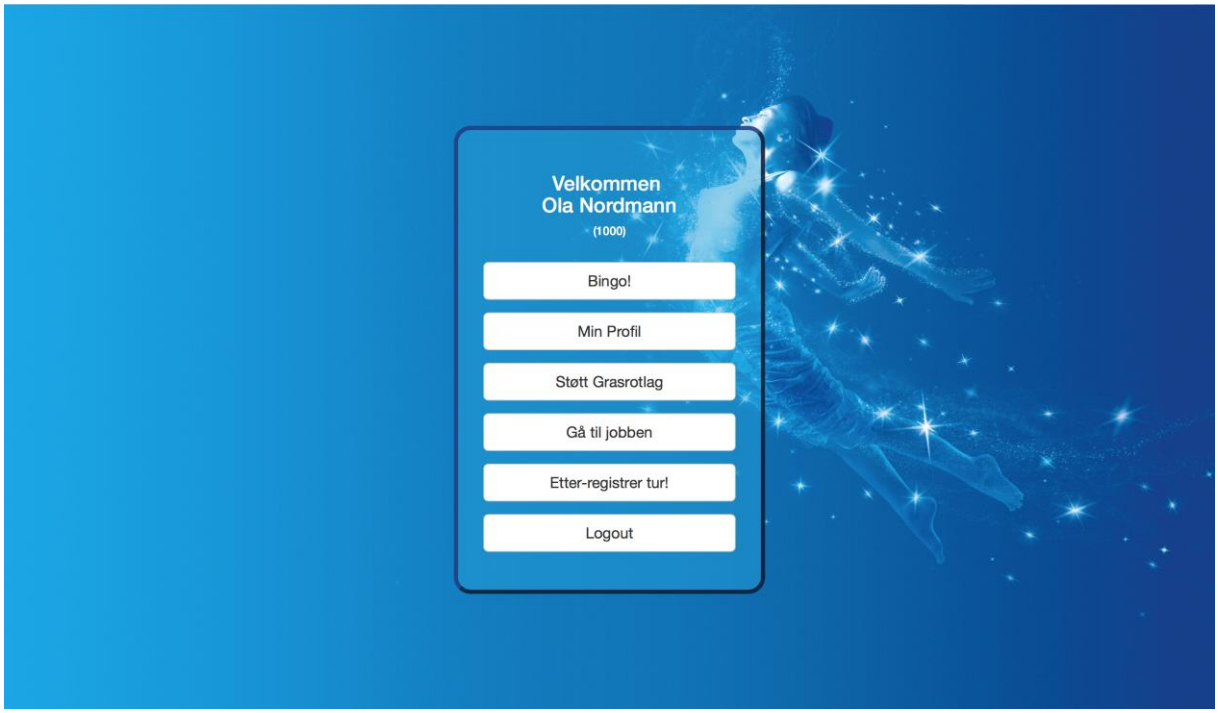
Registeringssiden er forholdsvis lik Påloggingssiden både på PC og mobile enheter. Forskjellen er brukeren blir presentert for flere felter som krever inntasting. Det blir vist et "Registreringsskjema" med 5 felter som må fylles inn. I disse feltene er det tekst som beskriver hvilken input som skal i hvilke felter, som vist på figur 7 under. Denne siden skalerer på samme måte som Påloggingssiden med tanke på skjermforskjell på PC og enheter med mindre skjerm. Vi har valgt å bruke fargekoder for å illustrer om brukernavnet er tilgjengelig eller ikke.



Figur 7 - Registreringssiden på smarttelefon

3.3.3 Meny

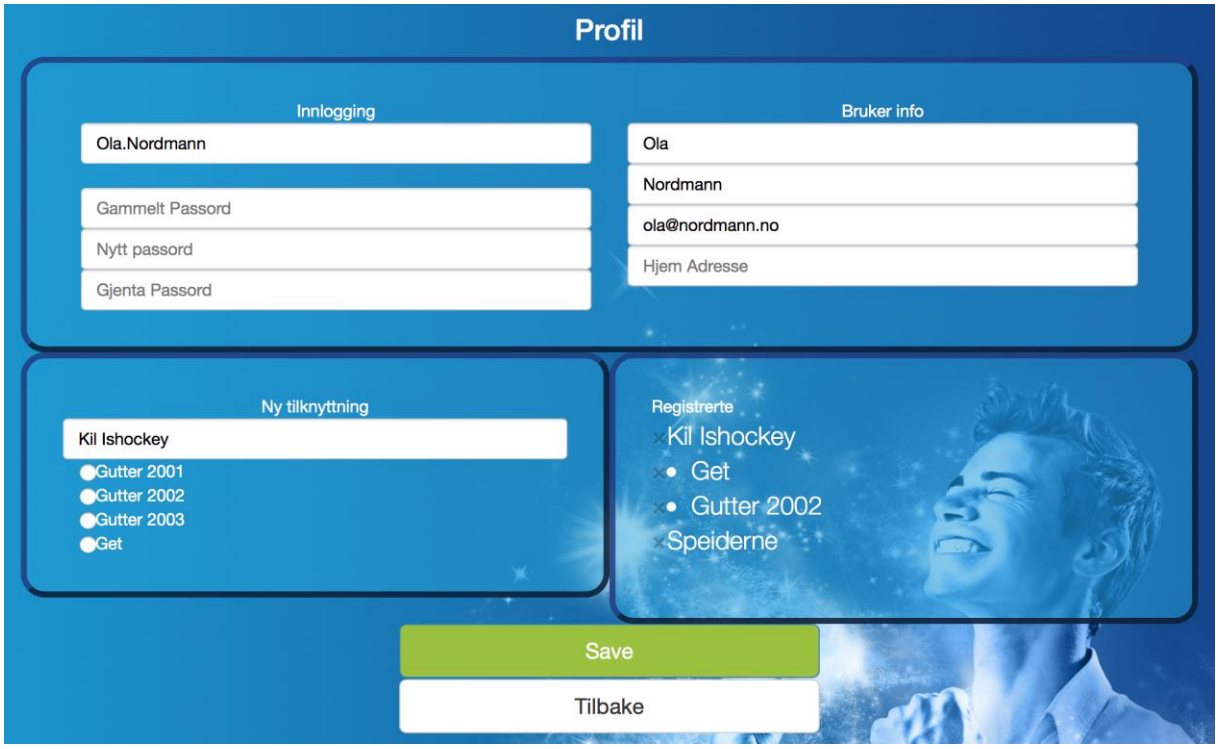
Når brukeren har registrert og logget inn blir den sendt direkte til Menysiden, som man kan se på figur 8 under. Her er boksen med innhold sentrert, som plasserer fokuset til brukeren rett på menyen. Øverst i meny-boksen står det "Velkommen" og på linjen under brukerens fornavn og etternavn. Under navn står antallet på brukerens opptjente Grasrotcoins i parentes. Under dette igjen kommer 6 knapper. På samme måte som resten av knappene i applikasjonen er knappene store og lett synlig. Teksten på knappene forklarer enkelt og presist hvilken funksjon i applikasjonen knappen representerer. Også skalerer elementene på skjermen i forhold til skjermens størrelse. Skjermdumpen under er tatt på en PC, mens på en smarttelefon vil meny-boksen dekke hele skjermen.



Figur 8 - Menysiden på datamaskin

3.3.4 Profil

På profilsiden til brukeren så er det blitt gjort ett forsøk på å sette opp ett logisk oppsett, så brukeren lett kan navigere seg rundt. Øverst er det bruker personalia og brukeren får mulighet til å endre infoen. Passordet blir ikke vist, for at ingen skal kunne sjekke dette om noen går vekk fra mobilen eller dataen. Nederst på siden så har brukeren infoen om alle grasrotlagene. På venstre side, så får brukeren muligheten til å legge til grasrotlag, og om grasrotlaget har undergrupper, så vises disse. Mens på høyre siden, så er det en liste over lagene brukeren støtter for øyeblikket. På mobil så vil de 4 delene satt over hverandre og skjermen er skalert slik at den passer på alle plattformer med hjelp av bootstrap.



Figur 9 - Profilsiden på datamaskin

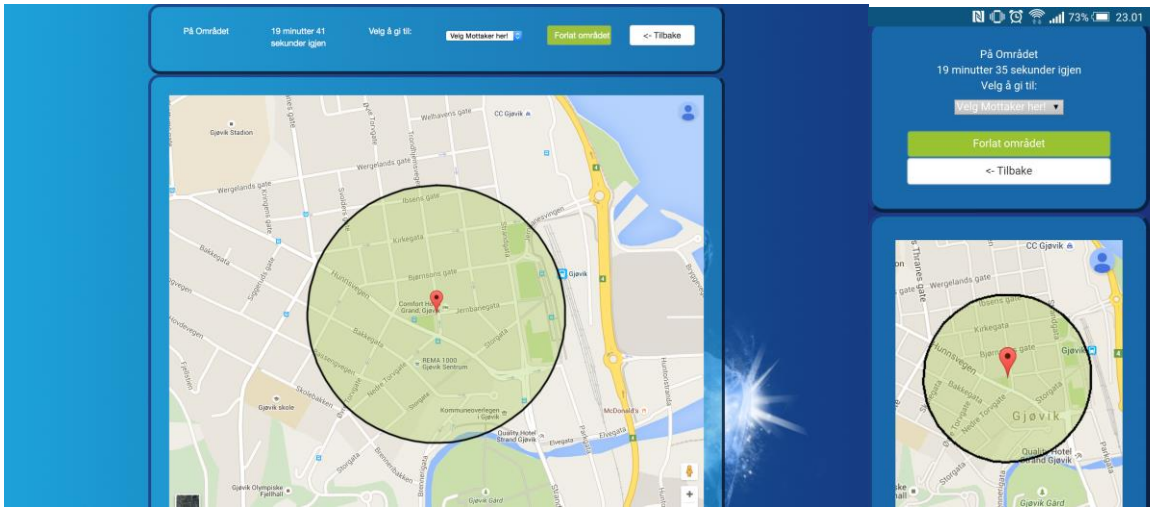
3.3.5 Bingo

På figur 10 under er et utsnitt av spillfunksjonen, bingoen. Her blir spilleren med engang opplyst om hva som foregår gjennom tittelen øverst på siden. Under tittelen er selve bingobrettet, som er et 5x5 spillbrett generert med tilfeldige tall i henhold til bingoens regler. I utgangspunktet er alle rutene blå, men dersom brukeren ønsker å markere en rute trenger bare brukeren trykke på ruten. Som et tegn på at ruten er markert blir ruten grønn. Under brettet er "bingo-knappen". "Bingo-knappen" er veldig visuell på grunn av størrelsen på den og fordi den har samme grønn farge som de markerte rutene.

Figur 10 - Bingobrettet

3.3.6 Støtt Grasrotlag

I vår "Støtt Grasrotlag"-funksjon vil brukeren få en oversikt over informasjon som er viktig får å kunne tjene Grasrotcoins. For at brukeren skal kunne vite om brukeren har vært på området i mer enn eller mindre enn 20 minutter, er det en dynamisk nedtelling fra 20 minutter midt på skjermen. Etter nedtellingen så vises det en melding som forteller brukeren at det nå går ann å forlate området eller fortsette å tjene opp Grasrotcoins ved å blir værende når tiden er telt ned. Det er tre knapper på siden, en knapp som viser hvilke Grasrotlag/Undergrupper brukeren kan gi Grasrotcoinsene til, en "Forlat Området" som lagrer Grasrotcoinsene til det valgte Grasrotlaget/Undergruppen og en "Tilbake" knapp som sender brukerne tilbake til menyen. På skjermen genereres også et kart med brukerens startposisjon så brukeren vet hvordan han/hun startet. Det blir også generert en sirkel med en radius på 250 meter rundt brukerens startposisjon så brukeren kan se hvor langt man kan bevege seg og fortsatt befinne seg innenfor distansekravet.



Figur 11 - Støtt Grasrotlag på datamaskin (venstre)

Figur 12 - Støtt Grasrotlag på smarttelefon (høyre)

3.3.7 Gå til jobben og Etter-registrer

Gå til jobben er enkelt presentert til brukeren. Den befinner seg i en boks midt på skjermen, som er i tråd med resten av designet i applikasjonen. I denne boksen kan brukeren hele tiden se tilbakelagt distanse som dynamisk oppdateres etter hvor brukeren beveger seg. På samme måte som på "Støtt Grasrotlag"-siden velger man Grasrotlaget/Undergruppen man ønsker å gi grasrotcoins til og trykker "Stopp" da turen er over. Når brukeren trykker "Stopp" vil grasrotcoins og data lagres og på brukerens skjerm blir statistikk om turen som, lengde, gjennomsnittshastighet, tid og antall opptjente grasrotcoins vist.

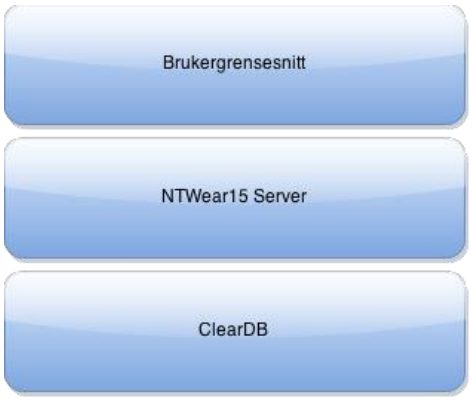
Etter-registrering skal være en enkel måte for brukeren å fylle inn informasjonen om turen brukeren var på. På skjemaet kan man enkelt velge distansen og fremkomstmetode for turen og man et av Grasrotlagene/Undergruppene brukeren støtter. Alt dette ved hjelp av avkrysning slik at det skal være lett for brukeren.

4 Arkitektur

I dette kapittelet blir det beskrevet om arkitekturen til applikasjonen. Dette innebærer at vi skal illustrere oppsettet bak det vi har utviklet. Dette vil være strukturen, eksempel på sekvensdiagram, fil organisering og database oppsett.

4.1 Trelag Struktur

En vanlig måte å dele opp ett slikt prosjekt på, er å dele det inn i lag. De forskjellige lagene representerer nivåene i applikasjonen. Med en trelagsstruktur så får du en fin oversikt over objektene i prosjektet, og det blir fulgt en "top-down"-struktur(6). Dette vil si at lagene kun henter data under seg. Som du ser på figur 13 under, så tilkaller brukergrensesnittet på funksjoner fra serveren, og serveren henter data fra databasen.



Figur 13 - Trelag-struktur

Det øverste laget er brukergrensesnittet og består av det brukeren ser, og har tilgang til på siden. Dette innebærer hva brukeren kan klikke på og fylle inn.

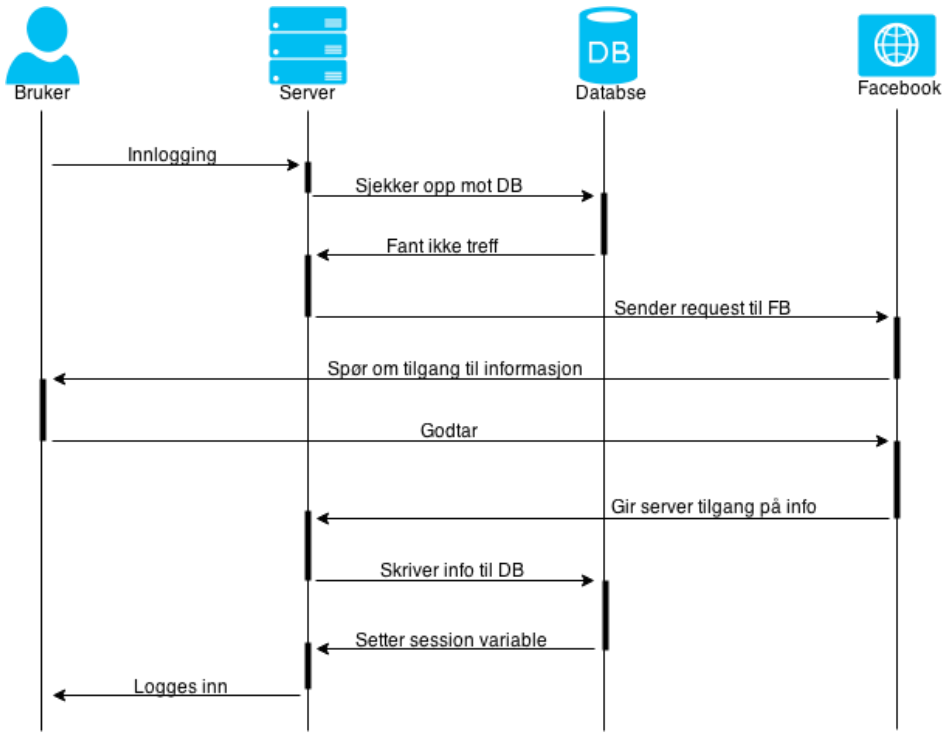
Det andre laget er serveren, og her ligger det mange funksjoner og data. Dette laget styrer funksjoner ut ifra brukerens handlinger.

I det nederste laget er databasen. Her ligger all data som er lagret og det er serveren som henter ut denne informasjonen, og dette ved hjelp av php-funksjoner fra serveren i vår applikasjon.

4.2 Sekvensdiagram

Vi har valgt å bruke et sekvensdiagram, se figur 14 under, for å illustrere hvordan Facebook-innloggingen fungerer da brukeren logger inn via Facebook for første gang. Vi ønsker å illustrere dette på en visuell og detaljert måte. Slik at vi kan se hvordan de forskjellige delene i den detaljerte use casen analysen forholder seg til hverandre. Ved bruk av sekvensdiagrammet er det tydeligere å se hvordan hendelsesforløpet til autorisering med Facebook fungerer og når brukeren har kontakt med databasen.

Vi antar her at brukeren ikke har bruker og logger inn for første gang med Facebook-knappen, og at han trykker godtar at vår applikasjon henter profil informasjonen hans fra Facebook.

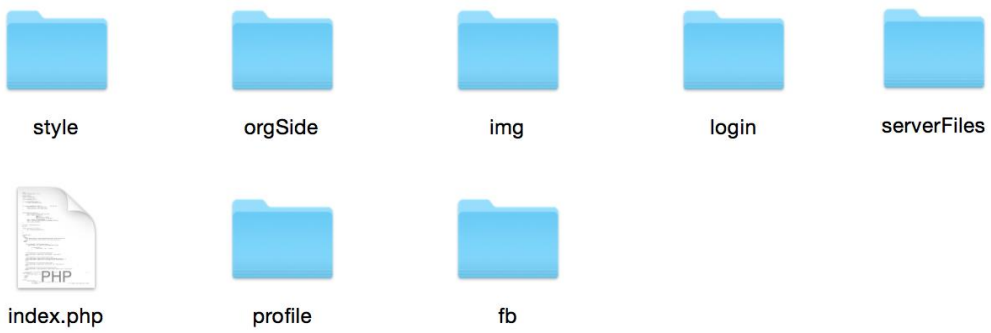


Figur 14 - Sekvensdiagram for Facebook-innlogging

4.3 Filstruktur

Det å ha god orden og en bra struktur er en selvfølge for at du skal kunne utvikle applikasjonen på en effektiv og oversiktlig måte. Det er viktig å ha ett logisk oppsett av mappestrukturen, og ha korte og presise navn slik at det er enkelt å navigere rundt, og med

tanke på url'en. På figur 15 så ser du hvordan vår filstruktur er satt opp i *rotmappen* på en god og oversiktlig måte.



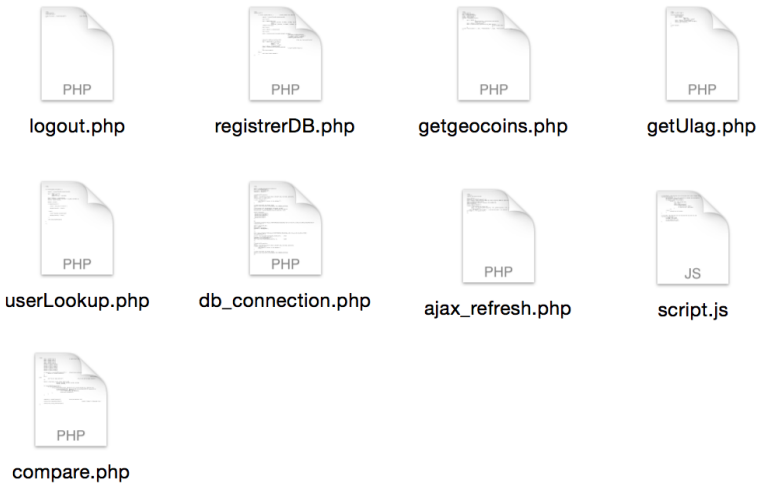
Figur 15 - Oversikt over applikasjonens filstruktur

Det er ingen tvil her om hvor du finner de forskjellige filene, og det er enkelt og finne den filen du vil endre på i senere anledninger. Vi har absolutt alle filene våres i mapper utenom index, som er menysiden til brukeren, og den første siden vi vil at brukeren skal komme inn på da han går inn på applikasjonen, eller adressen til nettsiden.

I rotmappen ,se figur 15 over, er vår egne og eksterne css, som inneholder bootstrap og NT sin css fil. Vi har valgt å ha alle bildene som er vist i applikasjonen i en egen mappe, så du lett kan finne dem uten å gå inn å sjekke på mappene, og her ligger også lisensene til bildene om dette har vært nødvendig. På profil mappen så ligger alle sidene som brukeren får se da han er innlogget, unntatt meny siden som er index beskrevet tidligere. Vi har valgt å kun ta en nærmere titt på en av mappene, siden vi ikke ser det hensiktsmessig å gå inn på alle sammen da det er veldig enkelt å forstå.

Mappen vi tok for oss og mente var en av de viktigste er serverFiles. Her har vi forsøkt å legge filene som skjer på serversiden. Tidligere har du sett ett eksempel på ett sekvensdiagram og hvordan requestene skjer imellom server og klient. I mappen så er det for det meste .php filer, men også .js. Grunnlaget for denne mappen er for å sortere ut de filene og funksjonene som blir brukt i flere av våre HTML eller PHP filer. db_connection.php er ett eksempel på en slik fil. Navnet er presist og forklarer hva den gjør. Denne filen

inneholder tilknytning til databasen vår og blir brukt av alle filene som trenger en tilknytning til databasen i applikasjonen.



Figur 16 - Oversikt over mappen serverFiles

4.4 Databasedesign

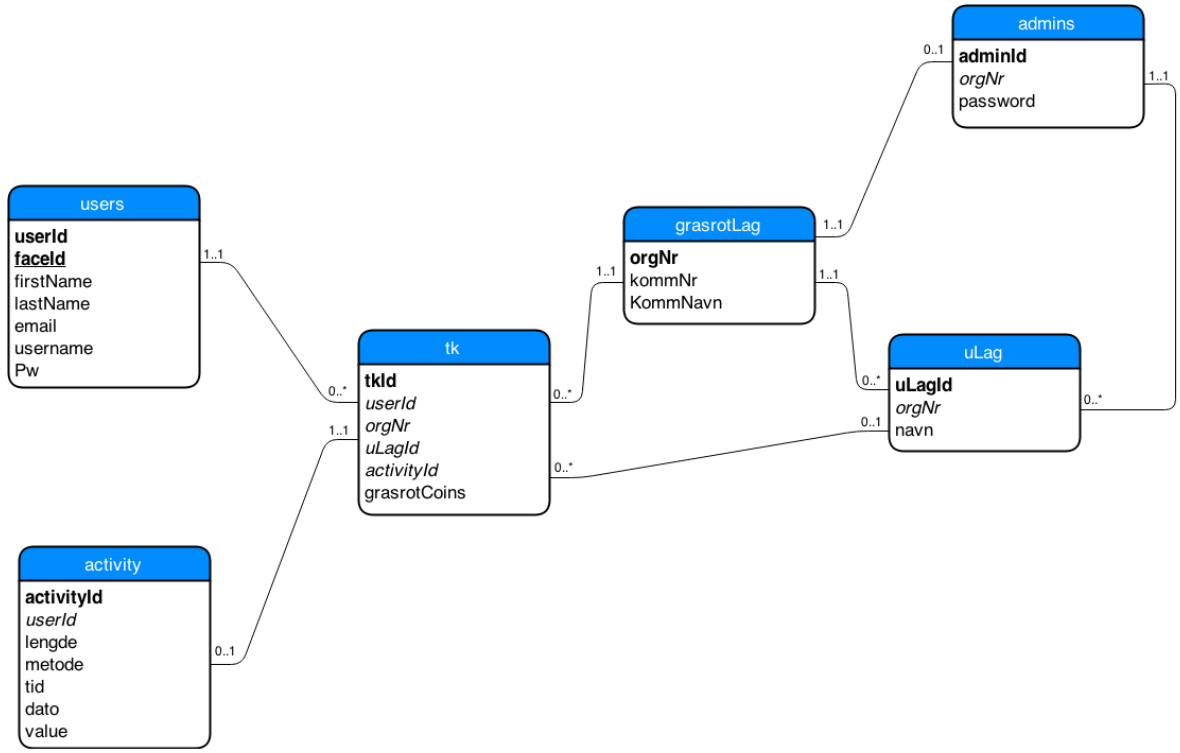
4.4.1 Bakgrunn

Før vi startet med å sette opp databasen vår, så var det visse ting vi måtte tenke på, logisk skjema og at tabellene hadde normalform. Vi tok hensyn til at databasedesignet hadde første-, andre- og tredje normalform. Vi passet også på om noen av tabellene hadde "mange til mange" relasjoner, så skulle det opprettes en ny tabell. Eksisterer det en "en til mange" eller "en til en" relasjon så må det være en fremmednøkkel.(7)

4.4.2 Design

I vår database så har vi endt opp med følgende struktur. Vi har satt opp 6 tabeller hvor alle har sin primær nøkkel. På figur 17, så er det illustrert hvordan databasen er satt opp. Vi har vist attributtene på følgende måter på figuren:

- Primær nøkkel ved tykk skrift
- Sekundær nøkkel ved tykk med strek under.
- Fremmed nøkkel er vist ved kursiv.
- Andre attributter er i vanlig tekst



Figur 17 - Relasjonsdiagram for applikasjonens database

4.4.3 Tabeller

users (userId, faceId, firstName, lastName, email, username, pw)

PN userId

SN faceId

tk (tkId, userId, orgNr, uLagId, activityId, grasrotCoins)

PN tkId

FN userId RELASJON users (userId)

FN orgNr RELASJON grasrotLag (orgNr)

FN uLagId RELASJON uLag (uLagId)

FN activityId RELASJON activity (activityId)

activity (activityId, userId, lengde, metode, tid, dato, value)

PN activityId

grasrotLag (orgNr, kommNr, kommNavn)

PN orgNr

uLag (uLagId, orgNr, navn)

PN uLagId

FN orgNr RELASJON grasrotLag (orgNr)

admins (adminId, orgNr, password)

PN adminId

FN orgNr RELASJON grasrotLag (orgNr)

4.4.4 Kommentar til databasen

Utover det som er nevnt over så har vi satt ett par variable til å være "allow NULL" i vår database. Dette vil si at vi lar celler i databasen mulighet til å være uten verdi. For brukeren betyr dette at han slipper og fylle ut dette feltet om han ikke ønsker det. For applikasjonen vil det si at for eksempel at en sekundærnøkkel ikke trenger å være satt. Attributtene vi har satt til å være "allow NULL" i applikasjonen er

- FaceId (users)
- activityId, uLagId (tk)

Vi har også satt opp databasen slik at grasrotcoins ligger i "tk" tabellen, se figur 17. Det ble vurdert og ha denne i en egen tabell og referere til user og tk, men endte med å ha den i tk. Ved denne løsningen så oppnår vi en god oversikt til senere utvikling og lage spørringer som finner ut hvor mange grasrotcoins en bruker har, samt se hvor hvilke organisasjoner som har mest aktive brukere og hvor mange grasrotcoins de har fått tjent opp.

5 Implementasjon

5.1 Utviklingsverktøy

I et utviklingsprosjekt er det viktig å bruke gode verktøy. Med gode verktøy blir prosessen mer strukturert, ryddigere og lettere. I dette prosjektet har vi brukt en del verktøy vi kjente til fra før, men vi har også brukt nye verktøy som vi ikke hadde erfaringer med. Noe som har gjort prosessen enda mer spennende og lærerik. Under er en liste over alle verktøyene vi har brukt.

5.1.1 Bluemix

Utviklingen av vår applikasjon har blitt gjort i en ny skytjenste levert av IBM, med navn *Bluemix*. Norsk Tipping skaffet oss igjennom sine kontakter hos IBM en lisens på Bluemix som varte gjennom hele utviklingsperioden. Bluemix gjøre det mulig for utviklere å lett skrive, bygge og modifisere kode, og det gir muligheten til å slippe applikasjoner på en sky, slik at du får alt samlet på en plass. Bluemix er en implementasjon av IBMs "*Open Cloud Architecture*" som er basert på *Cloud Foundry*. Cloud Foundry er en "open source" PaaS(*Platform as a Service*).(8)

Det som gjør Bluemix til en så interessant nye skytjenste er at man lett kan integrere tjenester som databaser og andre spennende funksjoner og APIer. Man kan søke i mange forskjellige tjenester og med et enkelt trykk kan man inkludere tjenesten eller APIet i sin egen applikasjon. Man slipper og sette opp tjenesten eller konfigurere den, alle tjenester kommer med kodeeksempler som gjøre ting mye lettere å anvende.

I utviklingen har vi brukt Bluemix til en rekke forskjellige ting. Vi opprettet først vår web-applikasjon slik at vi fikk en tilgjengelige web-server. Det var på denne web-serveren vi lastet opp funksjonalitet som vi hadde utviklet ferdig lokalt. Ettersom Norsk Tipping hadde skaffet oss et nytt og spennende verktøy som Bluemix, så ønsket vi å gjøre så mye som mulig i denne skytjeneste. Når vi skulle integrere en database med applikasjonen benyttet vi oss dermed av Bluemix sin søkefunksjon blant tjenester. I søket fikk vi opp flere forskjellige databaser, vil valget dermed den vi mente var mest hensiktsmessig og la den til i applikasjonen.

5.1.2 Git

I et utviklingsprosjekt er det viktig at alle i prosjektgruppen hele tiden har tilgang på den nyeste versjonen av arbeidet. Dette kan man oppnå ved hjelp av å bruke et versjonskontrollverktøy. En annen funksjon versjonskontroll har som gjør verktøyet så nyttig er fletting av filer. To utviklere kan jobbe på samme fil samtidig og, når utviklerne da laster opp sine versjoner av samme fil vil versjonskontrollen i nesten alle tilfeller klare å flette sammen filene til en fil. I dette prosjektet har vi brukt HubJazz, levert av IBM, som vår versjonskontroll. HubJazz er også implementert direkte i Bluemix som er den fordel for oss som utviklet i denne skyen.

5.1.3 Git klient

Som Git klient har vi brukt SourceTree. SourceTree er en frontend til Git som kommuniserer mellom filmappen på din datamaskin og Git. Grunnen til at vi valgte nettopp SourceTree som vår Git klient er a t vi har brukt SourceTree i tidligere utviklingsprosjekter, dermed kunne vi allerede bruke verktøyet på en god måte. Vi hadde også bare gode erfaringer med verktøyet på forhånd, så vi så ingen grunn til å ikke benytte oss av verktøyet

Det som har gjort SourceTree til et nyttig verktøy i en utviklingsprosess for oss er filoppdateringen og loggføring. SoureTree sjekker hele tiden om filene dine lokalt er like filene som ligger på Git. Dersom en av filene dine lokalt er av en nyere versjon enn samme filen på Git vil Sourcetree oppdatere den filen til Git. Samme gjelder andre vegen, er filen på Git av en nyere versjon enn samme filen lokalt, vil SourceTree laste ned den nye versjonen og oppdatere filen.

SourceTree fører også logg på alle opplastninger som blir gjort til Git. Gjennom SourceTree kan man se hvilken utvikler som har lastet opp, når de lastet opp, i hvilken fil noe er endret, hvilke endringer som er blitt gjort i filen eller eventuelt om filer har blitt slettet eller lagt til.

5.1.4 JIRA

Som verktøy har vi også anvendt JIRA i utviklingsperioden for å holde styr på hva som skal gjøre, når det skal gjøres og hvem som skal gjøre det. JIRA er et prosjektstyringsverktøy som har hjulpet oss med å beholde struktur og oversikt over

oppdraget. Vi i gruppen hadde lite erfaring med prosjektstyringsverktøyet, så da vår veileder ved HiG anbefalte JIRA, fortalte at skolen tilbød prosjektplass på JIRA og i tillegg tilbød seg å ordne brukere valgte vi å benytte oss av JIRA. JIRA fungere ypperlig sammen med utviklingsmodellen vi fulgte, Scrum. I JIRA satt vi opp prosjekts produktkø sammen med oppdragsgiver. Vi har også kjørt våre sprinter og hatt vår sprintkø i JIRA. Ved å bruke JIRA som prosjektstyringsverktøy har vi hatt muligheten til å estimere størrelsen på oppgaver og delegere dem til gruppemedlemmer.

5.1.5 Koderedigeringsverktøy

Kode har vi skrevet i fire forskjellige koderedigeringsverktøy i løpet av prosjektet. Disse fire er like på mange området og gjør så og si den samme jobben. Vi har brukt Notepad++, som kun er tilgjengelig på maskiner med operativsystemet Windows, Brackets, AptanaStudio3 og TextWrangler. Fordelen med å skrive kode i disse koderedigeringsverktøyene er at de støtter alle språkene vi har kodet prosjektet i. En annen fordel som gjøre disse, og stortsett alle lignende verktøy, så nyttige er fargekodene på koden. Da disse verktøyene støtter språkene vi koder i, klarer de å "lese" koden og gi forskjellige elementer forskjellige farger. Disse fargekodene gjør at koden er mye lettere å tyde, samtidig som det er lettere å ha oversikt over koden i filen.

5.1.6 Wamp

Wamp har vi brukt til lokal utvikling og hosting. Wamp gir muligheten til å kjøre apache-server, MYSQL-server og PHP-server lokalt. Wamp er kun tilgjengelig på maskiner med operativsystemet Windows, men det finnes lignende applikasjoner for både Mac og Linux. Ved å bruke Wamp som et utviklingsverktøy klarte vi å effektivisere arbeidet vårt veldig. Ved å teste og utvikle lokalt, for å så laste opp ferdige moduler til web-serveren, fjerner vi mulig problemer som kan forsinke utviklingen.

5.1.7 Sequel pro

Sequel Pro er ett MySQL database håndterings program for Mac OS X, som vi har benyttet oss av. Dette har gitt oss muligheten til å håndere dataene og tabellene i databsen vår på Bluemix. I sequel Pro så får du muligheten til å se tabell oversikt

over innholdet i databasen, kjøre SQL queries, og mye, mye mer som har vært nyttig for oss.

5.1.8 Tekstbehandlingsverktøy

Vi har i løpet av perioden brukt fire forskjellige typer tekstbehandlingsverktøy. Vi har brukt Microsoft OneNote og Notisblokk til å skrive logger og til å ta notater i møter med oppdragsgiver og veileder. I rapporten har vi brukt Microsoft Word og Word online. Word online er en versjon av Microsoft Word som kjører i nettleseren. Selv om vi skrev Forprosjektet ved hjelp av Google Docs valgt vi å skrive selve rapporten i Microsoft Word og Word Online fordi disse tjeneste tilbyr flere funksjoner som er veldig nyttig, enn Google Docs, i forbindelse rapportskrivingen.

5.1.9 Skype og Microsoft Lync

I løpet av perioden har vi brukt både Skype og Microsoft Lync som begge gir muligheten til å føre telesamtaler via internett. Skype har blitt meste brukt for å kommunisere innad i gruppen dersom medlemmene ikke hadde mulighet til å møtes på skolen eller lignende. Microsoft Lync har blitt brukt i forbindelse med Norsk Tipping. Det er via Lync vi har hatt demoer og møter i tilfeller da oppdragsgiver har vært bortreist eller vi ikke hadde mulighet til å møte på Norsk Tippings hovedkontor på Hamar.

5.1.10 Draw.io

Prosjektet har krevet en dele figurer, tegninger, diagram og modeller. Vi har så lagt det lar seg gjøre, valgt å lage våre figurer selv. Disse er laget i en nettsiden som heter Draw.io. I Draw.io er grafisk verktøy med et utrolig bredt spekter av muligheter. Draw.io tilbyr mange funksjoner som er svært nyttige i forbindelse med visualiering av design og arkitektur i en applikasjon.(4)

5.1.11 Dropbox og OneDrive

All kode har vi hatt liggende på Git, vi har også valgt å ha alle dokumenter som har oppstått i løpet av prosessen liggende på nettet i en skytjenste. Når vi skrev Forprosjektet til oppdraget lå alle dokumenter på Dropbox, men når vi startet på på selve på prosjektet hadde vi dokumenter liggende på Microsoft sin skytjeneste, OneDrive. Det er ingen spesiell grunn til at vi byttet skytjeneste etter Forprosjektet.

Det var av den enkle grunn at de fleste av dokumentene våres ble skrevet i Microsoft produkter. Dette gjorde at det var mye enklere å lagre dokumentene på OneDrive.

5.1.12 Phonegap

Vår oppdragsgiver ønsket at applikasjon som kunne brukes på Iphone og helst flere plattformer. Phonegap sin build-funksjon krever kun at man laster opp en zippet mappe med filen i sin web-applikasjon. Så alt vi trengte å gjøre var å laste opp en zippet mappe med våre PHP/html, javascript og css filer så bygde Phonegap applikasjonen vår og ga oss en installasjonsfil som installerer applikasjonen. Da vi fikk tips om at dette var en bra løsning på å gjøre koden om til en applikasjon som fungere "*Cross-Platform*" valgte vi å bruke Phonegap sin build-funksjon til dette.(9)

5.2 Språk vi har brukt

5.2.1 HTML

HyperText Markup Language, er det mest kjente språket da det kommer til web-programmering og blitt en standard da det kommer til dette. I de aller fleste nettsider så er det HTML som blir brukt, og dette er ikke uten grunn. HTML har ett logisk oppsett og dermed lett å bruke og lett å sette seg inn i. Kjennetegn på at det er ett HTML dokument er at det slutter alltid på .html på filnavnet. Det kan også bli brukt HTML i andre filtyper som .php uten at du initierer at det er en HTML fil. I dokumentet så er alle deler delt opp i "tags". Og alle taggs forteller nettleseren hva som kommer av tekst. Om det er styling, import av filer, etc. Ett HTML dokument er delt inn i to hoveddeler, <head> og <body>. Hvor det i head blir definert filen og import av filer, mens i body taggen så er all koden av det som blir vist på siden.

Under er det illustrert hvordan HTML språket er brukt. Dette er en eksempel av koden vi har brukt i applikasjonen, og det blir brukt flere attributter og annet i språket som det blir lagt mer vekt på i implementasjon kapittelet.

```
<body>
    <form action="../index.php">                          <!-- tilbake knapp -->
    <input type="submit" value="<- Tilbake" class="save"/>
    </form>

  <div>
    <header>
      <h1 class="green2">B I N G O !</h1>
    </header>
```

Kodeeksempel 1 - Eksempel på bruk av HTML

5.2.2 CSS

Cascading Style Sheets blir brukt for å style nettsiden din. Det er flere måter å style i HTML. Du kan for eksempel ha all din css øverst i filen med <style> tagen, som vist på kodeeksempel 2. Tekst 1 på bildet ved siden av er ett eksempel på dette. Det er også mulig å importere en egen css fil ved hjelp av <link> taggen, som er en av de mest vanlige metodene. På bildet så har tekst 3 fått stylingen på denne metoden. Om det bare er ett element som skal ha en liten css endring så er det også vanlig å bruke inline metoden, hvor du kaller på style attributtene og setter inn hva du vil ha. For eksempel på tekst2 på bildet under. Den siste og svært lite brukte metoden er å importere en css fil fra en css fil. Dette gjøres ved å bruke "@import "nystyle.css";" på din css fil. Dette kan være hensiktsmessig hvis du for eksempel bruker mange css filer, og ikke vil include alle på toppen av hver nye side du lager, men bare linke til en css fil hver gang.(10)

```
<!DOCTYPE html>
<html>
  <head>
    <title>Tittel på siden</title>
    <link rel="stylesheet" type="text/css" href="style.css">
    <style>
        .tekst1 {
            color: red;
        }
    </style>
  </head>
  <body>
    <h1 class="tekst1">Tekst 1</h1>
    <h1 style="color: blue">Tekst 2</h1>
    <h1 class="tekst2">Tekst 3</h1>
  </body>
</html>
```

Tekst 1

Tekst 2

Tekst 3

Kodeeksempel 2 - Illustrasjon på forskjellige måter å implementere css

5.2.3 PHP

PHP er ett *backend* språk. Dette vil si at php er ett server programmeringsspråk, som betyr at all koden som skjer i PHP skjer på server-siden, og at klienten ikke ser operasjonen som

blir gjort. PHP er også det programmeringsspråket vi har brukt som er mest likt C++. Om det skal brukes PHP i en av filene i applikasjonen, så brukes det fil endingen .php. Ved bruk av PHP så får du muligheter som å ta i bruk mange funksjoner som php tilbyr. Dette kan være standard kode som løkker og annen kode som if / else, men det kan også være annen kode som sessions, som lar deg lagre variable i nettleseren til du lukker den. Dette lar deg lagre variable som hvem du er, navn også videre, så brukeren kan forbli pålogget da han oppdaterer en side. PHP blir hovedsakelig brukt på funksjoner for applikasjonen, siden alt skjer på serversiden og du får tilgang til flere funksjoner som PHP har. I kodeeksempel 3 er det illustrert ett eksempel på hvordan vi har brukt PHP og sessions i vår applikasjon.

```php
<?php
session_start();                              //For å ta ibruk sessions

echo "Welcome " . $_SESSION['name'] . "!";      //Innloggings melding, (fult navn)
```

Kodeeksempel 3 - Eksempel på bruk av PHP

5.2.4 MySQL

MySQL er database språket vi har brukt. Grunnlaget for at vi valgte å bruke MySQL er for at vi hadde hatt erfaring med dette tidligere. I eksempelet under så er det illustrert hvordan en av våre funksjoner bruker MySQL i samarbeid med PHP for å hente ut alle underlagene til en organisasjon og returnerer dette i en array.

```php
<?php
$orgNr = $_POST['orgNr']

function getUlag($orgNr){
    global $dbConn;
    $sql = "SELECT *
            FROM ulag
            WHERE orgNr = :orgNr";
    $stmt = $dbConn -> prepare($sql);
    $stmt -> execute(array(":orgNr"=>$orgNr));
    return $stmt->fetchAll();
}
?>
```

Kodeeksempel 4 - Eksempel på bruk av MySQL i samarbeid med PHP

5.2.5 JavaScript

Javascript, ofte forkortet til js, er et skriptspråk som er støttet av alle de store nettleserne. Javascript blir brukt til å utføre handlinger på klient ettersom brukeren foretar seg noe. Det vil si at js kan tilføre eller endre elementer dynamisk på brukerens nettleser uten at

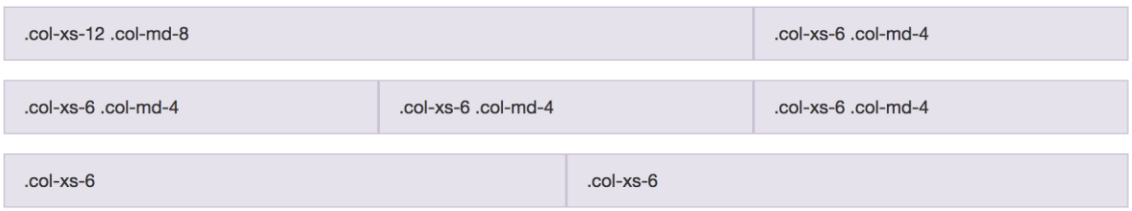
brukeren trenger å oppdatere siden. Det er viktig å vite at js ikke kan kjøres på serveren, men kun på klienten. Javascript har heller ikke tilgang til å modifisere filer eller bruke programmer på brukerens enhet. Vi har benyttet oss mye av Javascript i utviklingen av vår applikasjon. Et eksempel på bruk av Javascript er i vår "Støtt Grasrotlag"-funksjon. Her kjøres en js-funsksjon på brukerens klient som gjør at en digital klokke og telle ned fra 20 minutter, synlig, midt på skjermen med engang brukeren laster inn «Støtt Grasrotlag». Alt dette uten at siden må oppdateres hele tiden, på grunn av bruk av Javascript.

5.3 Bilbloteker & API'er

5.3.1 Bootstrap

Vi har også valgt og tatt i bruk bootstrap i vår løsning. Bootstrap 3 bruker ett mobile-first stil oppsett. Det vil si at ved bruk av denne stylesheeten så vil den automatisk være på style til mobil før hvordan nettsiden vil se ut på en pc eller mac.

Noe av det mest kjente med bootstrap er deres grid system. Dette er ett system som gjør det enkelt å plassere og håndtere utseende på applikasjonen / websiden. Bootstrap sitt grid system går i ett såkalt 12 system. Om du tenker deg en tabell, så skal hver rad ha 12 kolonner. Som vist på bildet under så vil du se ett eksempel på hvordan dette systemet funker. De har gjort det enkelt med å dele inn om du er på mobil eller en datamaskin. Måten du bruker grid systemet på er at du lager en tagg rundt innholdet du vil plassere en plass på siden, og setter bootstrap sin class til denne taggen. eks: . Med denne taggen så vil diven vises på halve skjermen på mobilen. Bootstrap har delt opp klassene sine i xs- som er mobil, sm- som er tablets, md- laptops, og lg- som er store skjermer. Grid systemet eksisterer gjerne inne i en .container tagg.



Figur 18 - Bootstraps grid system

Som vist på bildet over, og på første raden, så er første kolonne ".col-xs-12 .col-md-8" Dette vil si at på xs, altså mobiltelefoner så vil denne delen ta 12/12 deler av siden og dermed dekke hele siden. Og på datamaskin bare ta 8/12 deler, og de gjenværende 4/12 deler gå på høyre side på samme rad, mens på mobil vil denne dekke halve siden på ny rad. Bootstrap fungerer også slik at om den andre kolonnen i første raden hadde vært større så raden ikke gikk opp i 12, så ville den wrappet om automatisk og bare vært 8/12 deler og wrappet om det gjenværende. I bootstrap har du også muligheter til å sette en kolonne offset. Dette vil si at du kan sette tomme kolonner så før du vil at din div skal vises for eksempel 4/12 deler inn på siden. Dette vises ved å bruke klassen "col-md-offet-4".

5.3.2 jQuery

jQuery er ett bibliotek og hjelpemiddel til JavaScript. jQuery er ett frontend språk til å modifisere brukerens HTML. Med bruk av jQuery så gir det deg muligheten til å oppdatere deler av applikasjonen din uten at siden trenger å lastes på nytt. Ett eksempel på dette kan være innholdet i en tagg, eller om en hel div skal fjernes eller vises om brukeren trigger noe i programmet. Ved å ta i bruk jQuery, så gjøres det ved å bruke script taggen.

```
<script src="http://code.jquery.com/jquery-1.11.0.min.js"></script>
```

Kodeeksempel 5 - Eksempel på inkludering av jQuery

For at jQuery skal kunne oppdatere noe på siden, så må det en hendelse til. En hendelse er noe brukeren gjør for eksempel og endre input i en tekstboks, eller trykker på en knapp. Når denne hendelsen tilkalles gjerne en funksjon som gjør noe med stylingen eller oppdaterer innholdet. Ett eksempel på dette er på vår organisasjons side. Her har vi en animert effekt på en div da brukeren trykker på en av knappene, og en ny rute vil vises. Som vist på kodeeksempel 6 under så vil reg-diven på kodeeksempel 7 vises da knappen med id regi klikkes.

```
<script>
$(document).ready(function(){
  $("#regi").click(function(){
      $("#reg").slideDown();
      $("#slett").slideUp();
      $("#endr").slideUp();
  });
```

Kodeeksempel 6 - Eksempel på bruk av jQuery

```html
<div class="row" id="reg">
  <div style="text-align: left" class="col-md-3 col-md-offset-1 loginbox orgBox">
    <form method="post">
        Nytt ulag: <input type="text" name="navn" /><br>
        <input type="submit" class="btn btn-default btn-lg btn-block knapp" name="add" value="Add!" />
    </form>
  </div>
</div>
```

Kodeeksempel 7 - Eksempel på bruk av jQuery i HTML

5.3.3 Ajax

Ajax er ett hjelpe bibliotek som gjør det enkelt og elegant håndtere data til og fra server siden i bakgrunnen uten at brukeren merker det. Dette gjør at vi kan sende og kjøre SQL spørringer og annen PHP kode i andre filer og returnere data, uten at brukeren trenger å oppdatere siden, for å oppnå en dynamisk og mye mer elegant side. Ett eksempel på dette har vi brukt på vår registrer side, hvor det blir sjekket om brukernavnet finnes i databasen ifra før. Om dette er tilfelle, så blir feltet markert rødt øyeblikkelig, og brukeren slipper å trykke registrer knappen og sjekke om det gikk eller ikke, og han må fylle inn all info på nytt fordi siden oppdatertes.

```javascript
<script>
  $("#username").change( function(){
      $.ajax({
          type: "post",
          url: "../serverFiles/userLookup.php",
          dataType: "json",
          data: { "username": $("#username").val() },
          success: function(data,status) {
              if (data['exists'] == "true")  {
                  $("#checkUsername").css("color","red");
                  $("#username").css("background-color","red");
                  $("#username").focus();
              }
              else {
                  $("#checkUsername").css("color","");
                  $("#username").css("background-color","rgb(154,195,49)");
              }
          }
      });
  });
</script>
```

Kodeeksempel 8 - Eksempel på bruk av Ajax

Som du ser på kodeeksempel 8 over så blir PHP'en og SQL'en kjørt hver gang det er en endring I feltet med id username. Ajax tar og sender innholdet i feltet til PHP filen vist på kodeeksempel 9 under, og serveren sjekker om det eksisterer i databasen. Så returneres det i tekst om det er treff eller ikke. Når Ajax funksjonen mottar variabelen ifra PHP filen, så blir det sjekket om den er "true". Er den det så finnes brukernavnet i databasen og feltet vil bli rødt, Det blir grønt så fort det er skrevet noe i feltet og det ikke finnes i databasen. Brukeren slipper da og teste seg frem på hvilke brukernavn som er tatt.

```php
<?php
if (isset($_POST['username'])) {
    require 'db_connection.php';
    $sql = "SELECT username
            FROM users
            WHERE username = :username";

    $stmt = $dbConn -> prepare($sql);
    $stmt -> execute( array(":username" => $_POST['username']));
    $record = $stmt->fetch();

    $output = array();

    if (empty($record)) {
        $output["exists"] = "false";
     } else {
        $output["exists"] = "true";
     }

     echo json_encode($output);
}
?>
```

Kodeeksempel 9 - PHP-filen ajax tilkaller

5.3.4 Facebook

Vi har tatt i bruk Facebook SDK for å få Facebook-innlogging på vår applikasjon. For å få til dette så må det opprettes en applikasjon på developers.facebook.com som kommuniserer med din server. Du må også importere de nødvendige filene. For oppsett av Facebook, se Vedlegg 9.8, Facebook.

```
Copyright 2014 Facebook, Inc.

You are hereby granted a non-exclusive, worldwide, royalty-free license to
use, copy, modify, and distribute this software in source code or binary
form for use in connection with the web services and APIs provided by
Facebook.
```

Figur 19 - Facebooklisens

5.3.5 Google Maps API og HTML's Geolocation API

I utviklingen av applikasjonen har vi også benyttet oss av Google Maps API og HTML's Geolocation API. Siden vi har laget en web-applikasjon så har vi brukt HTML's Geolocation API til å hente brukerens startposisjon og brukerens posisjon ettersom brukeren beveger seg.(5) Får å generere kart med brukerens startposisjon markert bruker vi Google Maps API. Vi bruker også Google Maps API til å lage en sirkel som markerer 250 meters radius rundt posisjonen.(11)

5.4 Sikkerhet

Sikkerhet er utrolig viktig i applikasjoner som krever innlogging, registering og lagring av brukerdata. Som utviklere av applikasjonen er det vårt ansvar å sørge for at brukerne er trygge når de bruker vår applikasjon.

5.4.1 SQL Injection

SQL Injection er en kode teknikk for å forbigå SQL statements og lignende i applikasjoner og web sider. Om du ikke tar hensyn til SQLI når du programmerer, så kan en som kan litt om SQL Injection for eksempel droppe hele tabellen din. Dette gjør han så enkelt som å bare skrive inn en SQL spørring som gjør dette, slik som du ser i figur 20 under til venstre. På denne figuren er det vist en enkel løsning på å forbigå innloggingen om du setter passordet eller brukernavnet lik dette, siden denne spørringen 1=1 alltid vil være sann.


```php
$sql = "SELECT *
        FROM users
        WHERE username = :username
        AND password = :password
        ";

$stmt = $dbConn -> prepare($sql);
$stmt -> execute(array(':username' => $_POST['username']
```

Figur 20 - Eksempel på SQL-injection (venstre)

Kodeeksempel 10 - Sikring mot SQL-injection (høyre)

For å forhindre dette, så bruker du ikke variablene som brukeren skriver inn direkte i SQL spørringen, men du henter variabelen i execute funksjonen. På den måten så blir ikke det brukeren skriver inn en del av spørringen siden du bare definerer en variabel etter at spørringen er kjørt. På kodeeksempel 10 over, så er det illustrert hvordan vi har løst dette problemet

5.4.2 SHA1

For å kunne sikre passordet til brukerne av applikasjonen, så har vi benyttet oss av SHA1. SHA1 er en hashe teknikk som krypterer inputen til brukeren i en 40 siffer kode. Ved å bruke kryptering av passord så oppnår vi også at passordet i databasen ikke blir lagret i klar tekst og det blir umulig for de som har administrativ tilgang til applikasjonen og databasen og se hva passordet for brukerne av applikasjonen er. Vi vurderte også andre hashe metoder som MD5 og SHA3, men endte til slutt med SHA1. Dette gjorde vi på grunnlag av at vi ville ikke bruke MD5 da dette var et javascript hashing språk. Vi er også klar over at SHA3 er ett mye nyere og forbedret versjon av SHA1, men SHA1 tar mindre plass i databasen. Så vi tok en beslutning og mente at dette holdt som sikring av passord for vår applikasjon.

5.5 Kodet funksjonalitet

5.5.1 Eksempel på bruk av DB

I mappen serverFiles illustrert tidligere så har vi en fil med navn db_connection.php. Denne filen er svært sentral i vår applikasjon og blir brukt i svært mange av våre filer. db_connection er en fil som gjør så vi får tilgang til databasen på applikasjonen vår. Vi kan hente ut informasjon og skrive til databasen. Vi har valgt å bruke PDO(PHP Data Objects) som tilknytning. Og vi har valgt å kalle vår connect variabel som dbConn. Se kodeeksempel 11 under på hvordan PDO connection er satt opp på vår applikasjon.

```php
<?php
$host = 'database-url';
$db = 'database-navn';
$username = 'brukernavn';
$password = 'passord';

//DB connection
try {
$dbConn = new PDO("mysql:host=".$host.";dbname=".$db, $username, $password);
$dbConn->exec("set names utf8");

} catch (exception $e) {
    echo "Unable to connect to the database!?";
    exit();
}

// Viser error da du prøver å connecte til db
$dbConn->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
?>
```

Kodeeksempel 11 - filen db_connect.php

Ved bruk av databasen så har vi valgt ut ett eksempel på kodeeksmepel 12. Dette er ett utdrag fra filen vår orgPage, som er organisasjon siden, etter at en org er logget inn. Her har vi hentet ut to SQL funksjoner som bruker SQL statements til å hente informasjon ut av databasen. Før du får brukt denne koden, så må du require eller include db_connection.php filen. Fordelen med require er at du krever at programmet finner filen. Den første funksjonen getOrg(), henter infomasjonen itl en bestemt organisasjon og fetcher kunn ett record i databasen. Mens i den andre funksjonen getInfo() så henter den all underlagene til organsisasjonen den finner og legger de i en array med fetchAll() funskjonen.

```php
require '../serverFiles/db_connection.php';

function getOrg($orgNr){                //henter infoen om ett bestemt grasrotlag
    global $dbConn;
    $sql = "SELECT *
            FROM grasrotorg
            WHERE orgNr = :orgNr";
    $stmt = $dbConn -> prepare($sql);
    $stmt -> execute(array(":orgNr"=>$orgNr));
    return $stmt->fetch();
}

function getInfo($orgNr){               //henter alle underlagene til en organisasjon
    global $dbConn;
    $sql = "SELECT *
            FROM ulag
            WHERE orgNr = :orgNr";
    $stmt = $dbConn -> prepare($sql);
    $stmt -> execute(array(":orgNr"=>$orgNr));
    return $stmt->fetchAll();
}
```

Kodeeksempel 12 - Eksempel på SQL

5.5.2 Bootstrap

På kodeeksempel 13 under, er det illustrert hvordan vi har brukt klassene I bootstrap for å lage vårt design. Fra bilblioteket til bootstrap så har vi benyttet oss av grid systemet. Om du ser på kodeeksempel 13 så ser du at vi har satt denne div'en offset 1/12 og 3/12 bred. Vi har også brukt xs klassen som gjør at denne diven vil vises over hele skjermen på en mobil. Innputten vår er også hentet fra en bootstrap som gjør det enkelt og at den strekker seg over hele diven, på en ellegant måte om du er på mobil eller laptop. Knappen loginbruker også mye av bootstrap biblioteket, men har også en egen klasse som heter "knapp" som vi har modifisert koden til å være tilpasset vår applikasjon i vårt egen bibliotek.

```php
<div class="form-group col-xs-12 col-md-3 col-md-offset-1 loginbox">
    <img src="img/Grasrotandelen_logo_RGB.png"/>
    <h3>PÅLOGGING</h3>

    <form method="post">
        <input class="form-control" type="text" name="username" placeholder="Skriv inn brukernavn"/> <br>
        <input class="form-control" type="password" name="password" placeholder="Skriv inn passord"/> <br>
        <input type="submit" name="loginForm" value="Login!" class="btn btn-primary btn-lg btn-block knapp" />
    </form>
```

Kodeeksempel 13 - Eksempel på Bootstrap

5.5.3 Norsk tipping sin css

Vi har også fått tilgang til Norsk tipping sin hovedside CSS fil. Her har vi hentet ut elementer som font type, til vår egen css fil kalt style.css. alt vi har hentet fra css filen har vi referert til med comments i vår css fil, ved å bruke taggen /* og */.

5.5.4 Innlogging

På vår innloggings side, så løste vi innloggingen ved å ha bruker info på databasen og brukte sql og PHP for å sjekke opp mot databasen om brukeren fantes. Under ser du SQL spørringen som sjekker om den finner noe i databasen. Lengre ned i koden så er det sjekk om det blir returnert noe eller ikke, og deretter heading til index siden, med å sette sessions variable, eller om brukernavnet og passordet er feil. Siden passordet er kryptert i databasen med SHA1, så må det hashes for å kunne sammenlignes med det som ligger i databasen. At det må hashes vil se at det brukeren skriver inn må bli kryptert til sha1, siden det er slik den ligger på databasen.

```php
<?php
session_start();                                //for bruk av sessions i PHP
require '../fb/fbauth.php';

if (isset($_POST['loginForm'])) {               //da login blir trykket

    require "../serverFiles/db_connection.php";     //connect til DB

    $sql = "SELECT *
            FROM users
            WHERE username = :username
            AND password = :password
            ";

    $stmt = $dbConn -> prepare($sql);
    $stmt -> execute(array(':username' => $_POST['username'] ,
                           ':password' => hash("sha1", $_POST['password'])));
    $record = $stmt->fetch();       //Fetcher en eller ingen treff
}
?>
```

Kodeeksempel 14 - Kryptering av passord

For å logge inn med Facebook, så har vi brukt deres API og I kodeeksempel 15 under så er det vist hvordan dette har blitt brukt. Arrayen du ser har public_profile som er basic personal info til navn, etternavn, og en facebookId som vi skriver til db om den ikke eksisterer. Se usecase om Facebook- innlogging, se avsnitt 2.3.3 Detaljert use case analyse. Email må også spørres om med egen parameter for at vi skal få tilgang til den. Se vedlegg 9.8, Facebook, for koden som er brukt for innlogging med Facebook.

```php
<?php
echo "<a class='btn btn-primary btn-lg btn-block knapp face' href='" .
$helper->getLoginUrl(array('public_profile', 'email'))  .
"'>Login with Facebook</a>";
?>
```

Kodeeksempel 15 - Facebook SDK

5.5.5 Støtt Grasrotlag

Når vi laget funksjonen "Støtt Grasrotlag" tenkte vi lenge på hvordan vi skulle innfri kravene Norsk Tipping hadde satt for den funksjonen. For å innfri kravet om at brukeren måtte være Støtt Grasrotlag i mer enn 20 minutter før brukeren kan begynne å tjene opp Grasrotcoins. Vi startet med å kjøre ett lite Javascript som henter tidsobjektet i det brukeren starter "Støtt Grasrotlag" funksjonen. Ut av tidsobjektet henter vi klokkeslettets time, minutt og sekund og regner alt om til sekunder. Som du kan se i kodeeksempel 16 under.

```
<script>
    var starttid = new Date();
    var startTime = starttid.getHours();
    var startMinutt = starttid.getMinutes();
    var startSekund = starttid.getSeconds();
                                //Klokkeslettet regnet om til sekunder
    var startRegnet = ((startTime)*60)*60+ startMinutt*60 + startSekund;

</script>
```

Kodeeksempel 16 - Her henter vi data og klokkeslett fra brukerens enhet.

Grunnen til at vi regner om start-klokkeslettet til sekunder er at når brukeren velger å avslutte stadion oppholdet med å trykke på knappen "Forlat Området" henter vi klokkeslettet igjen og regner dette også om til sekunder. På denne måten vi trekker start-klokkeslett ifra slutt-klokkeslett og dermed finne ut hvor lenge han har vært på området, i sekunder.

Vi tenkte lenge på hvordan vi klare innfri det andre kravet, å sjekke om brukeren virkelig var på området. Løsningen vi kom opp med var å hele tiden sjekke hvor langt brukeren er fra utgangsposisjonen. Når brukeren starter "Støtt Grasrotlag" funksjonen henter vi brukeren posisjon via HTML5 sitt Geolocation-API. Deretter brukte vi Geolocation sin watchPostiton-funksjon som oppdaterer brukerens lengde- og breddegrad ettersom brukeren forflytter seg.

```
                    //Henter ny posisjon på brukeren dersom brukerens posisjon endres
navigator.geolocation.watchPosition(function(pos) {
                    //Legger avstanden mellom startposisjonen og nåværende posisjon inn i distance
    distance = calculateDistance(start_pos.coords.latitude, start_pos.coords.longitude, pos.coords.latitude,

});
```

Kodeeksempel 17 – nåværende posisjon med watchPosition

Ved bruk av funksjonen calculateDistance sjekker vi hvor langt brukeren er fra startposisjonen bruker vi en Javascript versjon av Haversine's formel(12), også kalt "The great circle calculation". Det denne funksjonen gjør er at den tar imot 2 sett med koordinater og regner ut hvor langt det er i mellom dem, i luftlinje.

```
//Denne koden er hentet fra
//http://www.html5rocks.com/en/tutorials/geolocation/trip_meter/
function calculateDistance(lat1, lon1, lat2, lon2) {
    var R = 6371; // km
    var dLat = (lat2 - lat1).toRad();
    var dLon = (lon2 - lon1).toRad();
    var a = Math.sin(dLat / 2) * Math.sin(dLat / 2) +
            Math.cos(lat1.toRad()) * Math.cos(lat2.toRad()) *
            Math.sin(dLon / 2) * Math.sin(dLon / 2);
    var c = 2 * Math.atan2(Math.sqrt(a), Math.sqrt(1 - a));
    var d = R * c;
    d=d*1000;           //Gjør om til meter
    d = Math.round(d); //Runder av tallet til nærmeste hele
    return d;
}
```

ss

Kodeeksempel 18 - Regner vi ut lengden mellom 2 punkter.

Nå vet vi altså hvor langt brukeren er fra startposisjonen og hvor lenge han har vært på området i sekunder. Det som gjenstår nå er sjekken som gjøres får å se om brukeren har oppfylt kravene idet han velger å forlate stadion. Når brukeren velger å forlate området gjennom knappen «Forlat området» henter vi som sagt klokkeslettet igjen og regner det om til sekunder, trekker fra start-klokkeslettet og finner hvor lenge han har vært på området i sekunder. Vi henter også distansen brukeren er fra startposisjonen. Med disse to variablene sjekker vi om brukeren oppfyller kravene.

```
                //sjekker at tid er over 1200 sekunder og at distansen er under 250m
if(samletRegnet >= 1200 && distance < 250){ //krav
    var min = Math.floor(samletRegnet / 60);    //regner sekunder til minutter
    var sec = samletRegnet - min * 60;          //rest til sekunder

  $.ajax({          //sender data til getstadioncoins.php
    type: 'POST',
    url: '../serverFiles/getstadioncoins.php',
    data: {'gcoins' : gc, 'time' : samletRegnet},
    success: function( data ) {
        if (data == "") {
              console.log("Sendt data til PHP-fil: ");
              console.log( data );
            alert("Dine Grasroitcoins har naa blitt lagret!\nOpptjente Grasrotcoins: " + gc + "\nTid på stadion brukt: " +
            min + " Minutter " + sec + " Sekunder");                //alert data dersom sending av data går problemfritt
        }
        else {
            console.log("error");                                   //coins ikke lagret
        alert("En feil har oppstatt, dine coins har ikke litt lagret");
        }
    }

  });
  }
else            //krav ikke infridd
  alert("Feil! Enten har du vaert paa stadion i mindre enn 20min\n eller du har beveget det for langt unna stadion-omraadet");
}
```

Kodeeksempel 19 – Sender grasrotcoins som er tjent opp, AJAX

Brukeren får altså tjent opp Grasrotcoins etter hvor lenge brukeren har vært på området eller stadion. Etter at brukeren har valgt hvem av Grasrotlagene/Undergruppen som skal få de opptjente Grasrotcoinsene kan brukeren velge å forlate området. Måten vi har definert hvor mange Grarotcoins brukeren får at når trykker «Forlat området» ganger vi antall sekunder brukeren har vært Støtt Grasrotlag med 2 og legger det inn i en variable med navn «gc». Denne variabelen sendes, dersom kravene innfris, til getstadioncoins.php som legger den inn databasen. Se kodeeksempel 19 over.

I «Støtt Grasrotlag»-funksjonen blir det også laget et kart som vises på skjermen. På dette kartet vises en «pin» med brukerens startposisjon og en farget sirkel som markerer 250meters radius rundt brukeres startposisjon. Når brukeren starter «Støtt Grasrotlag»-funksjonen hentes brukerens posisjon og kartet genereres Posisjonen hentes ved bruk av HTML's Geolocation API(5), mens kartet, «pin'en» og sirkelen er generert ved bruk av Google Maps API(11).

5.5.6 Gå til jobben

Norsk Tipping ønsket at applikasjonen skulle ha en funksjon som belønnet brukeren med Grasrotcoins ettersom hvor aktive de var. Den skulle belønne brukeren som, for eksempel, valgt å gå til jobben eller var ute og trimmet. Dette har vi løst med en sporingsfunksjon. Den funksjonen har vi også løst ved hjelp av HTML sitt Geolocation API.

«Støtt Grasrotlag»-funksjonen sjekker vi hele hvor langt du er fra startposisjonen som betyr at om du går fra A til B, også fra B til A igjen skal den, i teorien, vise at du har gått 0 meter. Noe som er svært upraktisk i denne sammenhengen. Det vi har gjort forskjellige i denne funksjonen i motsetning til «Støtt Grasrotlag» er at hver gang watchPosition oppdager at brukeren har forflyttet seg, så har vi en funksjon, appendPosition, som sjekker om det finnes tidligere koordinater lagret. Dersom det finnes tidligere lagrede koordinater hentes den sist registrerte posisjonen ut og ved bruk av calculateDistance, samme funksjon som regner ut avstanden mellom to posisjoner i «Støtt Grasrotlag», regnes distansen mellom nåværende posisjon og sist posisjon ut. Distansen plusses på i en variabel som innholder tidligere tilbakelagt distanse eller null, dersom brukeren akkurat har begynt turen.

```
            //Regner ut distanse fra forrige posisjon, hvis mulig
function appendPosition(position) {
            //Prøver å legge forrig posisjons koordinater inn i lastPos
    var lastPos = coords[coords.length-1];
            //Kjører if-setningen om "forrige koordinat" eksisterer
    if(lastPos) {
            //Regner ut lengden mellom forrige koordinat og nåværende koordinat
            // og legger den inn i distance
        distance += calculateDistance(lastPos, position.coords);
    }

            //Pusher de nye koordinatene inn i arrayen
    coords.push(position.coords);
}
```

Kodeeksempel 20 - regner ut avstanden mellom sist posisjon og nåværende posisjon.

På sammen måte som i «Støtt Grasrotlag» henter vi ut startklokkeslettet til brukeren, og gjør det om til sekunder, i det funksjonen startes. Når brukeren velger å avslutte turen ved bruk av «Stopp»-knappen henter vi klokkeslettet igjen og gjør det til sekunder og finner ut hvor lenge turen har vart. Vi henter også ut distansen brukeren har oppnådd. Ved bruk av vei, fart, tid teoremet, som gir at distanse delt på tid gir gjennomsnittsfart, finner vi brukerens gjennomsnittsfart på turen. I koden bruker vi brukerens gjennomsnittsfart til anta om brukeren gikk, løp eller syklet. Dersom brukeren ender opp med en gjennomsnittsfart på over 16 m/s(58km/t) antar funksjonen at brukeren har benyttet seg av motorisert fremkomstmiddel og gir beskjed om dette, det vil da ikke bli utdelt noen Grasrotcoins. Har brukeren en snittfart på 16 m/s eller lavere blir antall opptjente Grasrotcoins, turens distanse og den antatte fremkomstmetoden sendt, ved bruk av Ajax, til filen getgeocoins.php som ligger i mappen serverFiles, som lagrer dataen i databasen.

5.5.7 Etter-registering av tur

For å lett kunne etter-registrere en tur om man glemmer å bruke «Gå til jobben»-funksjonen har vi laget et enkelt avkrysningsskjema. Vi har laget avkrysningsskjemaet med HTML's radio buttons(13) som gjør at bruker kun kan krysse av ett alternativ ved hvert valg. Når brukeren trykker på «Registrer»-knappen sjekkes det at det er valgt et alternativ ved hvert valgene, distanse, fremkomstmetode og Grasrotlag/Undergruppe. Dersom brukeren har krysset av alle valgene beregnes det hvor mange Grasrotcoins turen var verdt. Deretter lagres dataene i databasen og det valgt Grasrotlaget/Undergruppen får Grasrotcoinsene.

5.5.8 Bingo

Før vi kunne løse programmeringen med bingoen, så var det noen ting vi ikke viste om bingo og retningslinjer om oppsettet. Se vedlegg 9.5, Regler for bingo brett, for regler angående

oppsett av bingobrett. Eksempler fra vedlegget er at på brettet så er tallene i 1 kolonne er imellom 1-15 og kolonne 2 er 16-30, etc. Dette skulle genereres på en elegant og effektiv kode, og helst så kort som mulig. Vi endte opp med følgende kode på kodeeksempel 21 under.

På kodeeksempelet så blir det først laget 2 arrayer. Den ene er for tall som er brukt, og den andre er en 2 dimensjonal array som vil reflektere cellene i bingobrettet. Dette var en endring vi gjorde underveis, da vi først prøvde å lage en tabell som tallene ble generert i. Men det viste seg at det ble vanskelig og hente ut celle informasjonen da vi ikke hadde system på cellene som vi får ved bruk av en 2 dimensjonal array. Det går også 2 for løkker i koden, den ene for rad og andre for kolonner. Den siste for løkken er kolonner og det er her det blir sjekket hvilken kolonne du er i med if/else og legges til i arrayen om tallet ikke finnes ifra før med php sin in_array() funksjon. Og i enden av for løkken så blir tallet lagt til i tabellen. Etter hele raden er ferdig så går den videre til neste rad.

```php
<?php
  //array med "brukte" nummere
  $numbers = array();

  //2 dim array
  $board = array (
     array(), array(), array(), array(), array(),
     );

  echo "<table border=1>";
  for ($row = 0; $row <  5; $row++) {         //for løkker for hele bingo boardet
     echo "<tr>";
     for ($col = 0; $col <  5; $col++) {

          if ($col == 0) {                    //første kolonne med tall fra 1-15
              $newValue = rand(1,15);                 //generer random nummer
              while (in_array($newValue, $numbers)) {     //til nummer er unikt
                  $newValue = rand(1,15);
              }
          }
          else if ($col == 1) {               //første kolonne med tall fra 16-30
              $newValue = rand(16,30);                //generer random nummer
              while (in_array($newValue, $numbers)) {     //til nummer er unikt
                  $newValue = rand(16,30);
              }
          }
          else if ($col == 2) {                   //"-"
              $newValue = rand(31,45);
              while (in_array($newValue, $numbers)) {
                  $newValue = rand(31,45);
              }
          }
          else if ($col == 3) {                   //"-"
              $newValue = rand(46,60);
              while (in_array($newValue, $numbers)) {
                  $newValue = rand(46,60);
              }
          }
          else if ($col == 4) {                   //"-"
              $newValue = rand(61,75);
              while (in_array($newValue, $numbers)) {
                  $newValue = rand(61,75);
              }
          }
          array_push($numbers, $newValue);        //add på array brukte tall
          array_push($board, $newValue);          //legger på 2 dim array

          $board[$row][$col]=$newValue;           //setter value i celle
          echo "<td class='cell'>" . $newValue . "</td>";     //display 2 dim array
     }
      echo "</tr>";
  }
  echo "</table>";
  ?>
```

Kodeeksempel 21 - Generere bingobrett

En annen utfordring vi hadde med bingoen var å markere tall som brukeren kunne trykke på og legge de inn i en array. Dette løste vi ved hjelp av jQuery, og koden ser du på kodeeksempel 22 under. Funksjonen i seg selv er ikke så omfattende og det er kommentert hva all koden gjør, så tenker ikke utdype dette noe mer utover dette.

```
<script>
    var cellValues = [];

    $(".cell").click(function() {       //ved klikk på celle
        $(this).toggleClass("green");       //toggle grønn og ikke

        var cellValue = $(this).text();     //henter tallet som blir trykket

        if ($(this).hasClass('green')) {    //if cellen er grønn
            cellValues.push(cellValue);         //add verdien til array cellValues
        } else {
            var index = cellValues.indexOf(cellValue);      //hvilken cellValue som er markert
            cellValues.splice(index, 1);                    //sletter tallet som er valgt, splice = slett
        }
    });
</script>
```

Kodeeksempel 22 - jQuery for å farge markerte celler

For å sjekke alle bingomulighetene til de forskjellige brukerne så satte vi to krav til funksjonen. Det første var at brukeren selv må være aktiv i applikasjonen og markere tallene som blir sendt ut. Det andre er at tallene som er markert faktisk ligger på en bingo rekke. Dette løste vi på følgende måte som du ser på eksempelet under. Det er laget fire arrayer, hvor de to første er de skrå og de to andre rader og kolonner.

Om du ser på den andre for løkken som initierer skraa2, så er det brukt en for løkke med flere variable. Dette gjør at vi kan gå ifra board[4][0] til board[0][4] i en enkel for-løkke.

For rader så blir det hentet rett ut ifra bingo brettet, og lagt til i arrayen. Mens i Kolonner så er koden litt mer vrien, men den tar alle kolonnene og ett tall ifra hver rad og legger kolonnen inn i columns arrayen.

```php
$skraa = array();
for ($i=0; $i<5; $i++) {
    array_push ($skraa, $board[$i][$i]);
}

$skraa2 = array();
for($j=0, $i=4; $j<5; $j++, $i--) {
    array_push ($skraa2, $board[$i][$j]);
}

$rows = array();
$columns = array();

for ($i = 0; $i < 5; $i++) {
    $row = $board[$i];
    array_push($rows, $row);
}

for ($i = 0; $i < 5; $i++) {
    $column = array();
    for ($j = 0; $j < 5; $j++) {
        $row = $rows[$j];
        array_push($column, $row[$i]);
    }
    array_push($columns, $column);
}
```

Kodeeksempel 23 - Koden for mulige bingorekker

Etter at denne koden har kjørt så blir det kjørt Ajax med en array av de markerte tallene brukeren har som sender denne koden til en sammenligningsfil vi har kalt compare.php som ligger på serverFiles mappen vår. Denne filen mottar alle de muligehetene i arrays, og sjekkergjennom alle tallene at alle tallene (alle 5) ligger i arrayen med de markerte tallene til brukeren som også blir sendt denne filen. Så returnerer den alle arrayene som den får treff på til bingo.php.

```php
<?php
    $row1 = $_POST['row1'];             // identifiserer alle arrays
    $row2 = $_POST['row2'];
    $row3 = $_POST['row3'];
    $row4 = $_POST['row4'];
    $row5 = $_POST['row5'];
    $colum1 = $_POST['colum1'];
    $colum2 = $_POST['colum2'];
    $colum3 = $_POST['colum3'];
    $colum4 = $_POST['colum4'];
    $colum5 = $_POST['colum5'];
    $skraa1 = $_POST['skraa1'];
    $skraa2 = $_POST['skraa2'];

    if ($markerte = isset($_POST["markerte"])) {    //identifiserer markerte i array
        $markerte = $_POST['markerte'];
    }
    /*
    else {                                          //mulig ikke skal være med?
        echo "du har ingen markerte";               //else feil meld om ingen markerte
    }*/

    $check = array($row1, $row2, $row3, $row4, $row5,
                $colum1, $colum2, $colum3, $colum4, $colum5,
                $skraa1, $skraa2);

    if (isset($_POST["markerte"])) {        //sjekker alle 12 bingo mulighetne for bingo
        for ($i=0; $i<12; $i++) {           //går gjennom alle 5 tallene ihver
            if (in_array($check[$i][0], $markerte) && in_array($check[$i][1], $markerte) &&
                in_array($check[$i][2], $markerte) && in_array($check[$i][3], $markerte) &&
                in_array($check[$i][4], $markerte)) {
                    echo print_r($check[$i]);
            }
        }
    }
```

Kodeeksempel 24 - sjekk om mulige bingorekker er markert

5.5.9 Min Profil

Som vist i design kapittelet så er profil siden en side hvor brukeren kan endre infoen til hans personlige profil. Dette blir gjort ved forskjellige SQL funksjoner, som oppdaterer profilen. Det blir også sjekket at feltene ikke er tomme. Det er også brukt funksjoner som diplayer all dataen til denne siden. Som for eksempel som value i inputtene hvor brukeren kan endre for eksempel navn, så står det gjeldende navnet som er lagret i databsen.

```
<tr>
    <td>Fornavn: </td>
    <td><input type="text" name="firstName" value="<?=$userInfo["firstName"]?>"/></td>
</tr>
```

Kodeeksempel 25 - Eksempel på å hente ut info fra database

Vi har valgt å bruke table taggen, og delt inn i rader og kolonner for å kunne ha alle inputtene på linje uavhengig hvor langt navnet før er. Her blir det kjørt en SQL som henter bruker info med userId, og vi velger å displaye fornavnet vist på eksempelet over.

```
<td>
<?
    $Orgs = getGrasrotOrg($userId);
    foreach ($Orgs as $Org) {
        $uLags = getUserUlag($Org['orgNr'], $userId);
        echo "<form method='post'>" .
        "<input type='submit' name='slettOrg' value='X'/>" .
        "<input type='hidden' name='orgNum' value='" . $Org['orgNr'] . "'/>" .
        "<input type='hidden' name='userId' value='" . $Org['userId'] . "'/>" .
        "<span value='" . $Org['orgNr'] . "'>" . $Org['navn'];
        if (!empty($Org['grasrotCoins'])) {
            echo " (" . $Org['grasrotCoins'] . ")</span></form>";
        } else { echo "</span></form>";}
        foreach ($uLags as $uLag) {
            echo "<li><form method='post'><input type='submit' name='slettUlag' value='X'/>" .
            "<input type='hidden' name='tkId' value='" . $uLag['tkId'] . "'/>" .
            "<input type='hidden' name='userId' value='" . $uLag['userId'] . "'/>"  .
            "<span value='" . $uLag['tkId'] . "'>" . $uLag['navn'];
            if (!empty($uLag['grasrotCoins'])) {
                echo " (" . $uLag['grasrotCoins'] . ")</span></form></li>";
            } else { echo "</span></form></li>";}
        }
    }
?>
```

Kodeeksempel 26 - Viser alle tilknytninger til bruker, samt slettknapp

En litt mer kompleks funksjon er å vise frem hvilke lag en bruker støtter. Her har vi kjørt en SQL som henter grasrotlagene en bruker har tilknytning med, og den andre funksjonen sjekker om hver av organisasjonene brukeren støtter om den samme brukeren også støtter noen av dets undergrupper. Det er blitt lagt til en fjern knapp som du kan slette de organisasjonene du følger, om du vil slette hele organisasjonen med alle undergrupper, eller bare en undergruppe er opp til brukeren. Det blir brukt input hidden for å holde kontroll på

hvilke tilknytninger som skal slettes. På kodeeksempel 27 under så ser du hvordan den ene SQL funksjonen som sjekker om brukeren også støtter en av underlagene til organisasjonen.

```
function getUserUlag($orgNr, $userId){
    global $dbConn;
    $sql = "SELECT t.userId, t.tkId, u.uId, u.orgNr, u.navn, t.grasrotCoins
            FROM tk as t
            JOIN ulag as u
            ON t.uId = u.uId
            WHERE u.orgNr = :orgNr
            AND userId = :userId";
    $stmt = $dbConn -> prepare($sql);
    $stmt -> execute(array(":orgNr"=>$orgNr,
                           ":userId"=>$userId));
    return $stmt->fetchAll();
}
```

Kodeeksempel 27 - Henter brukers undergrupper

Når brukeren skal legge til ett grasrotlag, så har brukeren muligheten til å søke etter grasrotlag. Vi ville ta i bruk en autofill funksjon for å løse dette problemet, og endte med å bruke en kodesnutt ifra Ali Aboussebaba (14), for original kode. Og vi endret denne og tilpasset den til vår applikasjon.

5.6 Tekniske memoer

Tekniske memoer er til hjelp for å avklare problemer med eventuelle løsninger av problemer som ikke ble løst, som var tenkt å blitt gjennomført. Det er som regel utformet på en kort og presis måte. Vi bruker har brukt teknisk memo for å illustrere vårt syn på ett gitt problem, og hva vi ville ha gjort for å løse dette problemet.

Funksjon:

"Bruker støtter ett Grasrotlag"

Beskrivelse av ønsket løsning / mangel:

En bruker skal ha mulighet til å kunne støtte de Grasrotlag og undergruppene brukeren måtte ønske. Dette gjøres fra profilsiden. Det er for øyeblikket mulig og legge til samme grasrotlag som du er tilknyttet ifra før, og dermed så oppstår det duplikat.

Løsning på problemet:

Kjøres en SQL spørring opp mot databasen om den nye tilknytningen allerede finnes ifra før, og har en if-setning som sjekker om det er treff, og gir brukeren bare mulighet til å registrere da funksjonen ikke får treff.

Alternativer:

Det finnes en eksisterende kode i løsningen som gjør nesten det samme på registrer. Da det blir sjekket opp om brukernavnet finnes ifra før. Se på koden og tilpass denne.

Hvorfor det bør gjøres:

For å ha orden på dataen i databasen og ikke få unødvendig data inn, som bare tar opp plass. Det gir også brukeren ett helhetsinntrykk på applikasjonen da feil ikke er syndelige.

Funksjon:

"Spille bingo"

Beskrivelse av ønsket løsning / mangel:

Brukerne av applikasjonen skal ha mulighet for å spille bingo. Her vil det være mulighet for å markere tallene som har blitt sendt til brukeren fra serveren vår. Disse tallene skal bli sendt likt til alle brukerne og det er kun serveren som har tilgang til hele arrayen. Det skal også være funksjonalitet som gjør at når en bruker får bingo, så skal alle andre brukere som spiller få beskjed om dette, og det skal spilles om to rekker, etc.

Løsning på problemet:

Det må implementeres web-sockets i applikasjonen slik at det blir en full-duplex kommunikasjonslinje mellom klient og server, og det må være en admin bruker som starter bingoen, om vi ikke setter den til å starte på ett tidspunkt. På serveren ville det vært en for løkke som legger tallene 1-75 (bingo tallene) i en array, og denne ville bli stokket, og serveren ville sendt ut ett og ett element til brukerne, samtidig som vi ville lagt

de trukne tallene i en egen array for sammenligning.

Da en bruker trykker bingo, så ville vi ha sjekket opp hvor mange arrayer som blir sendt, denne koden er laget, og gått gjennom alle arrayene og sjekket om det er 5/5 treff i dem, og brukt web-sockets til å sende beskjed til alle at en har bingo. Da denne beskjeden blir sendt ut, så ville vi hatt en variabel som telles opp med 1, og dermed vite hvor mange rekker det spilles om.

Alternativer:

Finne en annen løsning som gjør at brukeren kan bruke grasrotcoins på en sosial måte. Evenutelt sende requests med Ajax til serveren fra klient hvert 30 sekund, får å hente nytt bingotall fra den stokkende arrayen med tallene.

Hvorfor det bør gjøres:

Mye av applikasjonen er basert rundt bingoen og dette er noe som er svært sentralt i oppgaven. Det er kun mulig og tjene Grasrotcoins på applikasjonen, men ikke bruke noen på bingoen da den ikke er helt spill bar.

Funksjon:

"Støtt Grasrotlag & Gå til jobben"

Beskrivelse av ønsket løsning / mangel:

Støtt Grasrotlag & Gå til jobben er to funksjoner som begge tjener grasrotCoins til brukeren. Mye av funksjonaliteten til disse funksjonene er programmert i JavaScript. Det disse funksjonene har til felles er at de begge henter ut tid med javascript sin newDate() funksjon og regner ut ifra start tid og slutt tid hvor aktiv brukeren har vært på enten gå til jobb eller hvor lenge brukeren har vært Støtt Grasrotlag. Siden vi bruker JavaScript så skjer dette lokalt, og brukeren har mulighet til å endre klokken på enheten sin og dermed jukse seg til grasrotCoins.

Løsning på problemet:

Vi er ikke helt sikre på hvordan vi kan få løst dette problemet. Men vi ser for oss at det burde være mulig og hente klokken fra serveren, og bruke den i stedet for brukerens klokke.

Alternativer:

Endre språk til for eksempel PHP eller noe annet som gjør at koden ikke skjer lokalt, men på serversiden og dermed sikker for at brukeren ikke får lurt til seg tid på denne måten.

Hvorfor det bør gjøres:

Vi ser for oss at applikasjonen vil skape mye konkurranse innad i lag og bedrifter og dermed er det synd om folk kan jukse.

6 Testing og Kvalitetssikring

6.1 Blackbox & Whitebox Testing

Å teste koden vår, har vært viktig gjennom hele prosessen. For å teste at all funksjonalitet fungerer som den skal, så har vi benyttet oss av blackbox og whitebox testing. Dette er en programvare testing metode som er mye brukt i IT programmer. Med dette oppnådde vi en mye mer strukturert og omfattende test fase enn om vi bare hadde kjørt noen enkle tester selv.

6.2 Blackbox

Blackbox testing er den delen av testingen, hvor vi finner utvalgte personer og andre brukere til å teste vår applikasjon. De får ikke tilgang til koden, men de tester inputs og andre enheter av koden, hvor de forventer en input eller output og ser at alt stemmer. Hensikten med blackbox testing er å sjekke om all funksjonaliteten fungerer som den skal i applikasjonen.

Vårt hovedmål med blackbox testing var å teste all inputt til applikasjonen vår. Vi fikk testerne til å teste sider som innlogging, registrering og endring på profilsiden. Utenom dette så ble de også satt til å teste funksjonaliteten på gå til jobben og Støtt Grasrotlag.

Med blackboxs testing så oppnådde vi å rette på ett par feil vi hadde i koden, og det viste seg derfor å være nyttig for oss å ha eksterne testere, som ikke hadde kjennskap til koden. En av testerne fant ut at på funskjonen gå til jobb, så var det mulig og gå fra A til B og B til A, så bli distansen null, og dette strider imot hva som var ønsket funksjonalitet på gå til jobb. En annen funksjonalitet som ikke funket opptimalt var registrere ny bruker. Her var det mulig og registere med blanke felt, noe vi raskt fikk rettet opp i.

6.3 Whitebox

Whitebox testing er testing av kode gjort av oss, og er som regel gjort av utviklerne. Grunnen til dette er at med whitebox testing så skal testerene ha tilgang til koden, og

dermed inne smutthull ved å teste koden. Ved whitebox testing så blir det testet alt som skjer med koden og ikke bare outputten som brukeren før. Dette kan være ting som hva som blir skrevet til databasen til hva som skjer med en loop. Hovedårsaken med Whitebox testing er å se hvordan applikasjonen presterer.

Vi har kjørt tester mange lokale tester under utviklingen av dette prosjektet. For å teste variable som blir sendt rundt i filer, så har vi brukt alerts og annen output metoder for å kontroll sjekke at det alltid er riktig innhold i variablene. Ett eksempel på dette er bingoen, hvor vi alerter de rekkene som er markert og i rekke tilbake til bingobrettet. Her har vi også benyttet oss av javascript sin «console.log()»-funksjon. Etter at vi mener testingen er forholdsvis god nokk, så laster vi opp koden på serveren. (15)

```
<script>
      $('.save').click(  function(){
           //alert( );

          var rows = <?php echo json_encode($rows) ?>;
          var columns = <?php echo json_encode($columns) ?>;

          var skraa = <?php echo json_encode($skraa) ?>;
          var skraa2 = <?php echo json_encode($skraa2) ?>;
          //console.log(rows);
          //console.log(columns);

          $.ajax({
              type: "POST",
              url: "../serverFiles/compare.php",
              data: {markerte : cellValues,
                  skraa1 : skraa,
                  skraa2 : skraa2,
                  row1 : rows[0],
                  row2 : rows[1],
                  row3 : rows[2],
                  row4 : rows[3],
                  row5 : rows[4],
                  colum1 : columns[0],
                  colum2 : columns[1],
                  colum3 : columns[2],
                  colum4 : columns[3],
                  colum5 : columns[4] },
              success: function(data) {
                  alert(data);
```

Kodeeksempel 28 - Eksempel på bruk av Whitebox-testing

6.4 Tester utført av potensielle brukere

Skal applikasjonen kunne brukes av alle medlemmer og andre som ønsker å støtte et Grasrotlag, må applikasjonen kunne brukes av et bredt spekter av personer. Får å teste at applikasjonen treffer og tilfredsstiller målgruppen den er laget for, valgte vi å utføre testing av applikasjonen på potensielle brukere. Vi spurte derfor utvalget personer med ulik alder

og datakyndighet om de var villige til å teste applikasjonen. Vi ønsket at testpersonene skulle gi tilbakemelding på noen spesifikke punkter og hva de syntes om applikasjonen som en helhet. Under kan man se punktene vi ønsket svar på.

- Hva syns du om brukergrensesnittet?
- Hva liker du?
- Hva kan forbedres?
- Helhetsinntrykk?

Vi fikk mange gode tilbakemeldinger og vi fikk innspill på ting som kunne forbedres, men alle tilbakemeldinger er gode tilbakemeldinger. Ut ifra hvordan vi tolker resultatene av brukertestene er vi fornøyd med hvordan applikasjon treffer målgruppen den er laget for.

Oppsummering av brukertestene finnes i Vedlegg 9.4, Brukertester.

7 Avslutning

7.1 Avgjørelser underveis

Vi diskuterte lenge hvordan vi i gruppen hadde lyst til å løse oppdraget. Norsk Tipping ønsket en løsning som fungerte på Iphone. Native applikasjoner til IOS må utvikles på Mac ved bruk av kodeverktøyet Xcode. Da kun en av gruppemedlemmene hadde Mac og tilgang til Xcode falt valget ned på å lage en web-applikasjon. Ved å lage en web-applikasjon vil applikasjonen kunne brukes på IOS, windows-telefoner og android-telefoner.

Som nevnt tidligere i rapporten har vi laget applikasjonen ved hjelp av skytjenesten Bluemix. Grunnen til dette er hovedsakelig fordi oppdragsgiver ønsket dette. Tidlig i prosjektet, rett før selve utviklingen startet, var vi i møte på NT sitt hovedkontor i Hamar og fikk en kjapp innføring i Bluemix av representanter fra IBM. NT hadde da akkurat, i forkant av dette møtet, kjøpt lisens på Bluemix fordi de syns dette var ett spennende konsept som de mente ville være var en god mulighet for oss å lære noe nytt samtidig som vi fikk bruke et verktøy som var så nyskapende. Vi var veldig positive til Bluemix etter presentasjonen av IBM, så vi valgte dermed å benytte oss av verktøyet.

Siden vi hadde besluttet at vi ønsket å bruke Bluemix som plattform for utviklingen valgte vi å gjøre så mye som mulig gjennom Bluemix. Dette fordi vi ønsket å lære så mest mulig om hva Bluemix hadde og tilby og samtidig teste ut potensialet skytjenesten hadde. En annen grunn til at vi valgte å samle alt ett sted er at det gjør utviklingen mye mer oversiktlig. Det er på grunn av dette valget vi har brukt HubJazz, som er IBM's Git, som er implementer direkte i Bluemix.

Da vi utviklet en web-applikasjon testet vi hele tiden i en nettleser på datamaskinen for å se hvordan funksjonalitet så ut og fungerte. Siden applikasjonen har en del funksjoner som krever registrering, innfylling av felter, registrering av Grasrotlag/Undergrupper også videre, valgte å ha nettsiden stående. Slik at, vi leverer i hovedsak en applikasjon som installeres på smarttelefonen, men det er også en nettsiden tilgjengelig så brukere og Org administratorer kan logge seg inne med vanlig innloggingsinformasjon og bruke applikasjonen i nettleseren på datamaskinen.

Som valg av database så hadde vi mange valg, på grunn av tjenesten i Bluemix. Da vi valgte database, så var det ett par faktorer som var viktig. At det var en SQL database og at vi kunne bruke PDO tilkobling, da vi hadde kjennskap til dette fra før. Vi endte vi opp med en tredje parts tjeneste Bluemix ga oss, som het ClearDB. Ulempen med å bruke denne og ikke Bluemix sin egen SQL database var at størrelsen på databasen bare er 5mb, mens Bluemix sin er 100mb. I databasen skulle det også bli lastet opp alle Grasrotlag fra frivillighetsregisteret. Dette var en CSV-fil og denne viste seg å være på litt over 33000 rader. Vi fant da ut at databasen vi hadde valgt ikke var stor nokk med våre egne tabeller og den nye så vi gjorde en vurdering på om vi skulle bytte. Den nye databasen hadde ikke PDO tilkobling, og vi bestemte sammen med oppdragsgiver at siden dette bare var en prototype, så holdt det med å bare laste opp 100 rader, for å vise funksjonaliteten.

En stor del av dette prosjektet var det nye oppsettet av Grasrotlagene. Under Problemområde, avsnitt 1.1, så står det om hvordan Grasrot Andelen fungerer per dags dato. Vi fikk i oppgave av oppdragsgiver og videreutvikle denne. Det som var ønske fra arbeidsgiver var å gi brukerne mulighet for å kunne støtte undergrupper av sitt Grasrotlag. Dette vil for eksempel være g15 til HamKam. Dermed så vi på muligheter for å få til nettopp dette, men etter samtale med oppdragsgiver, så ble vi enige om at dette ikke var en optimal løsning med tanke på undergruppenes navn. Siden at ønsket funksjonalitet var at en bruker kunne støtte for eksempel sønnen og eller datteren. Så vi diskuterte hvordan dette kunne løses, og kom frem til at underlagene bruke ha navn som tilsvarer fødselsår og ikke alder.Ett eksempel på dette vil være at han nå har muligheten til å støtte HamKam 02, i stedet for HamKam G13. Med dette oppnådde vi at brukeren ikke trenger og endre organisasjonen og underlaget han vil støtte fra år til år da han alltid vil gå ett alderstrinn opp.

7.2 Mål og resultater

Arbeidet med dette applikasjonen har vært svært lærerikt. Det har vært veldig spennende å ta en lite skritt ut i arbeidslivet fått oppleve hvordan det utvikles i en reel jobbsituasjon. Vi har i løpet av prosjektet lært oss viktigheten med planlegging, strukturering og rutiner i forbindelse med både arbeid og kode.

Vi i prosjektgruppen har også lært oss nye språk, vi har lært å anvende nye bibliotek og APIer, vi har også lært håndtering av database opp mot vår egen kode. Språkene vi kunne fra før behersker vi nå enda bedre. På bakgrunn av at vi hadde jobbet lite med web-programmering før prosjektet, har det vært spennende å lære hva av koden som kjøres på klient, hva som kjøres på server. Hvordan språkene kommuniserer med hverandre og hvordan vi får serveren tilknyttet en database.

Vi er fornøyd med hvordan applikasjonen treffer målgruppen. Ved å bruke applikasjonen har brukeren muligheten til å både tjene oppe og lagre Grasrotcoins ved å være engasjert og aktiv innad i og for Grasrotlaget. Applikasjonen vi har laget kan brukes på Iphone, android- og windows plattform, dette gjør at applikasjonen er tilgjengelig for flere potensielle brukere, noe vi svært fornøyd med.

Vi mener at ved bruk av denne applikasjonen vil det skapes et større engasjement i Grasrotlagene. Applikasjonen vil gjennom sine funksjoner skape ett konkurransemiljø som kommer til å resultere i flere aktive meldlemmer i Grasrotlagene.

Spill-funksjonen "Bingoen" ble dessverre ikke ferdig. Mye av funksjonalitet rundt bingoen er ferdig kodet, men det gjenstår litt arbeid før spill-funksjonen kan brukes. Vi slet i lengre tid med litt av funksjonaliteten, og valgte å prioritere annen funksjonalitet fra produktkøen som oppdragsgiver mente var viktigere .

7.3 Kritikk til oppgaven

Når vi ser tilbake på oppgaven, så er det mange ting som kunne vært lagt bedre til rette. Det hele startet med at problemstillingen til oppgaven var upresis, og den endelige oppgaven ble ikke gitt før midten av Februar. Som en konsekvens av dette endte vi med en litt annerledes oppgave enn det vi først trodde var utgangspunktet, se Forprosjektet, se Vedlegg 9.10, Forrapport, uten at dette i seg selv er noe negativt.

Oppdragsgiver kom under hele prosessen med ideer og funksjonalitet som kunne vært nyttig i forbindelse med applikasjonen. Det gjorde det til tider var vanskelig å vite akkurat hva oppdragsgiver ville at vi skulle prioritere av funksjonalitet.

Vi tar på oss litt av skylden for at prioriteringene ikke ble optimale, og skulle muligens brukt tiden annerledes på starten av oppgaven. Den første måneden i prosessen så hadde vi alt for stort fokus på bingoen, og jobbet ganske ustrukturert med denne. Vi så for oss, at siden dette skulle bli Grasrotbingo, så var denne svært sentral. Det viste seg dog at det å tjene Grasrotcoins med ett logisk oppsett av en database var mye viktigere.

Vi er veldig takknemlige for at Norsk Tipping ga oss muligheten til å jobbe med Bluemix. Hvis vi skal trekke ned på noe, så må det være at Bluemix er svært nylig lansert. Det førte blant annet til at bruksanvisningen ikke fungerte som lovet første måneden av utviklingen. Vi hadde noen få ganger problemer med at serveren mistet kontakt med den integrerte Git'en som gjorde at serveren ikke kjørte. Dette hindret oss ikke merkverdig i løpet av prosessen da vi også kunne kode mye av funksjonaliteten lokalt.

7.4 Videre arbeid

Når du jobber på ett slikt prosjekt og applikasjon, så er produktet aldri ferdig. Det er alltid muligheter for å kunne utvikle og forbedre applikasjonen. Det er viktig og innse hva som ikke er godt utført og hva som er potensiell videreutvikling. Som nevnt i avsnitt 5.6, Tekniske Memoer, så er det ett par tekniske memoer og prosjektet ble ikke helt ferdig slik vi ønsket.

Dersom Norsk Tipping ønsker å videreutvikle prosjektet, anbefaler vi å starte med de tekniske memoene og å få på plass bingoen. Deretter ville vi lagt inn en utløser i "Støtt Grasrotlag"-funksjonen. Denne burde avbryte funksjonen med engang brukeren beveger seg utenfor en radius på 250meter fra startposisjonen. Dette er mulig å implementere da vi i koden hele tiden vet brukerens avstand fra startposisjonen.

Videre ville vi ha utvidet applikasjonen slik at den har funksjonalitet som kan vise Norsk Tipping hvem som har flest aktive brukere, og annen relevant statistikk. Databasen er designet til å kunne gjøre dette per dags dato, men det er ingen side eller funksjoner som tilbyr dette i vår applikasjon.

Vi ville også ha laget flere funksjoner knyttet opp mot Facebook, så brukeren ville få en mulighet til å poste på veggen sin hvor aktiv de har vært gjennom å bruke vår applikasjon. Dermed få mulighet til å vise vennene sine hvor langt hun/han har trimmet og eventuelt

hvor mange Grasrotcoins brukeren har tjent. Dette mener vi er en funksjon som ville hjulpet Norsk Tipping og Grasrotlagene til å få flere aktive brukere og som vil resultere i større engasjement.

Om det er ønskelig for Norsk Tipping, så er det også mulighet for å videreutvikle denne for wearable-enheter. På denne måten blir det enklere for brukeren og tjene Grasrotcoins, da det eneste enheten trenger for kjøre en fungerende versjon av applikasjonen er GPS og internett-tilkobling. Enheten trenger også touchskjerm for å kunne navigere rundt på applikasjonen.

7.5 Gruppe evaluering

7.5.1 Dynamikken i gruppen

De to gjenværende på gruppen har jobbet godt sammen og det har vært meget god kommunikasjon mellom gruppemedlemmene. Ved prosjektstart var vi tre, hvor det tredje gruppemedlemmet var ukjent for de to andre. Det ble vurdert under gruppedanningen at det ville vært spennende å få inn et friskt pust, da de to andre hadde jobbet sammen før.

Dette viste seg og være en dårlig avgjørelse, da gruppen bare er bestående av to medlemmer ved innlevering. Man merket fort at kommunikasjonen og initiativ ikke var som ønsket, noe som etter hvert resulterte i at medlemmet trakk seg. Det vi lærte av dette er at det er ikke så lurt å danne gruppe med personer som man ikke har kjennskap fra før eller vet hva er god for.

Gjennom hele prosessen så har satt av mye tid til oppgaven, og vi er svært fornøyd med arbeidsinnsatsen til hverandre. Vi har møtt på skolen hver dag, så langt det har latt seg gjøre og hatt flere lange dager med effektiv jobbing. Vi mener vi har prestert så godt som mulig med tanke på omstendighetene.

Det har i løpet av perioden vært noen heftige diskusjoner og konflikter angående prosjektet. Vi ser på det som veldig positivt at begge to har sterke meninger og ikke er redde for å si ifra dersom noen er uenige. Vi mener dette har resultert i at vi har drøftet mye som gjøre at løsningen har blitt enda bedre.

7.5.2 Organisering

I starten av prosjektet fordelte vi ansvaret i gruppen for å sette noen faste arbeidsoppgaver. På denne måten ville prosessen gå litt mer av seg selv, vi ville få bedre struktur i gruppen og det ville være lettere for gruppemedlemmene, oppdragsgiver og veileder å henvende seg til riktig person.

Lars har vært prosjektleder i løpet av prosessen. Det har vært hans ansvar å kommunisere med veileder og oppdragsgiver. Han har også hatt ansvar for å holde bloggen oppdatert og hatt hovedansvaret på JIRA.

John har hatt ansvar for dokumentasjon og loggskriving. John har også hatt hovedansvaret for databasen.

Vi syns at dette har vært en god måte å organisere gruppen på. Vi føler også at vi har fått mye igjen for å dele inn i faste administrative arbeidsoppgaver. Resten av administrative oppgaver har vi prøvde å dele så likt som mulig underveis.

7.5.3 Arbeidsfordeling

Under hele prosessen har vi vært flinke til å fordele oppgaver etter hvor vi har mest ferdigheter, får få en så god løsning som mulig. Det er viktig å presisere at oppgaver også er fordelte på tvers av dette slik at alle har fått prøve alt, da læring har vært en viktig faktor for oss i løpet av prosjektet. Vi har forsøkt og fordele arbeidsmengden så likt vi kan underveis. Det er lagt inn omtrent like mange arbeidstimer i prosjektet av de to gjenværende gruppemedlemmene.

I starten av prosjektet, så ble det også laget et gantt-diagram, se Vedlegg 9.10 , Forrapport. I dette gantt-diagrammet prøvde vi å planlegge hvordan prosessen ville se ut, tidsmessig. Ut ifra dette gjorde vi ett forsøk på å fordele alle ønskene av funksjonalitet vi hadde mottatt av Norsk Tipping i sprinter. Siden vi fikk en presis oppgavebeskrivelse såpass sent sprakk første sprinten og måtte gjøre noen endringer i forbindelse med videre planlegging. Vi hadde satt av litt over en uke i slutten av skjemaet til ting som kunne dukke opp. Dette benyttet vi oss av, og satt opp en sprint til, da vi allerede hadde satt av 3-4 uker til å skrive rapporten.

7.5.4 Prosjektet som arbeidsform

I utviklingen av denne applikasjonen har vi fulgt utviklingsmodellen Scrum, se avsnitt 1.6 Utviklingsmodell. Det har vært veldig spennende og lærerikt å bruke Scrum i reel jobbsituasjon. Vi har etter beste evne tatt på oss rollene som kreves og holdt møter som er en del av utviklingsmodellen. Det er ikke til å legge skjule å på det har vært utfordrende. Estimering er nok det vi sitter igjen å ser på som det vanskeligste med Scrum. Estimere størrelsen på oppgaver og hvor lang tid en oppgave vil ta. Vi har dog blitt veldig mye bedre på dette i løpet av perioden.

Vi mener erfaringen vi har fått med å jobben sammen i et slikt prosjekt er veldig verdifull. I arbeidslivet er prosjektarbeid en veldig vanlig arbeidsform. Vi har i løpet av våre 3 år på HiG jobbet i prosjektgrupper, men ingen av dem kan måle seg med størrelse, arbeidsmengden og læringsutbyttet som bacheloroppgaven har gitt oss.

7.6 Konklusjon

Da vi fikk oppdraget var vi utrolig spente, endelig skulle vi få en smakebit på hvordan det funker ute i den store verden. Vi hadde store forventninger til oss selv og mange tanker om hvordan vi trodde de neste månedene ville bli.

Nå er eventyret over for denne gang og vi er veldig fornøyd med både arbeidsinnsatsen og læringsutbyttet. Vi har måtte takle mange situasjoner som simulerer reelle utfordringer man møter i arbeidslivet. Vi har utvidet horisonten vår på så mange måter. Vi har lært å bruke nye språk, nye APIer, nye biblioteker og ting som virket ukjent og skummelt i starten har vist seg være overkommelige utfordringer. Vi har også både lært og erfart viktigheten av skikkelige planlegging og struktur, og gode rutiner.

Bacheloroppgaven har gitt oss muligheten til å bruke det vi har lært gjennom våre tre år på Høgskolen i Gjøvik, noe som har vært svært tilfredsstillende. Det er ikke til å legge skjul på at disse månedene har vært tøffe, vi har møtt mange hindringer underveis har som bare gjort oss sterkere og mer erfarne. Alt sett under ett er vi fornøyd med resultatet og på måten vi har løst problemstillingen. Vi har allerede fått en tilbakemelding fra Norsk Tipping som sier at de er interessert i å teste deler av funksjonaliteten internt, noe vi er svært stolte over.

8 Litteraturliste

8.1 Nettsider

(1) Om Norsk Tipping - Sist sjekket 14. Mai 2015
https://www.norsk-tipping.no/selskapet/om-norsk-tipping

(2) Ivar Farup - Sist sjekket 14. Mai 2015
http://www.ansatt.hig.no/ivarf/

(3) HiG, oppgaveskriving - Sist sjekket 14. Mai 2015
http://hig.no/student/oppgaveskriving

(4) Draw.io - Sist sjekket 14. Mai 2015
http://www.draw.io)

(5) HTML5, Geolocation - Sist sjekket 14. Mai 2015
http://www.w3schools.com/htmL/html5_geolocation.asp

(6) Prosjekt oppgave - Sist sjekket 14. Mai 2015
http://olemartin.com/projects/SUogSikkerhet.pdf

(7) Databaseteori - Sist sjekket 14. Mai 2015
http://www.jbi.hio.no/bibin/KoG1db/2012/forelesninger/databaseteori-3.html

(8) What is Bluemix - Sist sjekket 14. Mai 2015
http://www.ibm.com/developerworks/cloud/library/cl-bluemixfoundry/

(9) Phonegap, Build - Sist sjekket 14. Mai 2015
https://build.phonegap.com/app

(10) Four methods of adding CSS to HTML - Sist sjekket 14. Mai 2015
http://matthewjamestaylor.com/blog/adding-css-to-html-with-link-embed-inline-and-import

(11) Google Maps JavaScript API - Sist sjekket 14. Mai 2015
https://developers.google.com/maps/documentation/javascript/examples/map-geolocation

(12) Havershins formel - Sist sjekket 14. Mai 2015
http://en.wikipedia.org/wiki/Haversine_formula

(13) Radio Buttons - Sist sjekket 14. Mai 2015
http://www.echoecho.com/htmlforms10.htm

(14) Autocompelete using php/mysql and jQuery - sist sjekket 14. Mai 2015
http://www.bewebdeveloper.com/tutorial-about-autocomplete-using-php-mysql-and-jquery

(15) Difference between blackbox and whitebox testing - Sist sjekket 14. Mai 2015
http://www.softwaretestingclass.com/difference-between-black-box-testing-and-white-box-testing/

(16) Regler for bingo – Sist sjekket 14. Mai 2015
http://www.norsk-bingo.com/bingo_regler.html

8.2 Oppslagsverk

- T. Connolly & C. Begg, Database Systems: A Practical Approach to Design, Implementation, and Management 5th Edition, Addison Wesley, 2010
- Software Engineering, Ian Sommerville, 9th edition, 2010

9 VEDLEGG

9.1 Definisjoner

Havershine formel	Formel for utregning av lengde mellom koordinater
GPS	En måte å presis spore en enhets lokasjon
Telefontårn triangulering	Sporer enhets lokasjon ved hjelp av telefontårn
Cloud Foundry	Er en åpen kilde, sky-utviklingsplattform
PaaS(platform as a service)	En sky-utviklingsplattform
IBM	The International Business Machines Corporation
IBMs Open Cloud Architecture	IBMs skyarkitektur
API	Application Progamming Interface
Bibliotek	Importerte rammeverk
Data	Data på databasen
Android Studio	Rammeverk for utvikling på android plattform
Daily scrum meeting	Daglig Scrum møte
Frontend	På klient
Cross-Platform	Fungerer på fler på tvers av platformer
Backend	På server
Instaspill	Spill lanser av Norsk Tipping
Nabolaget	Spill lansert av Norsk Tipping
Fargepalett	Et spekter av farger
Rotmappe	Øverste mappe i filstrukturen

9.2 Logg

TID	HVA
Onsdag 11. Februar	Var i møte med veileder, fikk beskjed om å presse på for å få konkret oppgave
Fredag 13. Februar	snakket om ny oppgave på premiere fest, Nabolaget
Mandag-Onsdag 16.-18. Februar	flere mailer med arbeidsgiver angående mer spesifikk oppgave.
Onsdag 18. Februar	Innkalt til møte og bekrefte oppgaven. Drøftet hele oppgaven, hvor vi også fikk sett bluemix og IBM for første gang.
Torsdag 19. Februar	den store skidagen (fridag =D)
Fredag 20. Februar	Begynte å jobbe på tallgeneratoren
Helg 20.-22. Februar	startet å jobbe med grunnsteinen aka bingoen Gjorde basic kode her og grunn konseptet, leste på jquery
Mandag 23. Februar	satt seg inn i jquery for farging av felt Valgte å genere random tall på brett på server så det er litt mer sikkert, (om det hadde vært lokalt kunne det blitt endret av bruker lett. valgte 2 dim array og ikke tabell pga lettere og sjekke om noen treffer på bingo
Tirsdag 24. Februar	kommisjonær kurs hos norsk tipping
Onsdag 25. Februar	jobbet med design og andre småting, fikk alt lokalt over i bluemix, og satt oss inn i det. Møte med veileder og forklart oppgaven, hva vi har utviklet til nå og hva som er planen fremover. (veileder virker mer happy nå)
Torsdag-Søndag 26.-1. Februar	Kurs i Oslo med NITO (lokalledermøte)
Mandag 2. Mars	Satt opp arrays for alle vinner mulighetene i bingoen til hver spiller.
Tirsdag 3. Mars	dro til norsk tipping for sikkerhetsmøte men ingen stilte,

	ajax for å sende array fra script og php for å samenligne om noen er i rad/skrå/colonne og markerte. Sette seg inn i ajax.
Onsdag 4. Mars	Gjorde om koden til å se rekker så du fikk tak i dem utenfor for løkka og gjorde koden kortere og mer elegant.
Torsdag 5. Mars	fikk til ajax og samenligne om de som var markerte også var i rekker. Fikk problemer med å sende tilbake? Endret litt på css og fikk knappen til å funke bedre. Også fikk vi den til å fungere på mobile enheter.. Sort møte innad i gruppen om å legge om rutiner. Føler dårlig fremgang går for sakte
Fredag 6. Mars	Lastet ned sequel pro, og sett på dette Fikk satt opp database i bluemix og koblet den opp mot Sequel Pro for å få oversikt over databasen. Satt meg inn i det, og lager nå den første tabellen.
Helg 7.-8. Mars	Fikk satt opp eksempel på innlogging og registrerings sidene
Mandag-tirsdag 9.-10. Mars	Kontaktet også espen fra IBM om hjelp med sockets og prøver få det til å funke! Alle jobber nå med dette
Onsdag: 11. mars	veileder-møte og gjennomgikk prosjektet så langt. jobbet på grupperom etter på denne dagen.
Torsdag 12. mars	besøkte norsk tipping, jobbet videre med databasen. Sikkerhetskurs denne dagen.
Mandag 15. mars	jobbet hjemme siden det tok så lang tid med DB og testing da jeg måtte rs app for hver lille kode endring som ble gjort. brukte mye tid på dette da det ikke kunnes kodes lokalt.
Tirsdag 16. mars	fikk lagt ferdig registrering men fikk ikke addet til DB, men fikk til hente username og check på dette. Lagde grasrot-lag table og henter

	dette til registrering. Men feil med skrive til DB.
Onsdag 17. mars	fikk til skrive til DB. La også til at felt må være fylt ut før brukeren kommer videre.
Torsdag-fredag 18.-19. mars	Nytt database design, mer logisk oppsett. Hele databasen er redesignet
Mandag-Onsdag 23.-25. mars	Endret style på hele applikasjonen og alle filene til NT stil. hadde også fremføring på engelsk. satt oss inn i android studio. Fiks i jira.
Torsdag 26. mars	Møte med NT hvor vi presenterte det som har blitt gjort og planen fremover (facebook login, android, ny db etc) Fikk høre at vi måtte skifte DB siden den vi hadde valgt ikke hadde nok kapasitet til hele grasrotdelen. Må sette seg inn i dette. Fikk laget og gjennomgått en ny relasjonsdiagram for databasen og vi tok en beslutning på hvordan grasrotcoins skulle brukes. Den ble bestemt at det skulle gå like mye til laget og brukeren. Vi valgte å ha den i TK siden da får vi count på lag, og kunne ikke ha den i uLag siden da kunne vi ikke vite hvor den kom ifra. Det ble også bestemt at vi ikke skal bruke android studio da Jørn ville at NT kunne teste dette til høsten og 90% hadde apple telefon. Så vi ble enige og å bruke phonegap. I går hadde vi statusmøte med veileder på engelsk! Gikk bra, veileder er positiv. Vi mailet også med en Terje Krogstad og avtalte møte angående sockets
Fredag 27. mars	sette seg inn i bootstrap og implemtere dette.
Mandag 30.mars	Går bort fra grasrot lag på registrering, blir flyttet til profil siden.
Tirsdag-fredag 31.-3. april	Tar en titt på facebook og maps API Opprette meny fil og profil side. Fikset profilside med endring av info
Mandag-Fredag 6.-10. april	Fiksett facebook innlogging og skrive til databasen, ferdig med nytt design, og funksjon "Gå til jobben" er ferdig

	Add grasrotlag til profil og kunne slette tilknyttninger Autofill funksjon
Mandag-Fredag 13.-17. april	Støtt grasrotlag og rydding av kode. Ny tabell i databasen, om aktivitet. Legge til underlag av grasrot organisasjonene Skrive grasrotcoins til DB
Mandag 20. april	Startet med rapport skriving.

9.3 JIRA

Her er en komplett liste over hele produktkøen vår og hvordan vi jobbet med dem i sprintene. Det hele starter på sprint 3, siden vi ikke var så drevne med JIRA i starten og brukte litt tid på å sette oss inn i dette verktøyet. En annen merknad til denne figuren er at du ser her under "Assignee" hvem som har gjort de forskjellige oppgavene. Petter er nevnt i case 5 og case 12.

Global Software Developement

Displaying **42** issues at **13/mai/15 7:57 PM**.

Project	Key	Summary	Issue Type	Status	Priority	Assignee	Creator	Sprint
ntwear15	NTWEAR-49	legge til bootstrap	Task	Done	Major	John Christian G. Fjeld	John Christian G. Fjeld	Sprint 6
ntwear15	NTWEAR-48	Send data til getstadioncoins.php ved "stopp"(Støtt Grasrotlag)	Task	Done	Major	Lars Bendik Dølvik	Lars Bendik Dølvik	Sprint 6
ntwear15	NTWEAR-47	Generere kart + pin + radius sirkel(Støtt Grasrotlag)	Task	Done	Major	Lars Bendik Dølvik	Lars Bendik Dølvik	Sprint 6
ntwear15	NTWEAR-46	Lag distanse måler, tidsjekk+stoppknapp(Støtt Grasrolag)	Task	Done	Major	Lars Bendik Dølvik	Lars Bendik Dølvik	Sprint 6
ntwear15	NTWEAR-45	organisere filsystem	Task	Done	Major	John Christian G. Fjeld	John Christian G. Fjeld	Sprint 5, Sprint 6
ntwear15	NTWEAR-44	kode kommentering	Task	Done	Major	Lars Bendik Dølvik	John Christian G. Fjeld	
ntwear15	NTWEAR-42	på profile add så velge flere ulag og skrive alle til db	Task	Done	Major	John Christian G. Fjeld	John Christian G. Fjeld	Sprint 5, Sprint 6
ntwear15	NTWEAR-41	org side hover man kan lage ulag til kunn sin org	Task	Done	Major	John Christian G. Fjeld	John Christian G. Fjeld	Sprint 5, Sprint 6
ntwear15	NTWEAR-40	lag admin table	Task	Done	Major	John Christian G. Fjeld	John Christian G. Fjeld	Sprint 5, Sprint 6
ntwear15	NTWEAR-39	Fix duplicate of gOrg in DB on add	Task	To Do	Major	*Unassigned*	John Christian G. Fjeld	Sprint 5, Sprint 6
ntwear15	NTWEAR-38	add grasrotcoins til db	Task	Done	Major	John Christian G. Fjeld	Lars Bendik Dølvik	Sprint 5, Sprint 6
ntwear15	NTWEAR-37	start og stopp button	Task	Done	Major	Lars Bendik Dølvik	Lars Bendik Dølvik	Sprint 5, Sprint 6
ntwear15	NTWEAR-36	Vise posisjon/rute på map	Task	Done	Major	Lars Bendik Dølvik	Lars Bendik Dølvik	Sprint 5, Sprint 6
ntwear15	NTWEAR-35	Hente startlokasjon og tracke distanse tilbakelagt	Task	Done	Major	Lars Bendik Dølvik	Lars Bendik Dølvik	Sprint 5
ntwear15	NTWEAR-34	oppdatere profil	Task	Done	Major	John Christian G. Fjeld	John Christian G. Fjeld	Sprint 5
ntwear15	NTWEAR-30	Sette seg inn i phonegap	Task	Done	Major	Lars Bendik Dølvik	Lars Bendik Dølvik	Sprint 5
ntwear15	NTWEAR-29	Flytte DB	Task	To Do	Major	John Christian G. Fjeld	John Christian G. Fjeld	Sprint 5, Sprint 6
ntwear15	NTWEAR-28	Legg til Facebook innlogging og legge til i DB	Task	Done	Major	John Christian G. Fjeld	John Christian G. Fjeld	Sprint 5, Sprint 6
ntwear15	NTWEAR-27	Rydde opp i DB og lage ny struktur	Task	Done	Major	John Christian G. Fjeld	John Christian G. Fjeld	Sprint 4
ntwear15	NTWEAR-26	Legge til NT stil på siden	Task	Done	Major	John Christian G. Fjeld	John Christian G. Fjeld	Sprint 4
ntwear15	NTWEAR-25	Lage profilside(legge til grasrotlag)	Task	Done	Major	John Christian G. Fjeld	Lars Bendik Dølvik	Sprint 5
ntwear15	NTWEAR-24	En spiller kan registere fler grasrotmottakere	Task	Done	Major	John Christian G. Fjeld	John Christian G. Fjeld	Sprint 5, Sprint 6
ntwear15	NTWEAR-22	Oppdatere bloggen	Task	Done	Major	Lars Bendik Dølvik	John Christian G. Fjeld	Sprint 4
ntwear15	NTWEAR-21	Endre db-struktur	Task	Done	Major	John Christian G. Fjeld	John Christian G. Fjeld	Sprint 4
ntwear15	NTWEAR-20	Sette seg inn i node.red	Task	To Do	Major	Lars Bendik Dølvik	John Christian G. Fjeld	
ntwear15	NTWEAR-19	Importere grasrotlag til db	Task	Done	Major	John Christian G. Fjeld	John Christian G. Fjeld	Sprint 4
ntwear15	NTWEAR-18	JIRA-FIX	Task	To Do	Major	Lars Bendik Dølvik	Lars Bendik Dølvik	Sprint 4, Sprint 5
ntwear15	NTWEAR-17	connect til DB fra bluemix	Task	Done	Major	John Christian G. Fjeld	John Christian G. Fjeld	Sprint 4
ntwear15	NTWEAR-16	Compare som sjekker at markerte tall også står i bingo rekke, eller fler	Task	Done	Major	John Christian G. Fjeld	John Christian G. Fjeld	Sprint 4
ntwear15	NTWEAR-15	sette opp jquery og markere felt i bingoen, hvor de markerte legges i en array	Task	Done	Major	John Christian G. Fjeld	John Christian G. Fjeld	Sprint 4
ntwear15	NTWEAR-14	Relasjonsdiagram	Task	Done	Major	John Christian G. Fjeld	Lars Bendik Dølvik	Sprint 4, Sprint 5, Sprint 6
ntwear15	NTWEAR-12	sette opp database	Task	Done	Major	Petter Grøttheim	Lars Bendik Dølvik	Sprint 4
ntwear15	NTWEAR-11	registering	Task	Done	Major	John Christian G. Fjeld	Lars Bendik Dølvik	Sprint 4
ntwear15	NTWEAR-10	Innlogging	Task	Done	Major	John Christian G. Fjeld	Lars Bendik Dølvik	Sprint 4
ntwear15	NTWEAR-9	Må nå ha bingo på 2rader++(deles opp)?	Task	To Do	Major	*Unassigned*	Lars Bendik Dølvik	
ntwear15	NTWEAR-8	Bingomelding til alle	Task	To Do	Major	*Unassigned*	Lars Bendik Dølvik	Sprint 4, Sprint 5, Sprint 6
ntwear15	NTWEAR-7	Fikse sleep på tallgen(sprint 2)	Task	Done	Major	Lars Bendik Dølvik	Lars Bendik Dølvik	Sprint 3
ntwear15	NTWEAR-6	Send TallGen til client(sprint 2)	Task	To Do	Major	*Unassigned*	Lars Bendik Dølvik	Sprint 3, Sprint 4, Sprint 5, Sprint 6
ntwear15	NTWEAR-5	Opprette websocket-forbindelse mellom client og server(sprint 2)	Task	To Do	Major	Petter Grøttheim	Lars Bendik Dølvik	Sprint 3, Sprint 4, Sprint 5
ntwear15	NTWEAR-4	Compare mellom markerte tall og brett, finne ut om korrekt markering.(sprint 2)	Task	Done	Major	John Christian G. Fjeld	Lars Bendik Dølvik	Sprint 3
ntwear15	NTWEAR-3	compare mellom KORREKTEmarkerte tall og Trektetall(sprint 2)	Task	Done	Major	Lars Bendik Dølvik	Lars Bendik Dølvik	Sprint 3, Sprint 4
ntwear15	NTWEAR-2	Motor som generer random tall, displayer og adder i array	Task	Done	Major	Lars Bendik Dølvik	Lars Bendik Dølvik	
ntwear15	NTWEAR-1	Lage brett som genererer tall på riktig måte.	Task	Done	Major	John Christian G. Fjeld	Lars Bendik Dølvik	

Figur 21 - Fullstendig produktkø, med sprintinformasjon

9.4 Brukertester

Test nr:	1
Testers bakgrunn:	Testeren går første året på HiG sin linje, Økonomi og Ledelse og bruker datamaskin og smarttelefon daglig. Testeren er gjennomsnittlig datakyndig.
Oppsummering av testers tilbakemelding:	Brukergrensesnittet: Det er lett å navigere rundt i applikasjonen på grunn av det grafiske oppsettet. Menyen gjør at det er enkelt å se hvilke muligheter applikasjonen gir. Liker at applikasjonens design ser "flytende" ut, uten skarpe kanter. Hva likte du best: Syns det er kult at det går å logge inn ved bruk av Facebook. Liker «Gå til jobben»-funksjonen og prinsippet rundt, at man holder seg aktiv samtidig som det gir en fortjeneste. Hva kan forbedres: Syns lastingen av noen sider kan være litt treig. Litt vanskelig å trykke på Grasrotlag som er søkt opp. Helhetsinntrykk: Applikasjonen virker solid. Liker det enkle designet å mulighetene den tilbyr. Et godt eksempel på at det enkle ofte er det beste.
Test utført av:	Lars Bendik Dølvik, 2. Mai 2015

Test nr:	2
Testers bakgrunn:	Testeren er en eldre person som ikke bruker datamaskin daglig. Testeren er under gjennomsnittlig datakyndig
Oppsummering av testers tilbakemelding:	Brukergrensesnittet: Alle knappene er store og lett å se. Kontrasten mellom tekst og bakgrunn gjøre også teksten lett å lese. Det er uproblematisk å bevege seg rundt i applikasjonen. Liker også fargebruken. Hva likte du best: Liker "Støtt Grasrotlag" og muligheten den medfører. Liker også at man kan støtte flere Grasrotlag/Undergrupper. Hva kan forbedres: Vanskelig å treffe riktig Grasrotlag når man har søkt opp. Må i så fall slette og prøve igjen. Føler kanskje "Støtt Grasrotlag" kunne hatt et annet navn, men usikker på hva. Helhetsinntrykk: Liker applikasjons muligheter. Syns det er flott at man kan bidra på sin måte gjennom å bruke applikasjonen. Fargebruken og balansen mellom de forskjellige tonene behagelig for øynene. Fargene og Norsk Tippings logo skaper en stemning som gir en følelse av sikkerhet.
Test utført av:	Lars Bendik Dølvik, 10. Mai 2015

Test nr:	3
Testers bakgrunn:	Andre års student ved Høgskolen i Gjøvik Programvareutvikling
Oppsummering av testers tilbakemelding:	Brukergrensesnittet: Appen er lett og overskikkelig. Kanskje litt kjedelig og lite på sidene, ellers er alt topp. Hva likte du best: Mange kule funksjoner og likte spesielt godt at man slapp og registrere bruker og bruke tid på dette, da Facebook-innloggingen var tilgjengelig. Hva kan forbedres: Fiks bingoen Helhetsinntrykk: Syns vi har gjort en god jobb, og liker at dere har laget den for mobil. Så slapp man å ta testen på en laptop!
Test utført av:	John Christian Fjeld, 28. April 2015

Test nr:	4
Testers bakgrunn:	Eldre dame. Gjennomsnittlig datakyndig, bruker data daglig og eier smarttelefon.
Oppsummering av testers tilbakemelding:	Brukergrensesnittet: Bra jobbet på siden, mange muligheter på siden og smart oppsett etter litt hjelp om funksjonene. Hva likte du best: Likte veldig godt de forskjellige metodene og tjene coins på. Hva kan forbedres: Ingen hjelp funksjon om det var ord eller annen funksjonalitet som brukeren lurte på Helhetsinntrykk: Det er mye bra vi har gjort, og liker at det er mulig og tjene Grasrotcoins for Grasrotlaget med å være aktiv. Dette kommer godt med da testeren er aktiv med trening.
Test utført av:	John Christian Fjeld, 25. April 2015

9.5 Regler for bingo brett

Regler for bingo(16)

Regler i bingo

- Her finner du en innføring i bingo. Reglene er veldig enkle som du ser.
I bingorommene vil du oppleve mange ulike spill på bingobrettet.

Bingo har blitt et populært spill blandt folk i alle aldre. Nå kan du også nemlig spille bingo utenfor de tradisjonelle bingohallene, du kan spille online hjemme ! Her finner du en liten oversikt over regler og noen få strategiske tips når du skal spille bingo.

Bingo Regler

Det er **75 mulige** bingo nummer. Det trekkes et og et nummer og du krysser av for de nummerene du har som blir trukket ut. Disse blir automatisk krysset av når du spiller online. En spiller har bingo når den matrisen det spilles etter er full. Det kan være en rad, en kollonene, diagonaler eller helt andre mønstre.

Spillerens vinnersjansen avhenger av hvor mange brett han har i spillet og hvor mange brett som er med.

Brettene

Hver spiller har så mange bingo brett som han eller hun har kjøpt. Hvert bingobrett har 5 rader og 5 kolonner, tilsammen 25 ruter. Kolonnene er navngitt fra venstre mot høyre med bokstavene **'B', 'I', 'N', 'G', 'O'**.

Med et unntak (den miderste ruta er free) er nummerene fordelt slik :

- hver rute i **'B'** kolonnen inneholder nummer fra **1-15** ;
- hver rute i **'I'** kolonnen inneholder nummer fra **16-30** ;
- hver rute i **'N'** kolonnen inneholder nummer fra **31-45** ;
- hver rute i **'G'** kolonnen inneholder nummer fra **46-60** ;
- hver rute i **'O'** kolonnen inneholder nummer fra **61-75**.

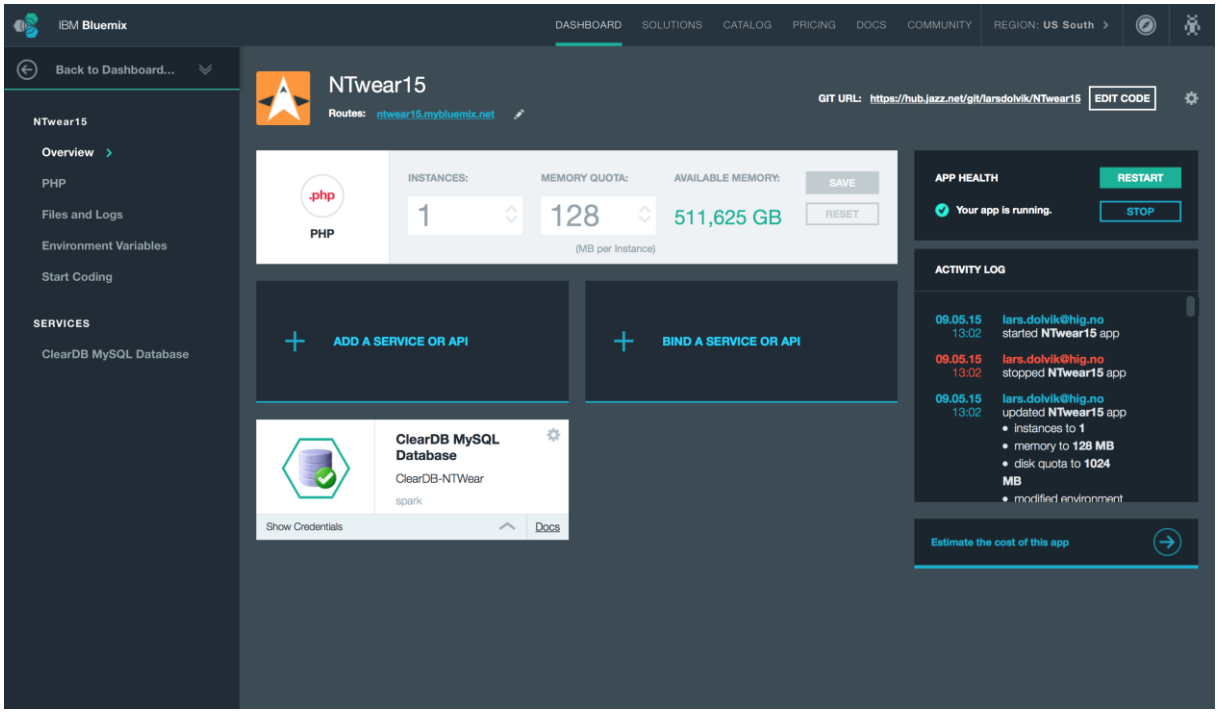
Et nummer kan bare forekomme en gang på hvert brett.

Bingo Strategi

Når du spiller online er det 75 mulige bingo nummer og i alle trekninger trekkes det til det blir en vinner. Det er ikke så mye du kan gjøre for å øke dine vinnersjanser enn å spille med så mange brett du kan. Bare ikke spill med så mange brett at du mister oversikten. Bingo er et sjansespill, innse at du ikke kan endre dette. Bare spill og ha det gøy og håp du blir den første med **BINGO** !

Figur 22 - Regler for bingo

9.6 Introduksjon i Bluemix



Figur 23 - Bluemix dashboard

Bluemix er som nevnt i utviklingsverktøy en skytjeneste hvor det er ett hav av funksjoner som gjør verden til utviklere mye enklere. På bildet under så ser du ett eksempel på hvordan dashboardet til applikasjonen vår ser ut. På navigasjon baren til venstre så ser du tilgjengeligheten til filene og loggene Bluemix har laget da applikasjonen deploy og builder. Du har også muligheten til å kode direkte inn på Bluemix uten behov for andre kode utviklingsverktøy.

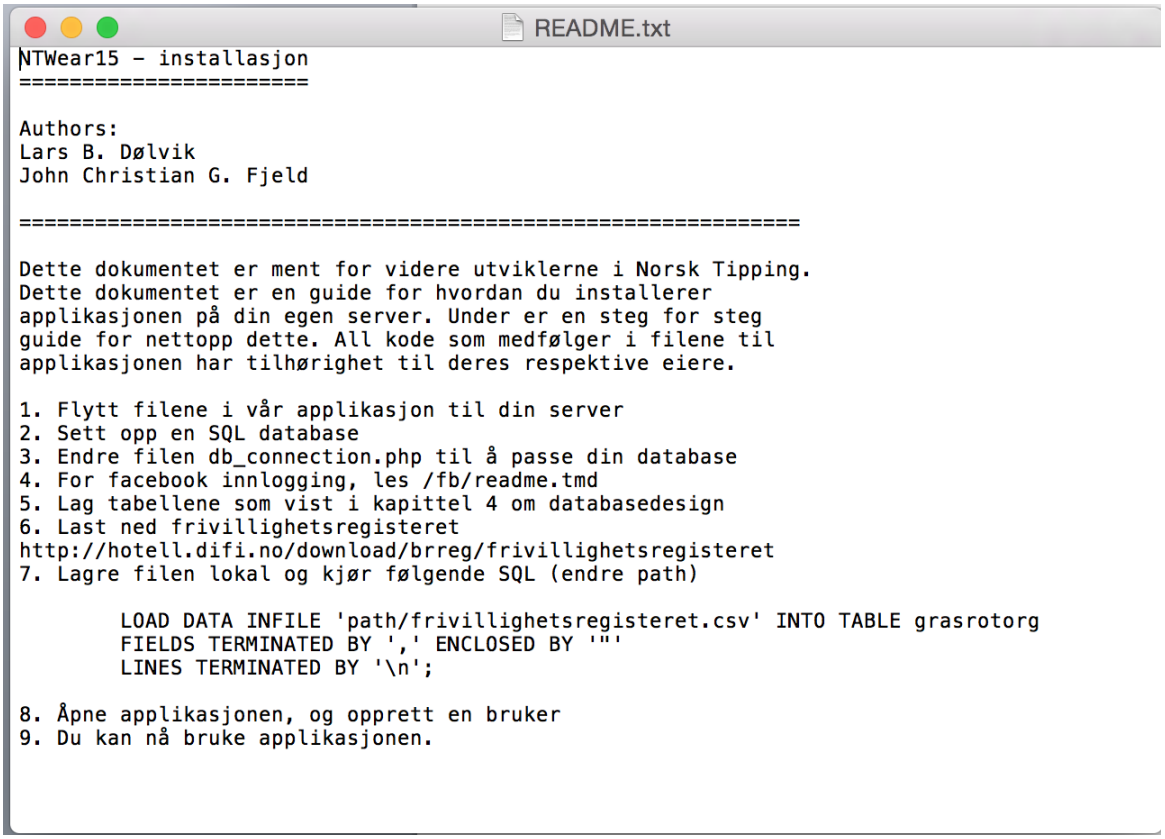
Midt på siden ser du en direkte link til websiden du får utdelt, og kan lett se hvordan applikasjonen ser ut på både web og mobil. Du kan også lett legge til flere instanser og memory de skal bruke. Som nevnt i utviklingsverktøy kapitelet om utviklingsverktøy så er det lett og legge til Bluemix sone APIer og andre funksjonaliteter. Under så ser du ett eksempel på dette da vi la til databasen vår.

Ser du oppe til høyre så ligger en DOCS tab, som er alle tutorials som ligger tilgjengelige på Bluemix.

På høyre siden så styrer du manuelt applikasjoenen og det vises en logg på aktiviteten til din applikasjon. Her vises alt ifra start/stopp og update av applikasjonen.

9.7 README.txt

Det er blitt laget en readme.txt i rotmappen til vår applikasjon som er for utviklerne i Norsk Tipping, så de kan jobbe videre med vår applikasjon. Denne filen inneholder en guide for hvordan vår applikasjon kan installeres på deres server.



Figur 24 - skjermdump av readme

9.8 Facebook

fbauth.php

Kobler opp mot applikasjonen din på developers.facebook.com og setter variable, krever autoload.php filen for å koble til. VI har endret innloggings informasjon for vår applikasjons sikkerhets skyld.

```php
<?php
ini_set('display_errors', '1');
session_start();
//FB

require_once 'autoload.php';

use Facebook\FacebookSession;
use Facebook\FacebookRedirectLoginHelper;
use Facebook\FacebookRequest;
use Facebook\FacebookResponse;
use Facebook\FacebookSDKException;
use Facebook\FacebookRequestException;
use Facebook\FacebookAuthorizationException;
use Facebook\GraphObject;
use Facebook\Entities\AccessToken;
use Facebook\HttpClients\FacebookCurlHttpClient;
use Facebook\HttpClients\FacebookHttpable;
use Facebook\GraphUser;

// init app with app id and secret
FacebookSession::setDefaultApplication('Din-app-id-her','Din-app-id-sitt-passord');

// login helper with redirect_uri
$helper = new FacebookRedirectLoginHelper('http://ntwear15.mybluemix.net/index.php' );

try {
  $session = $helper->getSessionFromRedirect();
} catch( FacebookRequestException $ex ) {
  // When Facebook returns an error
} catch( Exception $ex ) {
  // When validation fails or other local issues
}

// see if we have a session
if ( isset( $session ) ) {
    $user_profile = (new FacebookRequest(
      $session, 'GET', '/me'
    ))->execute()->getGraphObject(GraphUser::className());

     $_SESSION['fbId'] = $user_profile->getId();
     //$_SESSION['username'] = $user_profile->getName();
     $_SESSION['username'] = $user_profile->getProperty("email");
     $_SESSION['firstName'] = $user_profile->getFirstName();
     $_SESSION['lastName'] = $user_profile->getLastName();
     $_SESSION['name'] = $user_profile->getName();
     $_SESSION['email'] = $user_profile->getProperty("email");
} ?>
```

autoload.php

```php
if (version_compare(PHP_VERSION, '5.4.0', '<')) {
  throw new Exception('The Facebook SDK v4 requires PHP version 5.4 or higher.');
}

/**
 * Register the autoloader for the Facebook SDK classes.
 * Based off the official PSR-4 autoloader example found here:
 * https://github.com/php-fig/fig-standards/blob/master/accepted/PSR-4-autoloader-examples.md
 *
 * @param string $class The fully-qualified class name.
 * @return void
 */
spl_autoload_register(function ($class)
{
  // project-specific namespace prefix
  $prefix = 'Facebook\\';

  // base directory for the namespace prefix
  $base_dir = defined('FACEBOOK_SDK_V4_SRC_DIR') ? FACEBOOK_SDK_V4_SRC_DIR : __DIR__ . '/src/Facebook/';

  // does the class use the namespace prefix?
  $len = strlen($prefix);
  if (strncmp($prefix, $class, $len) !== 0) {
    // no, move to the next registered autoloader
    return;
  }

  // get the relative class name
  $relative_class = substr($class, $len);

  // replace the namespace prefix with the base directory, replace namespace
  // separators with directory separators in the relative class name, append
  // with .php
  $file = $base_dir . str_replace('\\', '/', $relative_class) . '.php';

  // if the file exists, require it
  if (file_exists($file)) {
    require $file;
  }
});
```

addFbUser.php

Funksjoner for å sjekke at bruker er I databasen.

```php
$dbConn = new PDO("mysql:host=".$host.";dbname=".$db, $username, $password);

//funskjoner
function getFbUser(){
    global $dbConn;
    $sql = "SELECT userId, email
            FROM users
            WHERE email = :email";
    $stmt = $dbConn -> prepare($sql);
    $stmt -> execute(array(":email"=>$_SESSION["email"]));
    return $stmt->fetch();
}

$fbUser = getFbUser();
if (isset ($fbUser['email'])) {
    //echo "treff på email";
    if (!isset($fbUser['fbId'])) {
        global $dbConn;
        $sql = "UPDATE users
                SET fbId = :fbId
                WHERE userId = :userId";
        $stmt = $dbConn -> prepare($sql);
        $stmt -> execute(array(":fbId"=>$_SESSION['fbId'],
                               ":userId"=>$fbUser['userId']));
    }

    $_SESSION['userId'] = $fbUser['userId'];
}

else {
    //echo "fant ikke email!";
    global $dbConn;
    $sql = "INSERT INTO users
            (username, firstName, lastName, email, fbId)
            VALUES
            (:username, :firstName, :lastName, :email, :fbId)";
    $stmt = $dbConn -> prepare($sql);
    $stmt -> execute(array(":username"=>$_SESSION['email'],
                           ":firstName"=>$_SESSION['firstName'],
                           ":lastName"=>$_SESSION['lastName'],
                           ":email"=>$_SESSION['email'],
                           ":fbId"=>$_SESSION['fbId']));

    $nyFbUser = getFbUser();
    $_SESSION['userId'] = $nyFbUser['userId'];
}
?>
```

9.9 Kontrakten

HØGSKOLEN I GJØVIK

PROSJEKTAVTALE

mellom Høgskolen i Gjøvik (HiG) (utdanningsinstitusjon),

Norsk Tipping (oppdragsgiver), og
Lars Dølvik,
John Christian G. Fjeld og
Petter Grotthelm (student(er))

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

1. Studenten(e) skal gjennomføre prosjektet i perioden fra 8. Januar til 15. Mai 2015

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

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

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

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

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

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

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

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

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

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

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

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

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

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

12. Deltakende personer ved prosjektgjennomføringen:

HiGs veileder (navn): Ivar Farup

Oppdragsgivers kontaktperson (navn): Jørn Berg Nordlund

Student(er) (signatur): Lars Dahle dato 27. Januar. 15

Vidar Fjeld dato 27. Januar. 15

Petter Gotthein dato 26. Jan. 15

_____ dato _____

Oppdragsgiver (signatur): J. Nordlund dato 27.01.15

IMT Dekan/prodekan (signatur): _____ dato _____

Versjon Januar 2013/hh

9.10 Forrapport

Prosjektplan

Forprosjekt av Bacheloroppgave
Av Lars Dølvik, John Christian G. Fjeld og Petter Grøttheim

Table of Contents

1. MÅL OG RAMMER

1.1. Bakgrunn

Norsk Tipping AS er ett aksjeselskap som er underlagt Kulturdepartementet.
Norsk Tipping er Norges eneste aktør innen drift av pengespill og har en viktig posisjon i å forhindre negativ spilladferd i samfunnet. Dette setter Norsk Tipping i en unik situasjon hvor størst mulig overskudd ikke er dets viktigste formål, men heller å føre pengespill i Norge i en forsvarlig sosialpolitisk retning, og sørge for at overskuddet fra pengespill skal komme resten av samfunnet til gode ved sponsing av lokale institusjoner som idrettslag og kulturskoler.

Norsk tipping er under stadig utvikling og hele tiden på utkikk etter andre måter og ekspandere seg på. I 2014 kom de med Instaspill og lanserte nylig Nabolaget.

Nå ser de for seg å prøve og ekspandere markedet enda mer og se hvilke muligheter det finnes innen wearable devices som sportsarmbånd, smartklokker og lignende apperater. Norsk Tipping har ikke noe forhold til disse enhetene og er på utkikk etter hvordan de kan bruke denne teknologien i deres spill og eventer.

Wearable devices som sportsarmbånd og smartklokkene til apple, android og windows er ganske nytt og blir bare større. Idag fungerer de fleste av disse smart klokkene som en mobil. Du kan se hvem som ringer, lese meldinger og mange av de kan du også gå på nett. Det finnes også andre funksjoner i forhold til trening og GPS funksjoner på hvor langt du har jogget med tidtagning og andre nyttige funksjoner. Smartklokkene er også designet slik at det er lett og få til ny programvare i form av app'er på den bare ved å koble den til en pc og laste ned, nesten som om de skulle lastet ned en app på telefonen din.

1.2. Prosjektmål

Etter dette prosjektet skal prosjektgruppen ha funnet ut:

- Utrede for mulighetene det er for bruk av wearables i spill- og eventbaserte løsninger i Norsk Tipping
- Utrede for hvilke muligheter vi har for å hente informasjonen fra sosiale medier og bruke det opp mot Norsk Tipping sin spillsammenheng.
- Utvikle en eller flere prototyper av spill som benytter seg av denne type informasjon i spillopplevelser
- Fullstendig bachelor-rapport med prosessen, funnene og resultatene

1.3. Rammer

Denne bacheloroppgaven er fordelt mellom 3 studenter, Lars Dølvik, John Christian G. Fjeld og Petter Grøttheim. Siden bacheloroppgaven IMT-3912 er ett 20 studiepoengs fag, så regner vi med at det vil være rundt 30 timer i uken pr. gruppemedlem i henhold til retningslinjene til HiG. Oppgaven skal leveres 15.mai.2015 så vi regner med å ha 19 uker arbeid. Dette vil da si at vi regner med å ha rundt 1700 timer på denne oppgaven. Dette inkluderer da forprosjektet, møter med oppdragsgiver og veileder, planlegging, rapportskriving og andre ting som kan komme opp under denne perioden ifra 08.jan.2015 - 15.mai.2015.

Vi er pliktige å følge skolen reglement og frister som er satt opp i forhold til faget IMT-3912. Dette vil være frister som leveringen av prosjektet 15.mai.2015, og øvrige innleveringer 22.mai.2015. Samt fremlegging av forprosjektet 28.jan.2015 og den muntlige presentasjonen i uke 23 hvor vi må være tilgjengelige.

- Prosjektgruppen har ingen økonomiske rammer
- Om prosjekgruppen trenger teknologiske ressurser vil det være Norsk Tippings jobb å dekke dette behovet
- Prosjektgruppen stiller ingen krav til arbeidslokaler

2. OMFANG

2.1. Fagområde

Prosjektgruppen vil være involvert i mange forskjellige grener innenfor fagområdet, Informasjonsteknologi. Når prosjektet er fullført skal prosjektgruppens medlemmer vært en del av:

- Forprosjekt: beskrivelse, planlegging og analyse av oppdraget.
- Gjennomføring: Følge plan, følge utviklingsmodell, utvikle produkt.
- Levere sluttprodukt: Ferdigstilling og overlevering.
- Presentasjon av resultatet.

2.2. Avgrensning

Prosjektgruppen har i samarbeid med Norsk Tipping funnet fram til en avgrensning av oppgaven.

- Norsk Tipping ønsker en sosial løsning som spiller kan delta i individuelt eller i grupper(spillag)
- Løsningen vil bli utviklet for smartklokker som benytter seg av android-wear som plattform
- Løsningen vil ikke direkte involvere pengespill

Norsk Tipping har i dag mange spillmuligheter, men ved spilling på wearable devices oppstår det en del problemer med tanke på gjenbruk av allerede eksisterende spill. De fleste av Norsk Tipping sin spill er pengespill, det vil si at penger blir satt inn og det blir gitt utbetaling ved gevinst. Det som gjør at pengespill byr på problemer i forbindelse med wearable devices er sikkerhet. Tanken med wearable devices er at det skal være tilgjengelig, interaktivt og enkelt å bruke, og inkludere bruk av sikkerhet som blant annet ID-brikke og/eller innlogging blir noe vi vil da se bortifra. En annen hindring er grensesnitt. Wearable devices som for eksempel en smartklokke har en liten skjerm som gjør at mange av de eksisterende spillene ikke egner seg på slike enheter.

2.3. Oppgavebeskrivelse

Det har i senere tid vært en stor utvikling av antall ulike enheter som inneholder sensorer som kan gi fra seg informasjon. Norsk Tipping er på utkikk etter nye måter å øke spillopplevelsen og skape nye spill basert på denne type informasjon.
Det blir vår oppgave å utvikle en løsning som gir mulighet for brukere av wearable devices til å delta i sosiale spill og events i regi av Norsk Tipping.

3. PROSJEKTORGANISERING

Vårt prosjekt består av en rekke roller som er viktig for opprettholde disiplin, effektivitet, veiledning og lande ett bra resultat etter oppdragsgivers krav. Under har vi satt opp en pyramide for å beskrive strukturen innad i prosjektet og i tillegg har vi beskrevet mer detaljert hva som innebærer i hver enkelt rolle.

3.1. Ansvarsforhold og roller

Figur: 3.1

Kommentar til figur: 3.1

På figur: 3.1 ser dere prosjektes rollefordeling. Vi har valgt og ha oss i prosjektgruppen på bunnen som tre utviklere, og prosjektleder/scrummaster som formidler av løsningene og prosessen fremover til norsk tipping. Hvor vi har valgt å ha veileder som ett mellom ledd dersom vi ønsker og/eller har behov for rådgivning.

Norsk Tipping/Produkteier(oppdragsgiver)
Kontaktperson: Jørn Berg Nordlund
Norsk Tipping er prosjektets oppdragsgiver. Det er de som ender opp med de kommersielle rettighetene til prosjektet. De har kommet til oss med ett behov som de ønsker at vi skal dekke. Under dette oppdraget er det Norsk Tippings jobb å uttrykke presist hvilket behov som skal dekkes og gjerne hvordan det skal dekkes i forhold til deres rettningslinjer. Deretter, under oppdraget, kommer de med innspill på løsnings-utkastene vi lager helt til de er fornøyd med resultatet.

Prosjektleder/Scrummaster: Lars Dølvik
Prosjektleders ansvar er først fremst å være talerøret mellom Norsk Tipping og prosjektgruppen. Når Norsk Tipping har kommet med sine ønsker og lagt fram hvilke behov skal dekkes må prosjektleder/scrummaster sørge for at utvilklinsteamet er underforstått med dette. Prosjektleder må har også ansvar for å lage et godt arbeidsmiløog ta hand om evenutelle hindringer for å gjøre utvilkingsprosessen så effektiv som mulig.

Utviklere: John Christian Fjeld, Petter Grøttheim, Lars Dølvik
I prosjektgruppen er det utviklerens ansvar å utarbeide en best mulig løsning av arbeidsoppgavene som er blitt gitt av prosjektleder. Dette skal gjøres innenfor Norsk Tipping sine gitte tidsfrister. Det er også utvikler sitt ansvar å raportere til prosjektleder om fremgangen og prosessen under utviklingen av løsningen.

Veileder: Ivar Farup
Under prosjektet er det veileders oppgave å rådgi oss, prosjektgruppen, under stortsett hele prosessen. Det er prosjektgruppens ansvar å søke råd og det veileders ansvar å svare etter beste evne. Dette prosjektet er et stort steg for oss i prosjektgruppen ut i noe ukjent og veileder skal være vår erfarne guide på reisen.

3.2. Rutiner og regler i gruppa

Se eget vedlegg, vedlegg A

4. PLANLEGGING, OPPFØLGING OG RAPPORTERING

4.1. Hovedinndeling av prosjektet

Vårt prosjekt er delt inn 2 deler.

Våres første del innholder utforskning, kravspesifikasjon og utvikling. I utforskningsfasen ønsker vi å fordype oss i informasjon og teknologi som er relevant for å løse oppgaven. Dette er kritisk å kunne før vi går videre til kravspesifikasjon og utviklingen. I Kravspesifikasjonsfasen vil vi bruke informasjonen vi tilegnet oss under utforskningsfasen for å lage en presis kravspesifikasjon. Den siste fasen er utvkling. I utviklingsfasen kommer vi til å benytte Scrum (argumentasjon og begrunnesle i underpunkt) som utviklingsmodell. Her vil vi bruke arbeidet som ble gjort i de 2 første fasene i samspill med utviklingsmodellen til å fullføre selve utviklingen.

Den andre delen vil bestå av dokumentasjon og rapportskriving. Dokumentasjonen vil inneholde all dokumentajson som er nøvending for å lage et rammeverk rundt prosjektet, som for eksempel kravspesfikasjonen. Rapporten vil inneholde hvordan prosessen går underveis, åssen oppdagelser vi gjør og hvordan resultater vi oppnår. Disse to delene vil foregår parallelt. Dette fordi vi ønsker en effektiv og nøyaktig dokumentasjon samtidig som vi ønsker en rapport som er omfattende og presis.

Valg av SU-modell/prosessrammeverk med argumentasjon

På bakgrunn av at prosjektet vil ansees å være litt utforskende med få forhåndsstilte krav vil en sekvensiell modell vil være upassende. De rigide kravene til slike modeller, e.g. fossefall, gjør at det må gjøres en kravspesifikasjon som vil være lite egnet for endringer som vi forventer i løpet av prosjektet. Dokumentasjonskravene til disse modellene er også for strenge og ville tatt for mye av tiden til den faktiske utviklingen.

Dokumentasjonskravene til Rational Unified Process er også en av grunnene til at denne heller ikke er valgt. På tross av sin noe smidigere prosess blir RUP for overspennende for et så lite team med for mange overflødige artefakter og roller.

Prosjektgruppen velger å ta i bruk Scrum som sin programvareutviklingsmodell. Hovedårsaken til valget av Scrum er at det gjør det lettere å gjøre eventuelle endringer som oppstår i løpet av prosjektet. Inndelingen av arbeidet i sprinter vil la gruppen ha hyppige møter med produkteier slik at prosjektet går i en retning som alle parter er enige i. Sprinter vil også gi gruppen fokus på å kunne ferdigstille kjernedeler av produktet først og ha “nice to have” funksjoner liggende hvis tiden skulle tillate.

Oppdraggivers egne ønsker er også tatt hensyn til i valget av Scrum. Norsk Tipping bruker selv til daglig Scrum som sin utviklingsmodell og dette gir verdifull erfaring som prosjektgruppen kan ta i bruk.

Andre smidige modeller som har vært oppe til diskusjon er eXtreme Programming. Som utviklingsmodell har den meritter som ikke kan ignoreres. Den oppmuntrer til innovasjon og en veldig åpen dialog mellom oppdragsgiver og utvikler, men som modell for en bachelorgruppe vil mangelen på dokumentasjon samt utelukkende bruk av par-programmering, som vil være for ressurskrevende for en så liten gruppe, være ugjennomførbart. Elementer av XP som supplement til Scrum der det er hensiktsmessig vil bli brukt. Par-programmering der det er nødvendig og brukerinteraksjon er eksempler på dette.

Se vedlegg C for hvordan vi planlegger at Scrum skal gjennomføres.

4.2. Plan for statusmøter og beslutningspunkter

Prosjektgruppen har sammen med sin veileder kommet frem til at det skal holdes statusmøter hver annen uke. Disse møtene vil være på onsdager kl 10.00 og lokasjon er veileders kontor.

På disse møtene skal vi diskutere hvordan de to foregående ukene har gått og hva som er planlagt for de neste to ukene. Vi vil møte opp med en grafisk plan i form av et gantt-diagram så veileder får et bedre innblikk i hva som er planlagt. Vi skal også diskutere hvordan det er gått med kommentarer som ble gitt på forrige statusmøte. På møte skal også milepæler diskuterers og eventuelle store valg som er tatt mellom møtene skal begrunnes kort.

Når prosjektgruppen skal ta besluttninger er det i hovedsak gruppeleder som har siste ordet, men det vil være en besluttningsprosess slik at alle gruppemedlemmer kan ytre sine meninger angående besluttningen som skal tas. Ved beslutningspunkter hvor det har vært uenighet rundt beslutningen skal alle meninger dokumenteres og det skal begrunnes hvorfor beslutningen ble som den ble.

Alle statusmøter med veileder og oppdragsgiver skal loggføres. Større avgjørelser tatt i løpet av prosjektet skal dokumenteres og begrunnes.

5. ORGANISERING AV KVALITETSSIKRING

5.1. Versjonskontroll og prosessverktøy

Som versjonskontroll har prosjektgruppen valgt å bruke Git med bitbucket som kode repository, og bruker JIRA Agile some prosessverktøy. Git er i dag en av de mest populære versjonskontrollsystemene som medlemmene i prosjektgruppen har erfaring med tidligere.

Som prosessverktøy har prosjektgruppen valgt å bruke JIRA Agile. bitbucket of JIRA er begge utviklet av Atlassian som tilbyr veldig gode funksjoner for sammenkjøring mellom de to produktene. I tillegg er også JIRA det systemet som Norsk Tipping bruker og som HiG har å tilby.

Alternativet som ble diskutert var Trello som medlemmene av prosjektgruppen har hatt erfaring med tidligere, og som er litt mindre og enklere enn JIRA, men gruppen ønsket å ofre enkeltheten for å ha muligheten til å utforske til det funksjonsrike verktøyet JIRA.

5.2. Risikoanalyse (identifisere, analysere, tiltak, oppfølging)

Teknologi, Forretningsmessig, Prosjektgruppemessig

Risikoanalyse:
For å kunne oppdage og avbøte på en uønsket hendelse som kan sette prosjektet i en uvelkommen situasjon vil det i bli utført en risikoanalyse. I første omgang for å identifisere mulige problemområder og de tiltak som vil bli gjort for, enten å forhindre at en slik situasjon skal oppstå, eller de handlinger som må benyttes for å begrense skadeomfang. I løpet av prosjektet vil disse handlingsplanene og retningslinjene bli fulgt, og oppdatert når nye potensielle problemer blir avdekket.

Det vil i denne analysen bli identifisert hva disse hendelsene kan være, og sannsynligheten for at denne hendelsen skal oppstå. Sammenhengen mellom konsekvensen av hendelsen og sannsynligheten av at den skal oppstå vil bli satt opp i en risikomatrise som vist i Skavland Idsø og Mejdell Jakobsen, Objekt- og informasjonssikkerhet, NTNU 2000, side 44

Beskrivelse av uønskede hendelser:

1. Gjentatte brudd på gruppens arbeidsfrister og møtetider av ett eller flere av gruppens medlemmer.
2. Tekniske problemer som fører til tap av data eller arbeid

3. Store endringer i oppgavens omfang som følge av uenigheter i prosjektets retning mellom oppdragsgiver og gruppemedlemmer
4. Feilberegning av tiden som er nødvendig for å kunne utføre et tilfredsstillende arbeid.
5. utilfredsstillende tilgjengelighet eller mangel på samarbeid fra oppdragsgiver.
6. sykdomstilfelle som setter ett eller flere medlemmer av prosjektet ute av stand til å ta del i arbeidet.
7. Frafall av gruppemedlemmer for resten av prosjektet.

Gradering av sannsynligheten

S-1 Svært sannsynlig: hendelsen inntreffer flere ganger i måneden
S-2 Meget sannsynlig: hendelsen inntreffer flere ganger i løpet av prosjektet
S-3 Sannsynlig: hendelsen vil inntreffe i løpet av prosjektet
S-4 Lite sannsynlig: hendelsen kan inntreffe i løpet av prosjektet
Gradering av konsekvensen
K-1 Katastrofalt: Hendelsen fører til konsekvenser som gjør at prosjektet må avsluttes
K-2 Kritisk: Hendelsen fører til konsekvenser som gjør at nøkkeldeler av prosjektet må innskrenkes
K-3 Farlig: Hendelsen fører til at det planlagte tidsforløpet får overskridelser
K-4 Lite farlig: Hendelsen kan medføre mindre endringer i tidsbudsjettet

Forklaring av risiko

1. S-2 K-3 Uventede omstendigheter og problemer kan fort oppstå som fører til at det er urimelig å forvente at alle arbeidskrav og møter blir fulgt. Dette er å forvente i alle prosjekt.
2. S-4 K-3 Det er alltid en mulighet for at det tekniske svikter, men de fleste tilbydere av tekniske løsninger er godt beskyttet for nedetid og har en tilfredsstillende tilgjengelighet.
3. S-3 K-3 I et slikt prosjekt som har mange mulige retninger man kan gå i er det risiko for forskjeller i visjon og ønske for hvor man skal ta prosjektet.
4. S-2 K-2 Hele prosjektgruppen har lite erfaring med prosjektplanlegging og uoverensstemmelser med tidsplanen vil forekomme i løpet av prosjektet.
5. S-4 K-3 Norsk Tipping har vist optimisme for prosjektet, men har også et annet prosjekt gående som kan føre til at de velger å prioritere det andre prosjektets
6. S-4 K-2 Sykdom kan alltid forekomme, og vil være opp til gudene.
7. S-4 K-2 Frafall av gruppemedlemmer ved at noen slutter eller blir tatt ut av gruppen er mulig, men gruppens medlemmer er motiverte for oppgavens

Risikomatrise

Konsekvens sannsynlighet	S-4	S-3	S-2	S-1
K-1				
K-2	7, 6		4	
K-3	5, 2	3	1	
K-4				

Risikomatrisen deler hendelsene inn i 3 kategorier;

- grønn er lav risiko
- gul er middels risiko
- rød er høy risiko

Hendelser som er blitt vurdert til lav risiko blir vurdert som akseptabel risiko, gul er risiko hvor tiltak og/eller handlingsplaner kan vurderes og rød er hendelser som krever tiltak og/eller en handlingsplan

Tiltak:

7 – For å motvirke enten at noen velger å hoppe av prosjektet, eller havner i en posisjon for å bli tatt ut av prosjektet må vi som gruppe være meget transparent vedrørende den motivasjonen vi sitter inne med, og om det er misnøye blant gruppemedlemmene

6 – Sykdom er ikke noe gruppen kan gjøre med, men for å kunne mitigere utfallet av en slik hendelse vil være å kontinuerlig ha tilstandsrapporter på arbeidet som er, har og skal gjøres slik at forstyrrelsen ved å overta arbeidsoppgaver blir minimert.

3 – For å motvirke at gruppens visjon for prosjektet ikke går imot oppdraggivers ønsker og forventninger vil vi ha regelmessige møter slik at slike utskeielser blir hurtig plukket opp og rettet på

1 – Her er også nøkkelen å ha åpenhet blant gruppemedlemmene slik at de kan «holde hverandre i ørene» og tidlig få rettet opp hvis det viser seg et mønster blant en eller flere av gruppemedlemmene

4 – For å unngå at tidsrammene blir for stramme vil gruppen være nøktern med fordelingen av tiden, samt bruke veileder eller andre støttespillere for å få innspill og veiledning for planleggingen av tid.

6. PLAN FOR GJENNOMFØRING

6.1 Gantt-diagram

Se vedlegg B

6.2 Kommentarer til Gantt-diagram:

Planleggingen eller forprosjektet er det blitt satt av 15 dager i henhold til de frister som er blitt gitt av skolen. Hoveddelen som det meste av arbeidet ligger, blir skrevet går over 77 dager. Her blir selve oppgaven løst og rapporten skrevet. Som vist i skjemaet over, så er utviklingen delt opp i 7 sprinter, hvor hver av disse går over 2 uker. Det er planlagt å ha statusmøter med veileder i hver av disse sprintene den siste onsdagen i sprinten. Det er også satt av 5 dager til gjennomgang av prosjektet på slutten, til å gå igjennom arbeidet og/eller om noe skulle dukke opp underveis. Da prosjektet er ferdig brukes den siste delen til å gjøre i stand en presentasjon for å presentere arbeidet som er blitt gjort. Våre milepæler er markert i gantt-diagrammet og markereret punkt i prosessen hvor delmål er oppnådd.

Vedlegg A

Grupperegler:

- Det kreves av alle gruppens medlemmer at det skal jobbes effektivt med prosjektet 25-30 timer pr. uke. Hvor arbeidstimer skal loggføres, og det meldes av til gruppeleder om unntak skulle forekomme.
- Da gruppen ikke opererer med faste arbeidstider må arbeidstid avtales på forhånd, senest dagen før planlagt arbeidsøkt.
- Alle gruppemedlemmer må vise initiativ og utføre arbeidet de har fått tildelt etter beste evne.
- Hvert gruppemedlem i prosjektgruppen har også fullt ansvar for å holde de tidsfrister som blir gitt, og si ifra senest 48 timer før fristen er ute til gruppeleder om problemer skulle oppstå. Dette gjelder alle frister som er satt av HiG, sprinter, arbeidsgivers frister og våres egne satte frister.
- Om uforusette kostnader oppstår som ikke oppdragsgiver ikke dekker og gruppa i enighet ser det hensiktsmessig å betale denne kostnaden, så deler alle i prosjektgruppen denne kostnaden likt. Kostnader som gjelder den enkelte person skal ikke dekkes av gruppa, men av personen selv.
- Det er prosjektsleder sitt ansvar å levere arbeidet som er blitt gjort på Fronter og inkludere alle i prosjektgruppen. Å formidle fremgangen i prosjektet til arbeidsgiver om dette er ønskelig og andre administrative beskjeder.
- Prosjektgruppen blir enige om hvilke verktøy som skal brukes for å løse oppgaven og holder seg til disse. Dette vil være skytjenester som google/drive, og andre skytjenester som dropbox. Det er gruppeleder sitt ansvar å sørge for at alle i prosjektgruppen har tilgang til disse tjenestene og fått invitasjon til verktøyet.

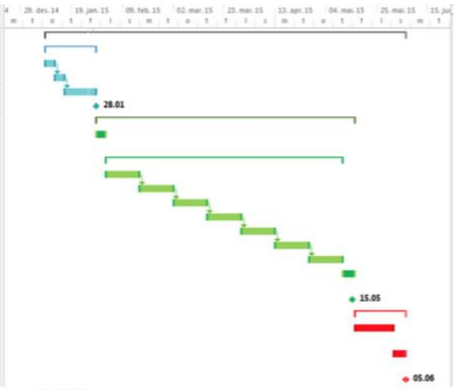
Ved brudd på grupperegler:

- Ved første brudd på en av reglene nevnt ovenfor så vil de bli gitt en muntlig advarsel til gruppemedlemmet dette gjelder. Det må da i enighet av de 2 andre gruppemedlemmene at denne skal bli gitt.
- Ved andre gangs brudd på reglene, så vil personen det gjelder få en skriftelig advarsel og veileder vil bli kontaktet om arbeidsforholdene i gruppen.
- Den tredje gangen en av de gitte reglene blir brutt, så vil veileder bli kontaktet og det vil bli vurdert om det er hensiktsmessig og ha vedkommende i dette prosjektet og kan risikere og bli kastet ut av gruppen.
- Ved alle advarsler som blir brutt skal de to andre medlemmene i enighet bli enige om det er hensiktsmessig å gi den gitte advarselen.

Lars Dølvik | Petter Grøttheim | John Christian Fjeld

Vedlegg B

Aktivitetsnavn	Varighet	Start	Slutt
Bachelor Oppgave	107 dager	to 08.01.15	fr 05.06.15
Forprosjekt	15 dager	to 08.01.15	on 28.01.15
Research	3 dager	to 08.01.15	sø 11.01.15
Planlegging	4 dager	ma 12.01.15	to 15.01.15
Utarbeide plan	9 dager	fr 16.01.15	on 28.01.15
Ferdig rapport	0 dager	on 28.01.15	on 28.01.15
Hoveddel	77 dager	to 29.01.15	fr 15.05.15
Forberedelse prosjekt start	3 dager	to 29.01.15	sø 01.02.15
Scrum	70 dager	ma 02.02.15	sø 10.05.15
Sprint 1	11 dager	ma 02.02.15	sø 15.02.15
Sprint 2	11 dager	ma 16.02.15	sø 01.03.15
Sprint 3	11 dager	ma 02.03.15	sø 15.03.15
Sprint 4	11 dager	ma 16.03.15	sø 29.03.15
Sprint 5	11 dager	ma 30.03.15	sø 12.04.15
Sprint 6	11 dager	ma 13.04.15	sø 26.04.15
Sprint 7	11 dager	ma 27.04.15	sø 10.05.15
Gjennomgang av prosjektet	5 dager	ma 11.05.15	fr 15.05.15
Innlevering	0 dager	fr 15.05.15	fr 15.05.15
Fremlering	15 dager	lø 16.05.15	fr 05.06.15
Forberede presentasjon	12 dager	lø 16.05.15	sø 31.05.15
Presentere prosjektet	5 dager	ma 01.06.15	fr 05.06.15
Ferdig Prosjekt	0 dager	fr 05.06.15	fr 05.06.15



Figur: 6.1

13

Vedlegg C

1. I starten av prosjektperioden vil det først bli satt opp en liste over alle gjøremålene som skal gjøres under prosjektet i en Product Backlog. Det vil bli delt inn sprinter og det gjøres plass til endringer og andre elementer vi muligens vil inkludere senere i prosessen.
2. Hver av sprintene går over en 2 ukers periode og i starten av hver sprint er det ett "sprint planning meeting" hvor det vil bli gjennomgått:
    a. Hvilke elementer vi vil trekke ut av backloggen og jobbe med den kommende sprinten
    b. Estimere hvor lang tid det vil ta og gjennomføre elementene og prioritere dem.
    c. Dele inn arbeidet i prosjektgruppen slik at det passer inn i den kommende sprinten.
    d. Start sprinten
3. I "daily scrum meeting" vil det bli gjennomgått:
    a. Statusrapport for hvordan hver enkelt ligger ann.
    b. Om noen av utviklerene har støtt på problemer.
    c. Evaluering av ferdige oppgaver i sprinten.
4. Sprint Review og Sprint Retrospective:
    a. Alle uferdige oppgaver legges tilbake til backlog.
    b. Vise det fullførte arbeidet som har blitt gjort til arbeidsgiver.
    c. Diskutere hva som gikk bra og dårlig i sprinten og hva som kan gjøres anderledes og/eller forbedres i neste sprint.