BACHELOROPPGAVE:

FACEBOOK-INTEGRERING AV SOSIALE SPILL

FORFATTERE:

Jan Fredrik Gundersen

Sondre T Johannessen

Dato:

15. mai 2015

SAMMENDRAG

Tittel:	Facebook-integrering av sosiale spill	Nr: | Dato :	15.05.15
Deltakere:	Jan Fredrik Gundersen		
	Sondre T. Johannessen		
Veileder:	Ivar Farup, ivar.farup@hig.no		
Oppdrags-giver:	Norsk Tipping AS ved Jørn Berg Nordlund		
Stikkord:	Webapplikasjon, systemutvikling, PHP, database, prediksjon		
Antall sider/ord: 110	Antall vedlegg: 9	Tilgjengelighet: Åpen	

Kort beskrivelse av bacheloroppgaven:

NTSosial er en webapplikasjon utviklet som en prototype med mål om å knytte sammen Facebook med Norsk Tipping sitt spillelagskonsept. Applikasjonen gir brukere mulighet til å være medlem av spillelag, foreslå kuponger til et spillelag og delta på kuponger foreslått av andre i et gitt spillelag. Brukere kan også få en oversikt over egen aktvitet.

Det er utviklet en databaseløsning som holder orden på alle brukere, deres spillelag, deres kuponger og om spillene er vunnet eller tapt.

Webapplikasjonen er utviklet ved hjelp av blant annet HTML, CSS, PHP, Javascript og MySQL. Prosjektet er gjennomført ved hjelp av systemutviklingsmodellen Scrum.

Rapporten er organisert i ni kapitler der vi starter med utgangspunktet vi hadde og jobber oss gjennom prossessen til resultatet vi har oppnådd.

ABSTRACT

Title:	Socially Integrated Gaming Experiences on Facebook	Nr: | Dato :	15.05.15
Participants:	Jan Fredrik Gundersen		
	Sondre T. Johannessen		
Supervisor:	Ivar Farup, ivar.farup@hig.no		
Employer:	Norsk Tipping AS, Jørn Berg Nordlund		
Keywords:	Web Application, Software Engineering, PHP, Database, Prediction		
Number of pages: 110	Number of appendix: 9	Availability: Open	

Short description of the bachelor thesis:

NTSosial is a web-application developed as a prototype with a goal of using Facebook with Norsk Tipping's concept of team gaming. The application gives users possibility to join a team, propose coupons to a team and to join coupons proposed by team members. Users can also see an overview over their own activity.

It is developed a database solution which keeps track of users, their teams, their coupons and if games are lost or won.

The application is developed using HTML, CSS, PHP, Javascript and MySQL. The project has been accomplished using Scrum, a system development methodology.

The report is divided into nine chapters from the starting point of the project, through the process of accomplishing the end product.

Forord

Denne rapporten tar for seg prosessen ved å utvikle en webapplikasjon integrert med Facebook, databaser og systemarkitektur.

Etter om lag fire måneder med planlegging og utvikling har vi en prototype som er stabil og fungerer. Det har vært veldig spennende å få lov til å være med og bidra med våre egne tanker og ideer rundt hva vi så for oss var relevant for konseptet, sammen med fastsatte krav fra oppdragsgiver. Vi håper vår prototype vil være til hjelp for Norsk Tipping når de setter i gang med videre arbeid av spillelagskonseptet som foreløpig er i startfasen.

Vi vil rette en stor takk til våre kontaktpersoner på Norsk Tipping, Håvard Kindem og Jørn Berg Nordlund for tilliten og god hjelp underveis.

Takk til alle som har bidratt med innspill og korrekturlest oppgaven:

Anders K. Veibust, Oda S. Osland, Erlend Vassbotten og Trine Skansberg

Innholdsfortegnelse

Figuroversikt

Kodesnutter

1. Innledning

1.1. Prosjektbeskrivelse

Facebook har over 1,3 milliarder aktive brukere, hvorav over tre millioner er nordmenn. Ca 80 prosent av trafikken foregår via mobil (1). Med et så stort segment for å nå ut til nye og eksisterende kunder er Facebook blitt en massiv plattform for bedrifter å uttrykke seg på. Gjennom blant annet sider eller applikasjoner. Å være synlig på sosiale medier hjelper både bedrifter og kundene ettersom det skaper tettere relasjoner. Når brukere allerede er inne på Facebook vil det være til stor fordel å kunne tilby løsninger gjennom denne plattformen.

Norsk Tipping er i konseptfasen med å utvikle en løsning for «spillelagsfunksjonalitet» på sine plattformer. Som bedrift jobber de kontinuerlig med å forbedre kundeopplevelsen gjennom gode spilltilbud og ved å møte kundene på deres premisser. Norsk Tipping er for øyeblikket ikke integrert på sosiale plattformer og det var et ønske fra de om å kunne være det, basert på å tilby kunder løsninger der de er.

Vi fikk i oppgave å utvikle en prototype for hvordan spillelagsfunksjonalitet kan integreres med Facebook for å gi større synlighet på sosiale medier. Norsk Tipping ga oss frie tøyler til å gjøre det vi mente ville være riktig for å integrere en slik funksjonalitet på Facebook. Det vi kunne forholde oss til var deres ide beskrevet i prosjektplanen (vedlegg F): «Spillelag er den digitale varianten av «kameratenes» tippelag og «venninnegjengens» lottoklubb.». Samt et ønske om tilgjengelighet for mobil. I første omgang var Facebook hovedmålet, men de ønsket også at det skulle være muligheter for å integrere det med andre sosiale medier.

1.2. Konseptbeskrivelse av Spillelag

Siden vi fikk frie tøyler til å utvikle et konsept for Spillelag var det viktig for oss å definere hva vi så for oss at et «Spillelag» var, for å kunne designe funksjonalitet for det.

Vi tok utgangspunkt i Norsk Tipping sitt ønske om å implementere «kameratenes» tippelag på Facebook. Et typisk tippelag går ut på at en gruppe har en felles pott for satsing av kuponger hvor det for eksempel rulleres på hvem som har ansvar for å designe ukens kupong. Gevinstmessig er det ingen forskjell på å gjøre det alene, men det er det sosiale

som er pådriveren her. Det er vanlig at eventuelle overskudd går til festligheter for gruppen, slik som fotballturer eller lignende.

For at folk skal ønske å bruke Spillelag på nett må det tilbys funksjonalitet utover bare det sosiale. Automatisk føring av statistikker og utdeling av poeng er eksempler som kan føre til økt spilleglede og interne konkurranser mellom brukerne. Alt det praktiske med leveringer av spill og kontroll på penger, kontoer og lag må håndteres av applikasjonen.

Ved registrering av poeng gis brukeren andre muligheter enn bare pengegevinster.Norsk Tipping kan for eksempel bruke det til konkurranser og det åpner muligheter for at brukere uten penger likevel kan være med på det sosiale og konkurrere internt. Det må i tillegg være muligheter for enkel kommunikasjon mellom medlemmer.

1.3. Avgrensing og forutsetninger

En slik konseptutvikling er omfattende og det var vår oppgave å avgrense den til det vi mente var realistisk å kunne utvikle innenfor tidsrammen. Det er derfor utviklet funksjonalitet med tanke på Langoddsen, et spill fra Norsk Tipping.

Norsk Tipping har som nevnt gitt oss frie tøyler og det er opp til oss å stille krav hos deres kjernefunksjonalitet for Spillelag. Vi har tatt utganspunkt i at Norsk Tipping har funksjonalitet som gir vår applikasjon tilbakemelding om en bruker har betalt og hvilket spillelag det ble betalt til.

1.4. Oppgavedefinisjon

Applikasjonen er utviklet som en ren responsiv webside. Her kan alle brukere i alle aldre logge inn via Facebook og opprette eller bli med i spillelag. Et spillelag består av en eller flere brukere og er et lukket miljø hvor medlemmene kan diskutere seg i mellom og levere inn kuponger.

Hver bruker har mulighet til å opprette en kupong basert på aktuelle kamper hentet fra Norsk Tipping. Etter at kampene er valgt får brukeren mulighet til å velge hvilket spillelag den gitte kupongen skal foreslås til, og om det er ønske om å delta ved å satse penger eller poeng. Det må velges hvor mye penger kupongen skal være verdt, og hvor mange det ønskes at skal være med på kupongen (andelseiere). Når disse kriteriene er innfridd får brukeren tilgang for å sende inn kupongen. Basert på valget mellom å bruke penger eller

poeng vil brukeren enten bli videresendt til Norsk Tipping for å legge av penger til kupongen eller bli trekket fra poeng. Summene som blir trekket fra er kupongens verdi delt på kupongens andelseiere.

Etter dette videresendes brukeren til spillelaget som ble valgt. Hvert spillelag har en oversikt over medlemmer, en ledertavle og et kommentarfelt med mulighet for diskusjon. Her er det en presentasjon over kuponger som er foreslått under fanen «foreslåtte kuponger». Hver kupong viser hvilken bruker som har foreslått den, informasjon om hvilke kamper, og hvor mange deltakere som gjenstår før kupongen kan sendes inn. Her får medlemmer i spillelaget mulighet til å delta på en kupong med samme kriterier, enten gjennom penger eller poeng. Det er to knapper på kupongen som enten trekker fra poeng eller videresender til Norsk Tipping for å godkjenne betaling.

Når en kupong oppfyller kravet til antall andelseiere blir den aktivert. Alle aktive kuponger kan følges i spillelagets «aktive kuponger». Her blir kampene oppdatert fortløpende og om en kamp ikke går inn går heller ikke kupongen inn og den blir avsluttet. Da havner kupongen inn under fanen «historikk» hvor alle avsluttede kuponger kan sees. Går alle kampene inn blir kupongen avsluttet og alle brukerne som er med i kupongen blir tildelt poeng basert på deres andel i kupongen. Brukerne som var med som betalende vil få gevinst på sin konto hos Norsk Tipping i tillegg.

Brukeren kan få en oversikt over alle sine spillelag, med en kort presentasjon over hvert spillelags antall medlemmer, antall foreslåtte og aktive kuponger. Ulike statistikker for brukeren blir presentert i form av antall kuponger i de ulike spillelagene, antall kuponger med gevinst og dato for plassering, samt seiersprosent i de ulike lagene. En oversikt over alle foreslåtte, aktive og avsluttede kuponger er gitt. I tillegg er det en oversikt over poeng i hvert spillelag og brukerens individuell poengsum.

1.5. Målgruppe

Applikasjonen: Norsk Tipping og deres kunder. Det er derfor viktig at prototypen fungerer sømløst og er brukervennlig slik at opplevelsen blir positiv for brukerne. Samtidig må den ha en back-end som kan være kompatibel eller overførbar for Norsk Tipping sine systemer.

Rapporten: Norsk Tipping og andre med IT-faglig bakgrunn. Rapporten er skrevet med forbehold om at den/de som leser den har kompetanse tilsvarende en dataingeniør. Norsk Tipping skal kunne bruke rapporten til å dokumentere valg i henhold til sin videre konseptutvikling av spillelag.

1.6. Mål

1.6.1. Effektmål

- Større synlighet for Norsk Tipping på sosiale medier.
- Økt kundedatabase.
- Nye samhandlingsmåter med spillere.

1.6.2. Resultatmål

- Merfunksjonalitet for Norsk Tipping sitt spillelagskonsept.
- En fungerende prototypeløsning for spillelag på Facebook.

1.6.3. Læringsmål

Vi kommer til å jobbe med prosjektstyringsverktøyet JIRA og versjonskontrollsystemet git. Førstnevnte verktøy har vi ingen erfaring med så der har vi mye å lære. Ingen av oss har arbeidet med så store prosjekter før og hatt en arbeidsgiver å forholde seg til når det gjelder utvikling. Dette fører til at vi må å lære og bruke en systemutviklingsmodell (se kapittel 1.8) i praksis. Andre læringsmål går ut på hvordan selve implementeringen av kode vil fungere og vi ser for oss at vi vil lære mye om APIer, databasedesign, serverspråk og webkoding.

1.7. Gruppens faglige bakgrunn og kompetanse

Vi studerer begge Dataingeniør ved Høgskolen i Gjøvik. Gjennom studieløpet her har vi fått innføring i emner som har tatt oss gjennom:

- Objektorientert programmering i C++.
- Algoritmiske metoder.

- Enkel PHP og MySQL databaser.
- Enkel webprogrammering og Javascript.
- Systemutviklingsmetoder.

Individuelt har Jan Fredrik bakgrunn fra egne prosjekter med HTML, JavaScript, Java, C++ og PHP. Sondre har bakgrunn fra egne prosjekter innen PHP, C++, Java, og Objective-C. Forståelsen for programmering innad i gruppen er over middels, men ingen av oss har erfaring fra et så stort prosjekt i PHP, HTML og JavaScript. Det ble erfart mye underveis og en bratt læringskurve i de respektive språkene, kodestruktur og jQuery.

Ingen av oss har arbeidet med systemutviklingsmetoder i praksis så dette var også helt nytt.

1.8. Arbeidsprosess

Grunnet boforhold og deltidsjobber har vi ikke hatt en fast timeplan for oppmøte og samarbeid. Vi har hele tiden hatt kontinuerlig dialog gjennom enten Skype eller Facebook. Det har vært jevnlige møter hos Norsk Tipping. Vi har også, når vi har sett behov for det, hatt workshop-dager på HiG for å synkronisere arbeidet og drive parprogrammering.

Norsk Tipping ønsket at vi skulle jobbet etter smidige metoder ettersom det er det de bruker. Vi gikk derfor for Scrum. Sprintene våre gikk over to uker med leveranse i hver sprint. På grunn av nevnte faktorer var det ikke alltid mulig med et daglige scrum-møter. Siden vi var to på gruppen var det ikke noe stort behov for det fordi vi hele tiden hadde dialog og var oppdatert på hvilke oppgaver hver av oss hadde å forholde seg til. I slutten av hver sprint hadde vi sprint review-møter med oppdragsgiver. Selv om vi var gitt ganske frie tøyler var det viktig å se til at vi var innenfor visse rammer av Norsk Tipping sin visjon.

Vi brukte JIRA til å legge til oppgaver i produktkøen, starte og avslutte sprinter, logge timer på oppgavene og for å få burndown-chart for å se om estimeringen vår var riktig. Her hadde vi også en ryddig oversikt over alle oppgavene, hvilke som måtte gjøres den aktuelle sprinten, hvem som var tildelt disse og om de var ventende, under arbeid eller ferdige.

I starten hadde vi en del møter med veilederen vår for å sikre oss om at vi angrep oppgaven på riktig måte. Vi førte en arbeidslogg hvor vi noterte oss problemer vi møtte på underveis for å kunne bruke det i rapporten når den skulle skrives. Vi har også skrevet en blogg og

logget møtereferat etter alle møter. Innad i gruppen har vi gitt frie tøyler til å implementere funksjonalitet vi ikke har planlagt, om en av oss har hatt gode ideer.

1.9. Øvrige roller

1.9.1. Oppdragsgiver: Norsk Tipping

Norsk Tipping er et statlig aksjeselskap underlagt Kulturdepartementet, som ble startet opp i 1948 og holder til på Hamar. Deres samfunnsoppdrag er å tilby et ansvarlig og attraktivt spilltilbud der overskuddet går tilbake til samfunnet. De har siden oppstarten bidratt med over 110 milliarder kroner til gode samfunnsformål (2).

Kontaktpersonene i Norsk Tipping er Jørn Berg Nordlund og Håvard Kindem. Vi har hatt flere møter underveis og de har vært til god hjelp under utformingen av oppgaven, både teknisk og når det kommer til konsept. Håvard har jobbet med konseptutvikling for Spillelagsfunksjonalitet i Norsk Tipping sine systemer og vi har brukt han for å få tilbakemeldinger på oppgaven. Jørn er Fagsjef for virksomhetsarkitektur hos Norsk Tipping. Han har vært en pådriver for ideer og funksjonalitet som vi kan implementere og gitt oss tilbakemelding på om vi er innenfor hans visjon også.

1.9.2. Veileder: Ivar Farup

Ivar er studieprogramansvarlig for Dataingeniør og Professor i Informatikk. Han har bidratt med tilbakemeldinger som har hjulpet oss å finne ut hvor mye vi skulle ta for oss i oppgaven.

1.10. Rapportorganisering

Rapporten er delt inn i sju kapitler med tilhørende underkapitler. De innledende sidene benytter romertall for sidetall, for å skille disse fra selve rapporten. Vi bruker en fastsatt skrifttype- og størrelse gjennom hele rapporten. Ord og uttrykk vi mener kan trenge forklaring finner man i terminologilisten (Vedlegg A).

Kapittel 1: Innledning

Gir en full oversikt over prosjektgrunnlaget. Man får vite litt om formålet og hva oppgaven går ut på.

Kapittel 2: Kravspesifikasjon

Her fastsetter vi kravene til løsningen. Disse vises hovedsakelig ved bruk av Use Case.

Kapittel 3: Design

Her viser vi frem skjermbilder av løsningen slik at man får et inntrykk av hvordan det ser ut før man leser videre.

Kapittel 4: Arkitektur

Her viser vi frem arkitekturen for løsningen vår.

Kapittel 5: Implementasjon

Hva slags verktøy vi benyttet oss av under utviklingen, hva vi har kodet og hvilke eksterne biblioteker/APIer vi benytter oss av.

Kapittel 6: Testing og kvalitetssikring

Her tar vi for oss testingen av løsningen, da med fokus på enhetstesting, brukertesting og kvalitetskontroll

Kapittel 7: Avslutning

Evaluering og refleksjon over arbeidet før konklusjon.

Kapittel 8: Litteraturliste

Inneholder kildene vi har brukt i rapporten.

Kapittel 9: Vedlegg

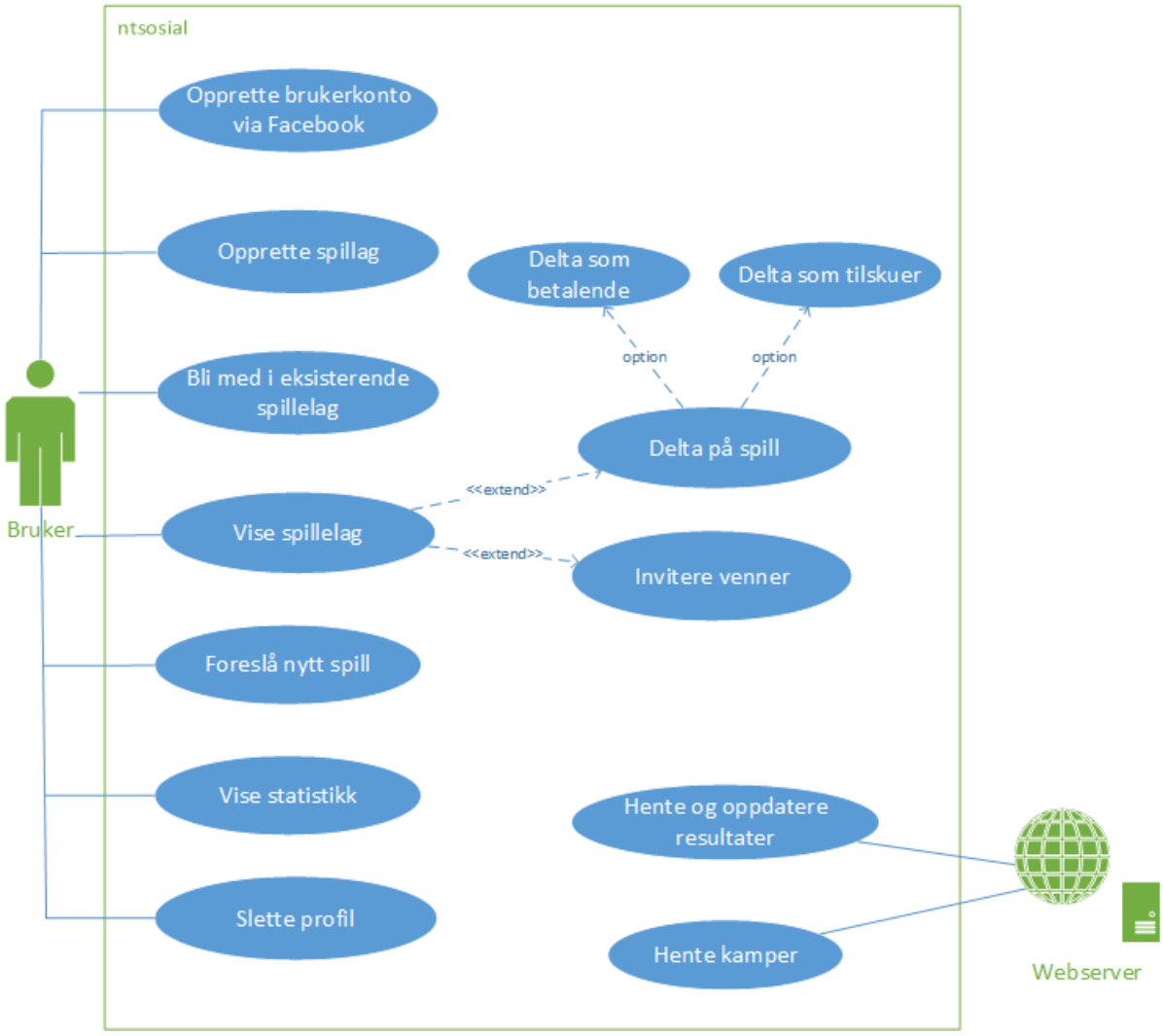
Terminologiliste, oppgavetekst, prosjektplan, statusrapporter, logg, møtereferater og kontrakt.

2. Kravspesifikasjon

2.1. Funksjonelle krav

2.1.1. Use case diagram

Vi har valgt å bruke Use Case til å formidle kravene for kjernefunksjonaliteten best mulig. Her har vi skissert overordnede og detaljerte use case, i tillegg til et sekvensdiagram for «Foreslå nytt spill» (Figur 2).



Figur 1 Use Case diagram

Figur 1 viser Use case diagrammet som er felles for mobilversjonen og fullversjonen siden vi har utviklet en responsiv applikasjon som tar hensyn til skjermstørrelsen. Dette medfører at man kan gjøre nøyaktig det samme om man sitter på en datamaskin, et nettbrett eller en mobiltelefon.

Webserveren skal hente kamper fra Norsk Tipping og oppdatere resultater for fullførte kamper.

2.1.2. Overordnet use case-beskrivelse

Use Case	Opprette brukerkonto via Facebook
Aktør	Bruker
Hensikt	Brukeren verifiserer applikasjonen i Facebook og blir registrert i vår database
Beskrivelse	Brukeren blir presentert med en velkomsttekst og får muligheten til å opprette konto med Facebook. Brukeren må godkjenne applikasjonen i Facebook, og når dette er gjort henter vi ut brukerens ID og navn og lager en oppføring.

Use Case	Opprette spillelag
Aktør	Bruker
Hensikt	Spillelag blir opprettet
Beskrivelse	Brukeren velger et navn på spillelaget. Spillelaget blir opprettet og brukeren blir med som administrator av spillelaget.

Use Case	Vise spillelag
Aktør	Bruker
Hensikt	Siden til et spillelag vises
Beskrivelse	Brukeren går inn på et spillelag og får opp oversikt over kuponger, hvem som eier spillelaget, opprettelsesdato, medlemmer av spillelaget, ledertavle og kommentarfelt.

Use Case	Invitere venner til spillelaget
Aktør	Bruker
Hensikt	Brukeren får venner til å bli med i spillelaget
Beskrivelse	Invitasjonsknappen genererer og viser invitasjonskoden og en «Facebook Send» knapp vises sammen med et felt for å sende SMS slik at brukeren kan dele koden med venner.

Use Case	Bli med i eksisterende spillelag
Aktør	Bruker
Hensikt	Brukeren blir med i et eksisterende spillelag

Beskrivelse	Brukeren taster inn invitasjonskoden og blir lagt til i spillelaget.

Use Case	**Foreslå nytt spill**
Aktør	Bruker
Hensikt	Brukeren sender et valgt spill til et av spillelagene
Beskrivelse	Norsk Tippings Langoddsen-spill vises. Bruker velger ut spill og sender til et spillelag.

Use Case	**Delta på spill**
Aktør	Bruker
Hensikt	Brukeren blir med på et foreslått spill
Beskrivelse	Foreslåtte spill vises brukeren. Brukeren velger å delta som betalende eller tilskuer.

Use Case	**Vise statistikk**
Aktør	Bruker
Hensikt	Viser stastikk til en bruker
Beskrivelse	Brukerens statistikk vises. Dette inkluderer seiersprosent, kuponger med gevinster i lag, og antall kuponger man har i de forskjellige lagene. Dette vises som diagram.

Use Case	**Slette profil**
Aktør	Bruker
Hensikt	Brukerens profil slettes
Beskrivelse	Detaljer registrert om brukeren slettes.

Use Case	**Hente kamper**
Aktør	Webserver
Hensikt	Fotballkampene tilgjenglig for spill hentes
Beskrivelse	Fotballkampene hentes og vises til brukeren slik at brukeren får velge kampene og spille.

Use Case	**Hente og oppdatere resultater**
Aktør	Webserver

Hensikt	Resultatene hentes fra Norsk Tipping og oppdateres
Beskrivelse	De siste kampene hentes og det sjekkes i databasen om noen har spilt på de fullførte kampene. Disse kampene blir da oppdatert.

2.1.3. Detaljert use case-beskrivelse

Vi har plukket ut noen av de mest kritiske delene av grunnfunksjonaliteten for å se nærmere på disse. Det er viktig at det er bra flyt på disse delene, og at feil som kan oppstå løses på en fornuftig måte. De trivielle beskrivelsene er flyttet til Vedlegg B.

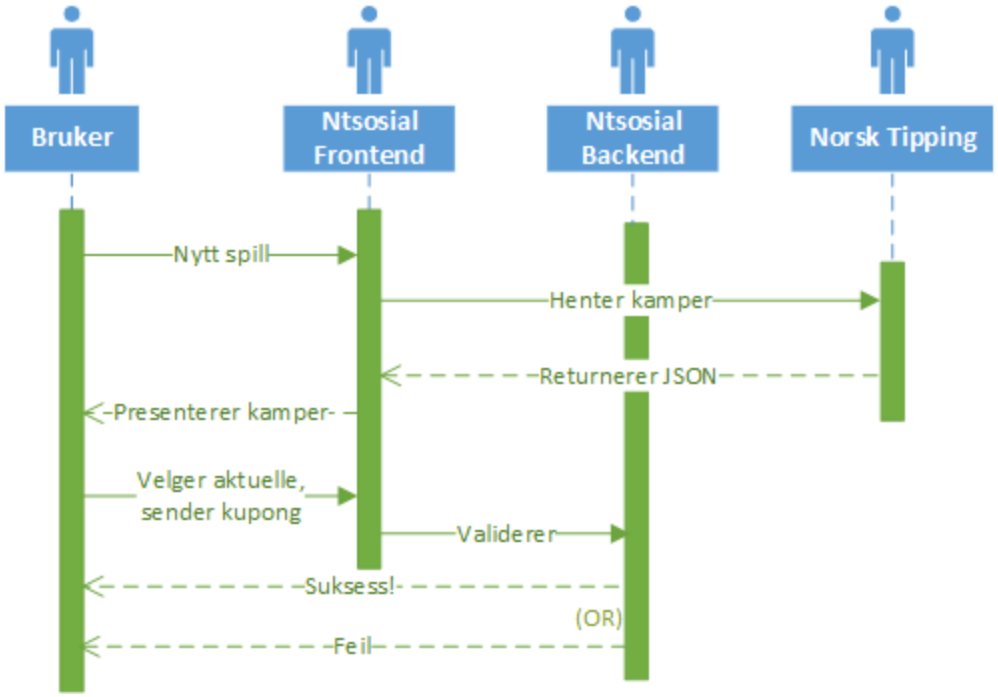
Hver av disse Use Case-ene inneholder et Scope, som i vårt tilfelle hovedsakelig er nettsiden. Vi har også pre- og postbetingelser, der prebetingelser er krav vi har satt som må innfris for at hendelsesforløpet skal utartes, og postbetingelser er hva som skal skje når hendelsesforløpet fullføres uten feilsituasjoner.

Use Case	**Opprette brukerkonto via Facebook**
Scope	Nettsiden, Facebook
Aktør	Sluttbruker
Pre-betingelser	Bruker må ha en aktiv Facebook-konto Bruker må være koblet til Internett
Post-betingelser	Brukerdataene må lagres i databasen etter autentisering/godkjenning
Detaljert hendelsesforløp	1. Brukeren klikker på «Logg inn med Facebook» 2. Verifiserer applikasjonen på Facebook 3. Returnereres til nettsiden 4. Informasjonen blir lagret i databasen
Feilsituasjoner	2. Brukeren verifiserer ikke applikasjonen 1. Brukeren får beskjed ved returnering om at dette kreves for å kunne benytte applikasjonen. 2. Brukeren får opp «Logg inn med Facebook»-knappen på nytt.

Use Case	**Vise spillelag**
Scope	Nettsiden
Aktør	Sluttbruker
Pre-betingelser	Bruker må være medlem av et spillelag
Post-betingelser	Bruker får opp siden til et spillelag
Detaljert hendelsesforløp	1. Brukeren klikker på et spillelag 2. Kuponger, medlemmer, kommentarfelt og info lastes inn

Feilsituasjoner	2.1 Ingen foreslåtte kuponger i spillelaget 1. Melding vises der kupongen ville vært, med en lenke til hvor man kan opprette en kupong. 2.2 Ingen aktive/historiske kuponger 1. Melding vises om at det ikke finnes noen aktive eller historiske kuponger.

Use Case	**Foreslå nytt spill (LangOddsen)**
Scope	Nettsiden, Norsk Tipping
Aktør	Sluttbruker
Pre-betingelser	Bruker må være koblet til Internett
Post-betingelser	Spillinformasjonen lagres i databasen
Detaljert hendelsesforløp	1. Brukeren klikker på "Nytt spill" 2. XML lastes fra Norsk Tipping 3. Presenterer aktuelle kamper 4. Bruker velger kamper 5. Bruker klikker på "Send inn kupong" 6. Bruker velger beløp, ant. andeler, spillelag 7. Bruker velger å delta med penger eller poeng 8. Bruker velger "Submit"
Feilsituasjoner	2) JSON blir ikke hentet på grunn av korrupt fil eller nedetid 1. Feilmelding vises 5) Ingen kamper er valgt 1. Feilmelding vises 2. Kupong kan ikke sendes inn 6) Ugyldige data 1. Viser feilmelding på hvilket felt som er ugyldig/mangelfullt. 2. Bruker kan endre på feltene og prøve igjen 7) Ikke nok poeng til å sende inn kupong 1. Bruker får beskjed om at poengsummen er for lav 2. Bruker kan endre beløpet



Figur 2 Sekvensdiagram for foreslå nytt spill

Use Case	Delta på spill (Tilskuer)
Scope	Nettsiden
Aktør	Sluttbruker
Pre-betingelser	Foreslåtte kuponger er tilgjengelige
Post-betingelser	Bruker blir lagt til som tilskuer på spillet
Detaljert hendelsesforløp	1. Foreslåtte spill presenteres 2. Bruker velger å delta som tilskuer
Feilsituasjoner	1) Ingen foreslåtte spill er tilgjengelig 1. Melding vises dersom det ikke finnes noen foreslåtte spill. 2. Bruker får en lenke for å opprette et eget spill 2) Ikke nok poeng for å delta som tilskuer 1. Bruker får opp en feilmelding om at han/hun ikke har nok poeng for å delta

Use Case	Hente og oppdatere resultater
Scope	Nettsiden, Norsk Tipping
Aktør	Webserver
Pre-betingelser	Kontakt med Norsk Tippings resultatserver
Post-betingelser	Fullførte kamper (og evt. kuponger) blir oppdatert
Detaljert	1. De siste kampenes resultater hentes hver time vha. Cron

hendelsesforløp	2. Det sjekkes om noen av spillerene har spilt på kampen(e) 3. Dersom dette er tilfelle blir kampen satt til seier eller tap i databasen. Skriver til loggfil. 4. Hvis kupongen også er fullført blir kupongen satt til seier eller tap i databasen. Skriver til loggfil.
Feilsituasjoner	1.1) Ingen kamper blir returnert 1. Skriptet fullføres uten å gjøre noen oppdateringer 2.1) Ingen av spillerene har spilt på kampen 1. Oppdatering gjøres ikke 2. Skriver til loggfil at ingen endringer er gjort

2.2. Supplementære spesifikasjon

2.2.1. Brukervennlighet

Siden skal fungere like godt på mobil som på PC. Brukermassen er potensielt av alle aldre, og dermed er det også ønskelig at presentasjonen av nettsiden og navigasjonen tar hensyn til dette. Rettningslinjene til Difi for universell utforming av IKT skal brukes (3).

2.2.2. Sikkerhet/omgivelser

For å få tilgang til Sosiale spill må brukeren ha en Facebook-profil. Det er Facebook som tar seg av autentiseringen, og videre gir oss muligheten til å hente ut objekter tilknyttet brukerkontoen som brukerens ID og navn.

Oppdragsgiver vil ha muligheten til å også kunne benytte andre aktører til å kunne lage konto, derfor lager vi denne autentiseringen i en egen fil for å gjøre det lett å bytte autentiseringsmetode til f.eks Twitter/Google eller andre.

Produktet vi skal utvikle er en prototype som bare er tilgjengelig for venner, og ansatte hos oppdragsgiver for testing.

2.2.3. Lisenser

Vi skal kun benytte oss usmittsomme lisenser (MIT/Apache 2.0) og skal kreditere forfatterene av de bibliotekene vi benytter oss av.

2.2.4. Delutgivelser

Nye implementasjoner og forbedringer blir lastet opp på webserveren daglig. Oppdragsgiver har dermed gjennom hele utviklingsfasen hatt mulighet til å logge seg inn og se status og komme med tilbakemeldinger etter hver sprint under retrospective/review.

2.3. Produktkø

Produktkøen vist i Figur 3 slik den så ut da vi var ferdige viser alle de funksjonene vi utviklet for å nå målene. Under utviklingen er denne alltid levende, og elementer kan bli lagt til eller fjernet. Vi kan derimot ikke legge til eller fjerne elementer når en sprint er aktiv.

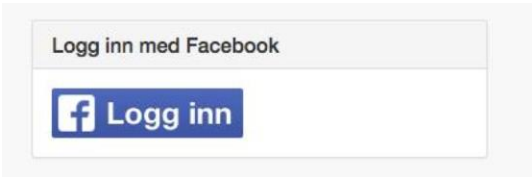
Key	Summary	Assignee	Done
NTSOS-4	Lagre innloggingsdata i db	Jan Fredrik Gundersen	Sprint 1
NTSOS-1	Sette opp server med database.	Jan Fredrik Gundersen	Sprint 1
NTSOS-2	Innlogging via Facebook	Sondre T Johannessen	Sprint 1
NTSOS-9	Designe poengsystem	Sondre T Johannessen	Sprint 1
NTSOS-21	Workshop med IBM	Unassigned	Sprint 2
NTSOS-20	Opprette spillelag	Jan Fredrik Gundersen	Sprint 2
NTSOS-3	Invitere venner til spillag	Jan Fredrik Gundersen	Sprint 2
NTSOS-25	Begynt på Mine poeng og Ny kupong	Sondre T Johannessen	Sprint 2
NTSOS-22	Bli medlem av invitert spillelag	Jan Fredrik Gundersen	Sprint 2
NTSOS-5	Liste over spillelag som en spiller er medlem i	Jan Fredrik Gundersen	Sprint 3
NTSOS-26	Utmelding av spillelag	Jan Fredrik Gundersen	Sprint 3
NTSOS-11	Slette spillag	Jan Fredrik Gundersen	Sprint 3
NTSOS-24	Lagre brukerID sikkert i nettleser	Jan Fredrik Gundersen	Sprint 3
NTSOS-23	"Min side"-funksjon	Jan Fredrik Gundersen	Sprint 3
NTSOS-17	Utvide database til å omfatte poengregistrering og lag.	Sondre T Johannessen	Sprint 3
NTSOS-6	Fylle ut kupong, levere den privat eller til spillelaget	Sondre T Johannessen	Sprint 4
NTSOS-14	Kartlegge relevant API funksjonalitet til NT.	Sondre T Johannessen	Sprint 4
NTSOS-31	visSpillelag: "Tegne" og skrive kode for hvordan det vil se ut med kuponger	Jan Fredrik Gundersen	Sprint 4
NTSOS-29	Hent kuponginformasjon fra DB og vis statistikk.	Sondre T Johannessen	Sprint 4
NTSOS-30	visSpillelag henter kuponger spillelaget har	Jan Fredrik Gundersen	Sprint 4
NTSOS-27	Restrukturere koden	Jan Fredrik Gundersen	Sprint 4
NTSOS-34	Resultatoppdatering	Jan Fredrik Gundersen	Sprint 5
NTSOS-10	Slette profil	Jan Fredrik Gundersen	Sprint 5
NTSOS-32	Skrive readme for å flytte prosjektet til annen server	Jan Fredrik Gundersen	Sprint 5
NTSOS-33	Poengberegning til spiller	Jan Fredrik Gundersen	Sprint 5
NTSOS-8	Skrive innledning	Sondre T Johannessen	Sprint 5
NTSOS-35	Rydde opp i layout	Sondre T Johannessen	Sprint 5
NTSOS-7	Fin statistikk/grafer	Sondre T Johannessen	Sprint 5

Figur 3 Produktkøen (Hentet fra JIRA)

3. Brukergrensenitt

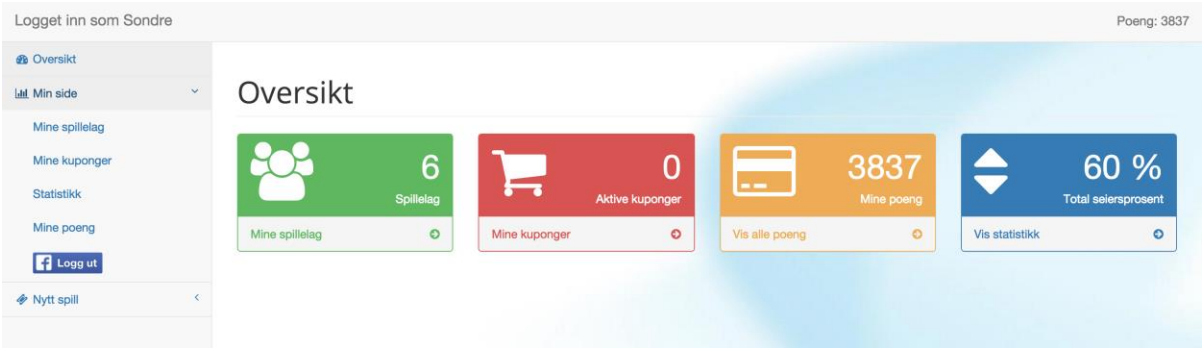
Websiden skulle være responsiv og pen, noe det var fokus på når brukergrensesnittet skulle designes. Vi valgte å gå for Bootstrap som er et rammeverk for HTML, CSS og JavaScript og må inkluderes i prosjektet (4). Valget ble gjort på bakgrunn av demosidene som Bootstrap har liggende ute var appellerende for mobil og web. I tillegg er rammeverket ekstremt populært som gjør at det er veldokumentert og at det ligger mange eksempler ute på nettet. Bootstrap gir enkelt muligheten for pent design og responsive websider som justerer seg etter alle enheter de blir brukt på. Det er registrert under MIT-Lisensen som gjør at vi kan bruke det til hva vi vil. Med tanke på det hadde vi allerede løst problemet med tilpassing til andre enheter før det startet, og vi kunne definere brukergrensesnittet slik vi ønsket.

Figur 4 Endelig design på innloggingssiden



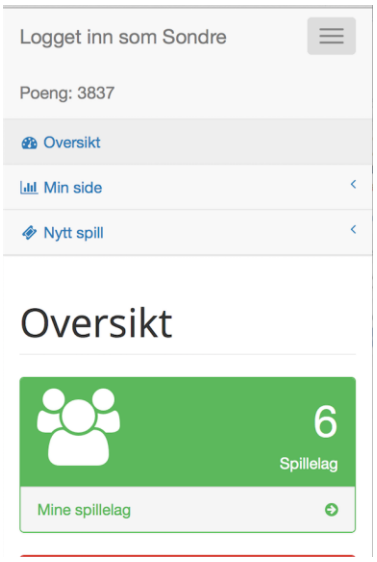
Figur 5 Første utkast på innloggingssiden

Det første brukeren møter er en side (Figur 4) som forteller at man må logge seg inn på webapplikasjonen gjennom Facebook og en liten tekst som viser at det er Sosiale spill det logges inn på. Ved første utkast (Figur 5) manglet dette og vi hadde en original Facebook logg inn knapp. Når innloggingen er fullført via Facebook blir bruker videresendt til oversikten. Her vises en oversikt over hvor mange spillelag en bruker er medlem i, hvor mange aktive kuponger en bruker har, total seiersprosent og en oversikt over poengene til brukeren. Disse er representert i fire ulike bokser (Figur 6) og er klikkbare med link til en mer detaljert oversikt.

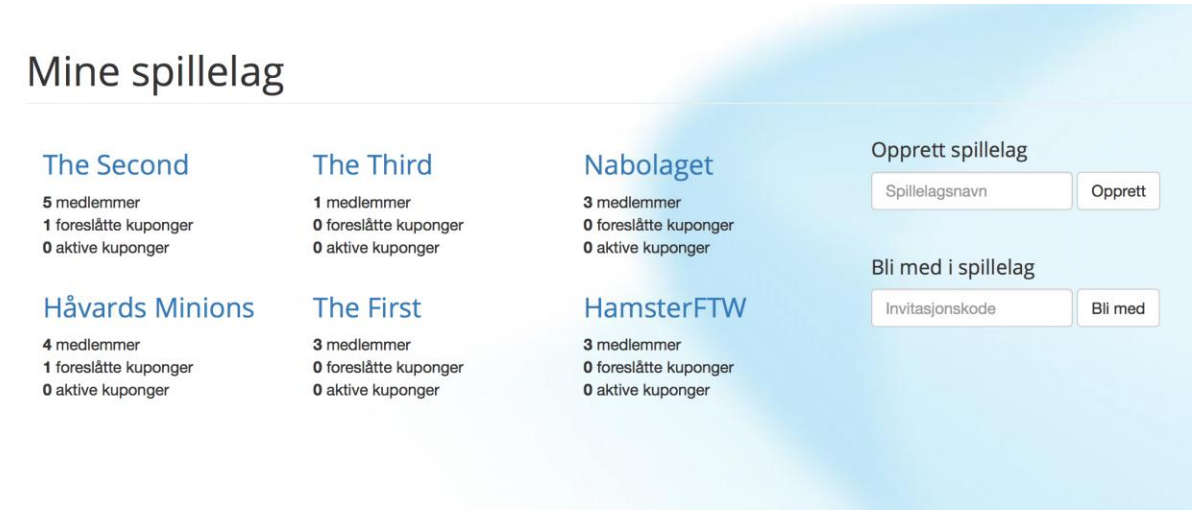


Figur 6 Utseende til oversiktdelen

Figur 7 Mobilversjon av oversiktdelen



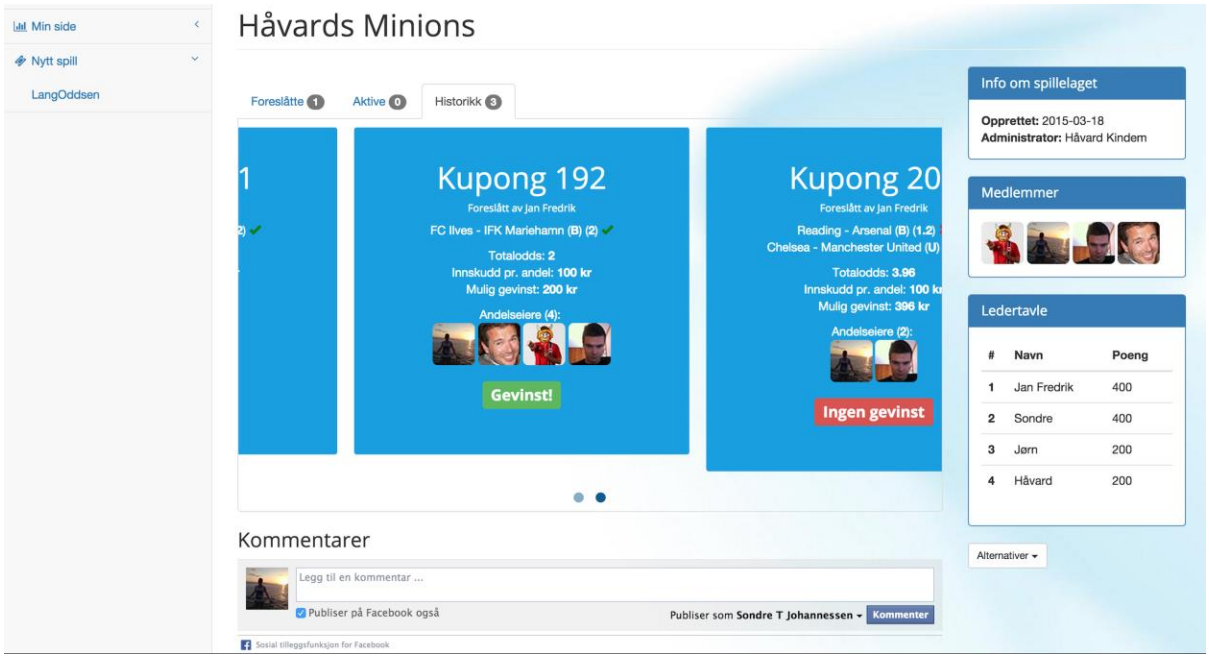
Til venstre finner man hovedmenyen (Figur 6) som består av tre hovedfaner: Oversikt, Min side og Nytt spill. Denne menyen er tilgjengelig uansett hvor bruker skulle befinne seg. Dersom man klikker på Nytt spill eller Min side får man opp flere faner. Under Min side får man faner med valgene: Mine spillelag, Mine kuponger, Statistikk, Mine poeng og Logg ut. Under «Nytt spill»-fanen får man en fane med valget: LangOddsen. Bootstrap funksjonalitet ordner det slik at det kun kan være en fane åpen samtidig slik at det ser mer ryddig ut. Ved visning på mindre enheter, som for eksempel mobil, blir denne menyen flyttet til en liten knapp oppe til høyre på enheten (Figur 7). Når bruker klikker på den vises menyen. På toppen finner man en bar som representerer hvem som er logget inn og hvor mange poeng han eller hun har.



Figur 8 Endelig design på Mine spillelag

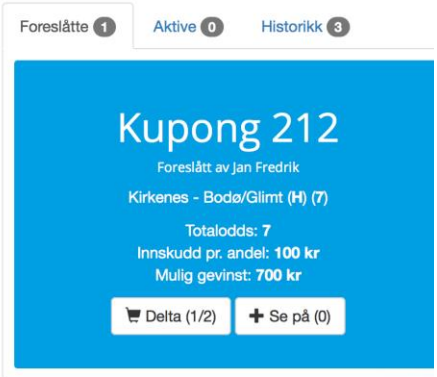
Figur 9 Første utkast på design av Mine spillelag

Når bruker klikker seg inn på «Mine spillelag» (Figur 8) blir han/hun videresendt til en side som viser en oversikt over alle spillelagene bruker er med i. Under hvert spillelag kan bruker se hvor mange medlemmer, foreslåtte kuponger og aktive kuponger spillelaget har. Det gis også mulighet til å opprette et spillelag og å bli medlem av et spillelag ved at bruker taster inn en kode. Utseende ble endret mot slutten av prosjektet da vi følte det manglet noe ved presentasjonen fra det første utkastet (Figur 9).



Figur 10 Et spillelags private side

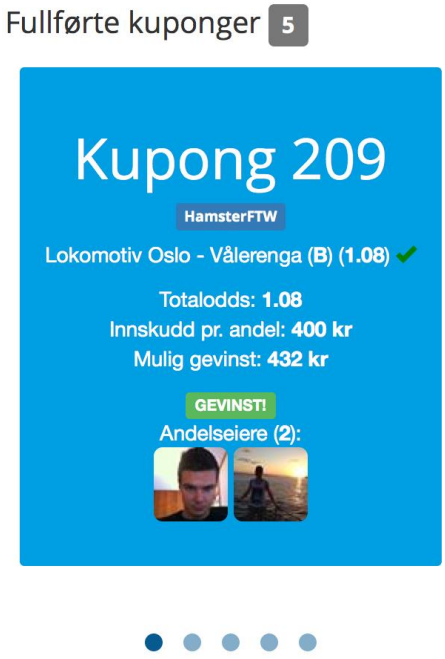
Når man entrer et spillelag sin private side (Figur 10) finner man en oversikt over kuponger innad i laget, en informasjonsboks, en boks med bilder over alle medlemmer, en ledertavle, en knapp som gir alternativene for utmelding av spillelaget eller invitere venner, og et kommentarfelt. Oversikten over kupongene er delt inn i fanene foreslåtte, aktive og historikk, hvor hver fane har et tall som viser hvor mange det finnes i den aktuelle statusen. Her blir kupongene presentert i en "karusell" der man ved hjelp av musepekeren drar seg bortover for å bla gjennom kuponger eller swiper på mobilen. Vi valgte denne måten å presentere på fordi med denne funksjonaliteten kan man se kupongene i sin helhet, bla gjennom alle og i tillegg samhandle med hver enkelt. Hver kupong viser hvilken bruker den er foreslått av, totaloddsen og kostnad. I tillegg har de ulikt design under hver fane.



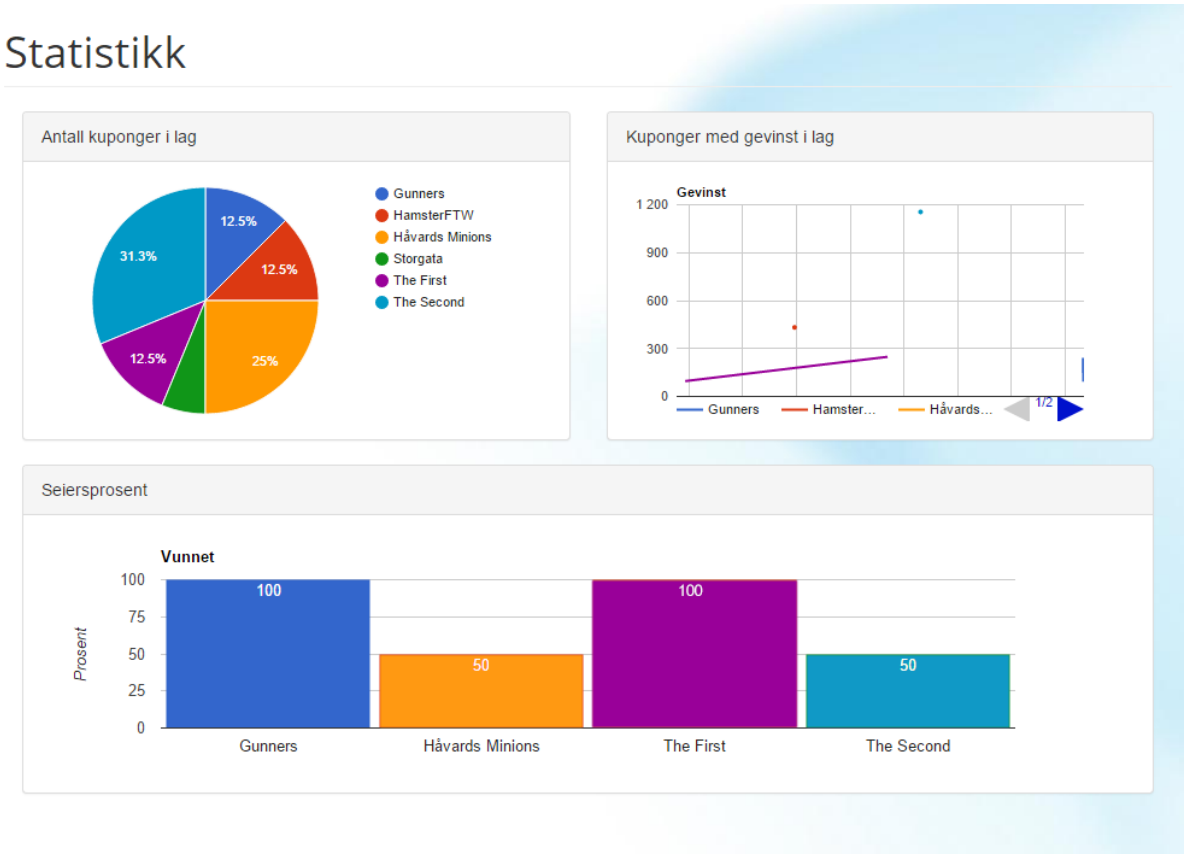
Figur 11 Endelig design på foreslått kupong

De som ligger under «foreslåtte» (Figur 11) har to knapper som gir brukeren mulighet til å delta på kupongen som betalende eller tilskuer. Under "aktive" vises det hvem som er andelseiere i kupongen og hvilke kamper som er gått inn, og under "historikk" vises det i tillegg om det ble gevinst eller ikke.

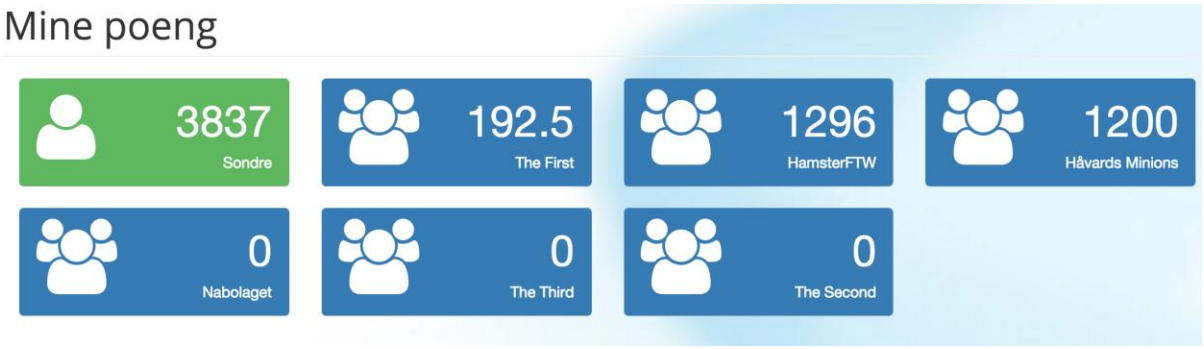
Velger bruker "Mine kuponger"-fanen i menyen blir han/hun sendt til en oversikt alle kuponger som er aktuelle for bruker. Denne oversikten er veldig lik kupongoversikten i spillelag, men har i tillegg en knapp som henviser til spillelaget (Figur 12) kupongen tilhører.



Figur 12 Endelig design av en fullført kupong på Mine kuponger - siden

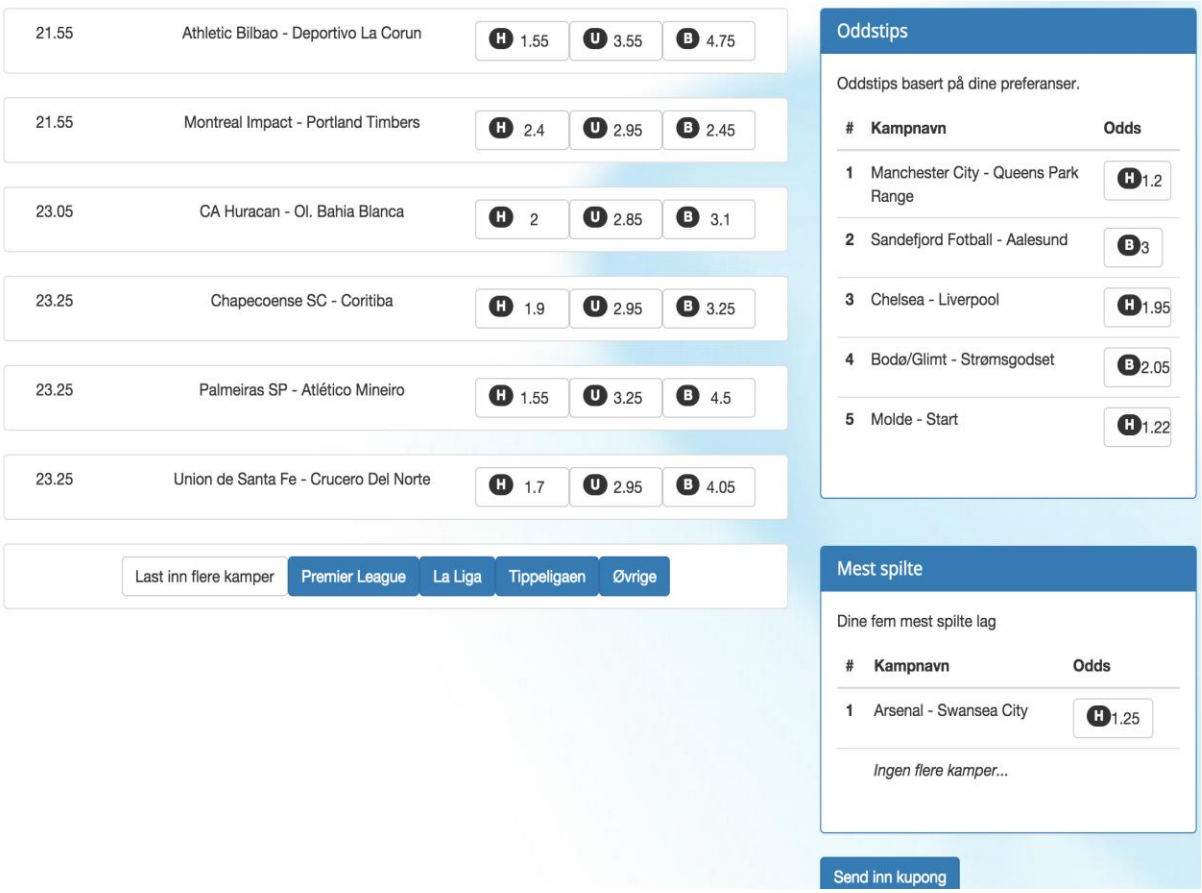


Figur 13 Utseende på statistikksiden



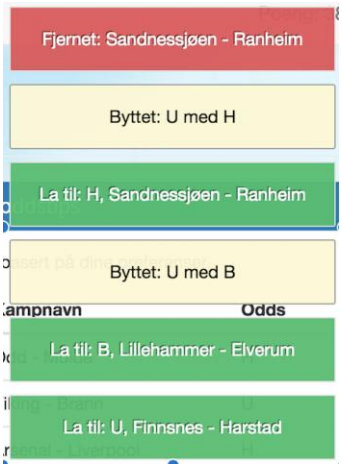
Figur 14 Utseende på Mine poeng-siden

Under "Statistikk" (Figur 13) vises oversikter over antall kuponger levert i hvert lag, antall kuponger med gevinst i lag og seiersprosent i de ulike lagene. Henholdsvis presentert ved pai-graf, linje-graf og kolonne-graf. Klikker bruker seg inn på "Mine poeng" (Figur 14) er det en oversikt over bruker sine poeng totalt og i de ulike lagene, vist i bokser som henviser til laget.

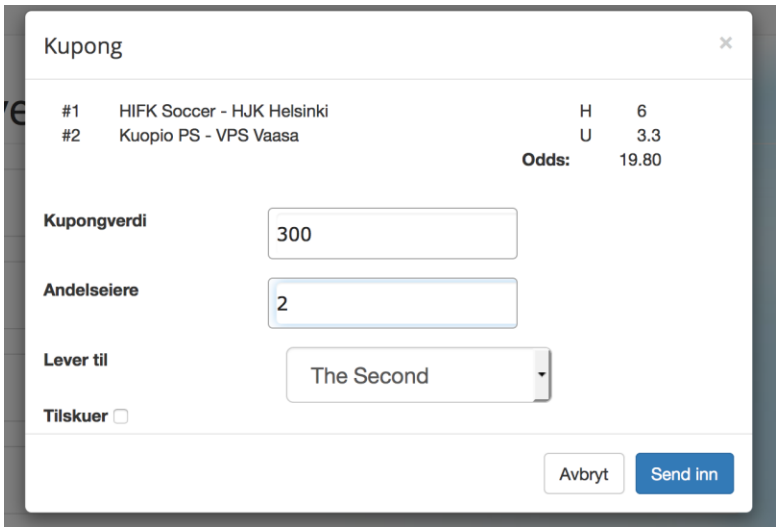


Figur 15 Representasjon av Nytt LangOddsen spill

Figur 16 Første utkast til design av LangOddsen-kamper



Figur 17 Varslinger ved kampvalg og kampbytte

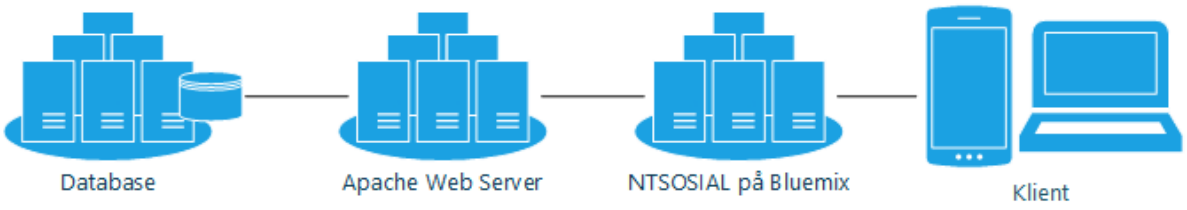


Figur 18 Spesifikke kupongopplysninger

Ved valg av "LangOddsen" kommer brukeren til en side (Figur 15) hvor alle fotballkampene som er tilgjengelige for spill fra Norsk Tipping med valg for å sortere kampene basert på liga og laste inn flere kamper, noe vi ikke hadde med i første utkast som vist i Figur 19. Til høyre er det to bokser som gir brukeren forslag til kamper, og under er en knapp for å sende inn kupongen. Når den knappen blir klikket får man et vindu (Figur 18) hvor bruker kan velge hvor mange andelseiere, kupongverdi, til hvilket lag og om man skal være tilskuer eller ikke, i tillegg til to knapper som enten sender kupongen inn eller avbryter. Når bruker klikker på kamper vises det varslinger (Figur 17) om hvilken handling som ble gjort.

4. Arkitektur

4.1. Klient-Tjener

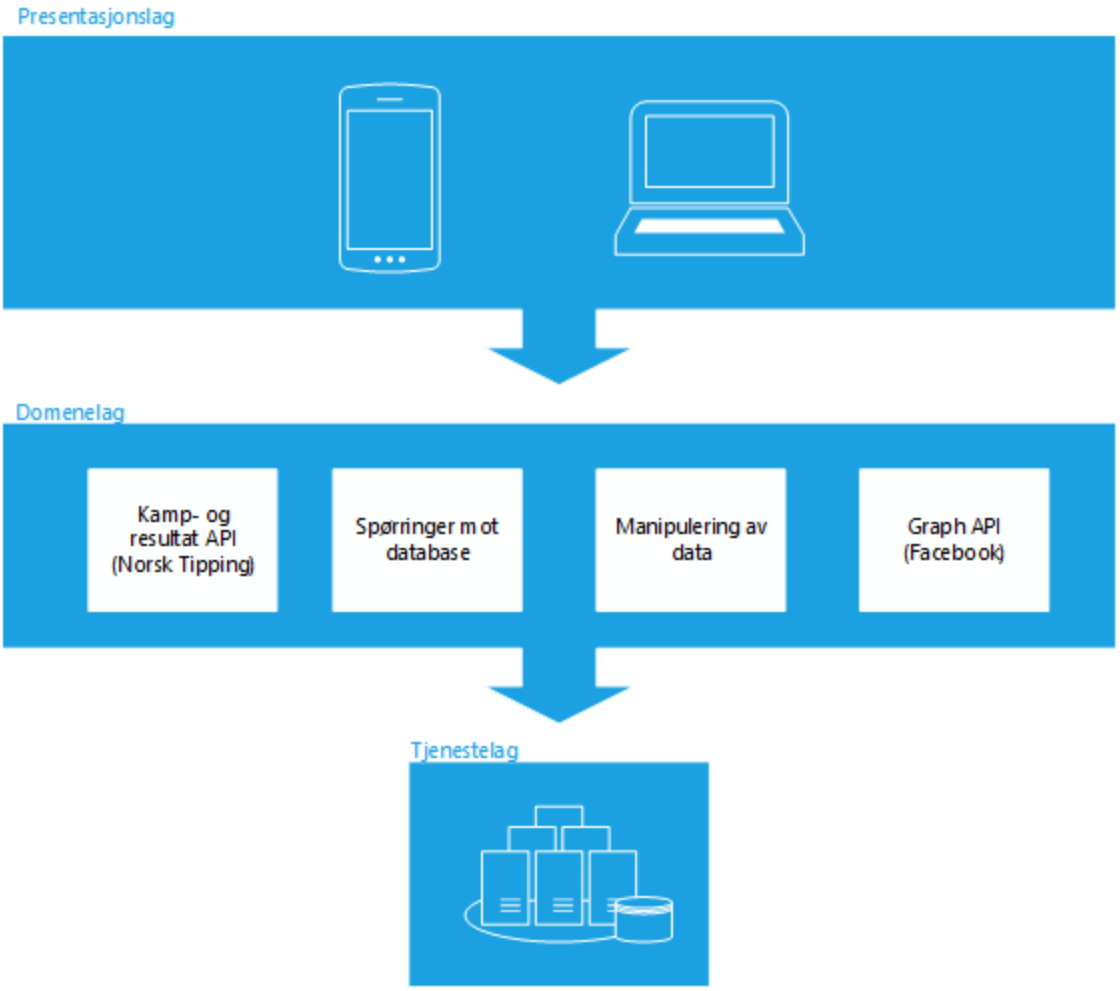


Figur 20 Klient-tjener arkitektur

Hele systemet er basert på en klient-tjener arkitektur som vist i Figur 20. Siden vi utviklet en responsiv webside med støtte for mobil trengte vi ikke å ta hensyn til en eventuell mobilapplikasjon.

Dette medfører at strukturen blir som modellen over med en webside som kommuniserer med en webserver som behandler databasen. Websiden tar seg av presentasjon av data fra databasen og noe behandling av data for å unngå unødvendig redundans i databasen. Mesteparten av databehandlingen blir gjort på serveren. Vi har etter beste evne kodet slik at webserver og databasen tar seg av så mye som mulig av arbeidskapasiteten. Dette fordi vi vil ha mest mulig forutsigbar responstid hos klientene og siden en klient kan ha ulik prosessorkraft og ulike nettlesere er det en fordel å gjøre arbeidet som kreves hos klienten så lite som mulig. Siden websiden også skal brukes mye på mobil vil vi spare batteritid ved å løse det på denne måten.

4.2. Trelagsstruktur



Figur 21 Trelagsstruktur

Vi valgte trelagsstruktur siden dette er den mest naturlige måten å strukturere en webapplikasjon på. Den består av et presentasjonslag, et domenelag og et tjenestelag. Ut i fra Figur 21 er lagdelingen ganske klar, men i webkoding er det noen ganger lettere og mer effektiv at funksjonalitet blir slått sammen, for eksempel at en fil gjør alt selv om strukturen tilsier at den skal bli delt opp. Dette har vi gjort noen ganger, men for det meste er strukturen i takt med modellen.

4.2.1. Presentasjonslag

Presentasjonslaget tar for seg det brukeren ser og gjør, det er dette laget som har med brukergrensesnittet å gjøre. Her er alle knapper, menyer, grafer, mm, som brukeren samhandler med og kaller på funksjonalitet fra domenelaget. Det kan være å opprette en

kupong, sende den inn og trykke seg inn på et spillelag. Koden i presentasjonslaget består av HTML, CSS, JavaScript, jQuery. Det er nettleseren selv som står for kompilering av koden slik at funksjonaliteten der bestemmer hvilke data som eventuelt skal bli forsøkt hentet.

4.2.2. Domenelag

Domenelaget sin oppgave er å koble sammen tjenestelaget og presentasjonslaget. Det består av webserver og PHP-filer med funksjonalitet for å hente og manipulere data fra tjenestelaget slik at det kan bli brukt i presentasjonslaget. Det sørger også for at ingen korrupte data blir sendt fra presentasjonslaget til tjenestelaget. Her har vi for eksempel addnykupong.php og getkuponger.php som begge behandler forespørsler fra presentasjonslaget, forsikrer seg om at disse er korrekte og utfører forespørselen på tjenestelaget.

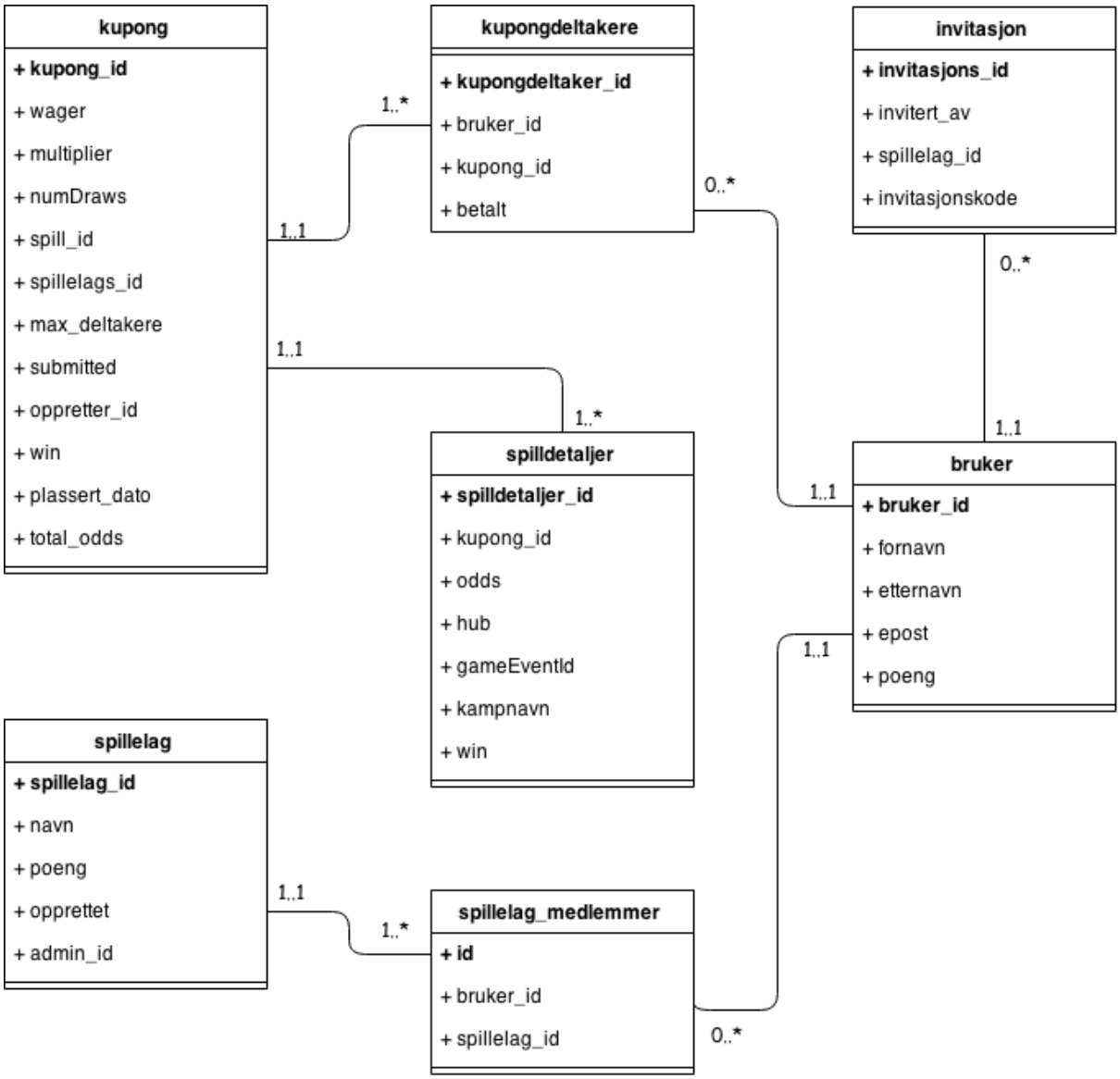
4.2.3. Tjenestelaget

Tjenestelaget består av MySQL databasen. Her blir alle relevante data lagret i tabeller med relasjoner til hverandre som brukere, spillelag og kuponger. Her mottas og behandles spørringer fra domenelaget.

4.2.4. Øvrig

Denne typen struktur gjorde at vi lett kunne separere og fordele oppgavene, siden de ulike lagene gjorde det lett å stykke det opp. Foruten trelagsstrukturen benytter vi oss av APIer fra Norsk Tipping, Facebook og som vi har laget selv ved hjelp av IBM Bluemix. Disse kommuniserer direkte med presentasjonslaget, men tillater kun lesing av data.

4.3. Databasedesign



Figur 22 Databasedesign

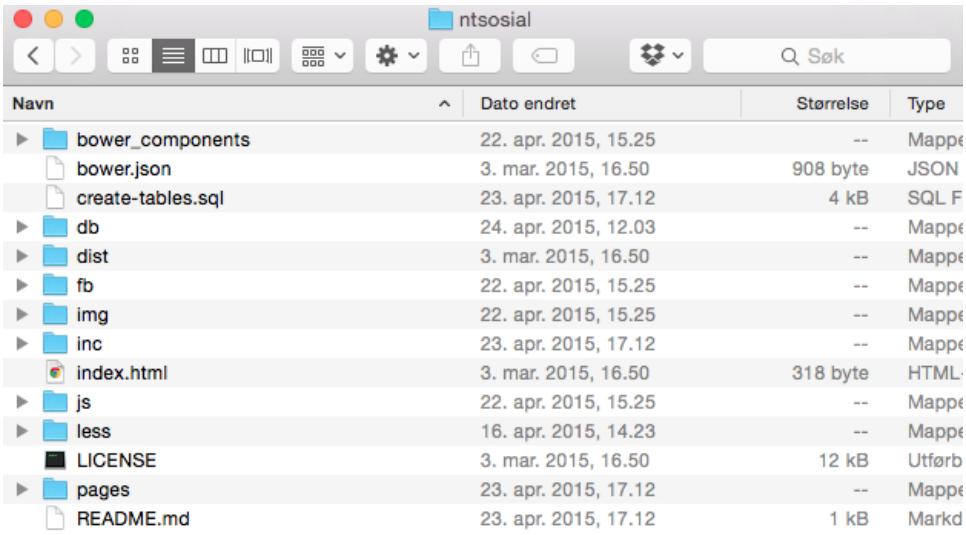
Figur 22 beskriver databasedesignet vi har laget. Når en bruker godkjenner applikasjonen vår via Facebook opprettes det en ny rad i tabellen "bruker", der bruker_id, navn og epost hentes fra Facebook. Poeng blir ved nyregistrering satt til 500 slik at man kan spille fra første besøk.

For å holde oversikt over hvilke brukere som er med i hvilke spillelag, og hvilke kuponger brukerene er med på har vi valgt å bruke to hjelpetabeller i henhold til normaliseringsstandarene (3NF)(5).

Når en bruker oppretter et spillelag opprettes det en ny rad i spillelag og i spillelag_medlemmer. Når en bruker blir med i et spillelag opprettes det kun en ny rad i spillelag_medlemmer som inneholder brukerens og spillelagets id. Det samme prinsippet gjelder for "kupong", der brukeren som opprettet kupongen blir lagt til som oppretter og samtidig blir lagt i "kupongdeltakere". Dersom man blir med på en kupong opprettes det kun en ny rad i kupongdeltakere, der det også angis om man er med som betalende eller tilskuer.

4.4. Filorganisering

Vi har strukturert filene våre i mapper som forteller hva slags filer de inneholder. Dette gjør det lettere å utvikle og gjør det enklere for evt. videreutvikling.



Figur 23 Rotmappen til ntsosial

I Figur 23 vises rotmappen til ntsosial. Bower.json inneholder informasjon om hva slags Bootstrap versjon vi bruker og hvike tillegg som hører med. Create-tables.sql oppretter de nødvendige tabellene for at applikasjonen skal fungere, index.html videresender til forsiden/login, license inneholder lisensinfo og readme.md inneholder info om prosjektet og installasjonsveiledining. Dette skjer i de forskjellige mappene:

- bower_components

  Inneholder Bootstrap-koden (som CSS og JS). Inneholder også andre filer som Font Awesome og JqueryUI.

- db

  PHP filer med databasespørringer.
- Dist

  Inneholder CSS og JS filer får vårt tema "SB Admin 2".
- fb

  Facebook PHP SDK
- img

  Bilder som brukes på nettsiden
- inc

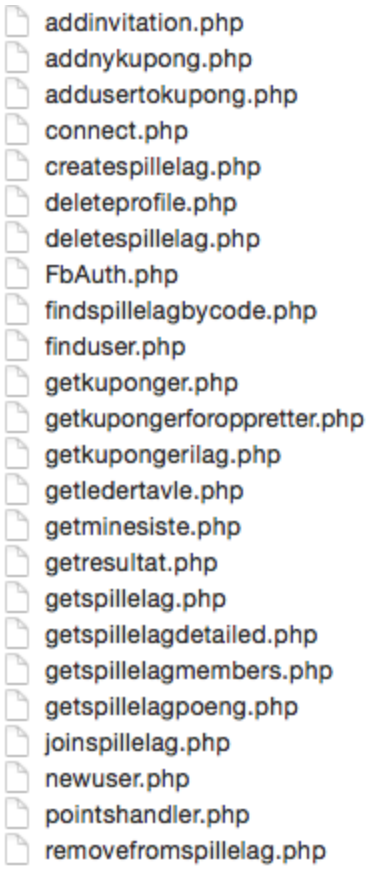
  Filer vi inkluderer på alle sider for å unngå duplisering av kode, som header og navigasjon
- js

  JavaScript filer, som Carousel, Alertify og Google Charts
- less

  Less filer som brukes av Bootstrap for utvides CSS funksjonalitet
- pages

  Navigerbare sider, som statistikk.php og mineKuponger.php

Figur 23 viser db-mappen. Vi valgte å lage en fil for hver databasespørring, med unntak av en av de siste funksjonene vi laget (PointsHandler.php). I ettertid ville vi ha gruppert flere spørringer i en fil, f.eks UserHandler.php som da tar seg av brukerbehandling.



Figur 24 Mappen for databasespørringsfiler

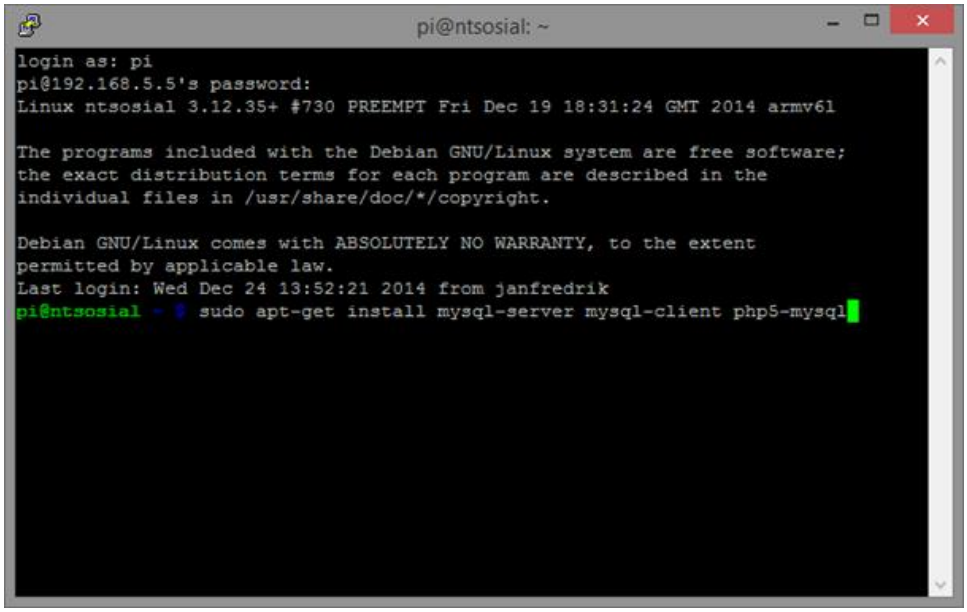
5. Implementering

5.1. Server

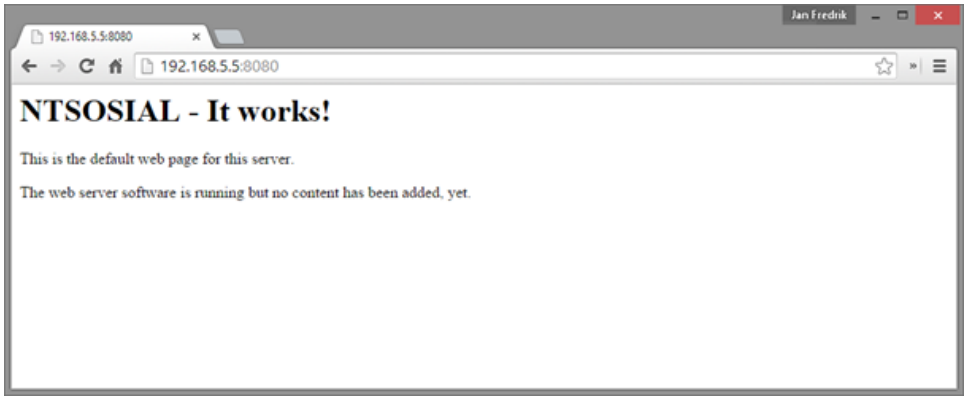
For å få på plass løsningen vår var det behov for en web/databaseserver. Vi hadde ikke dette tilgjengelig, så vi bestemte oss for å sette opp en egen midlertidig løsning. Til dette benyttet vi oss av en Raspberry Pi.

Vi installerte Debian operativsystemet på enheten og la deretter inn Apache, PHP og MySQL. Figur 25 og Figur 26 viser grensesnittet og websiden.

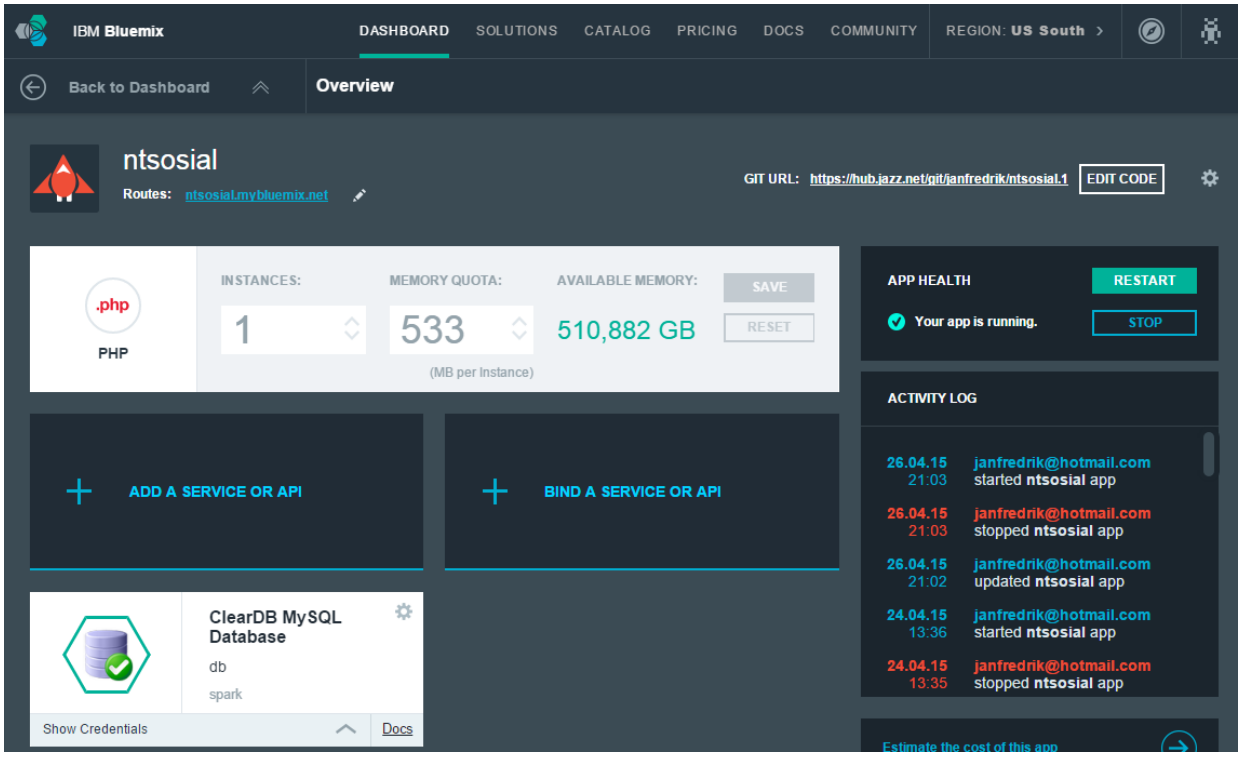
Denne løsningen stod hjemme hos Jan Fredrik som ikke hadde så bra nettlinje. På grunn av dette, og at enhetens disk var et SD-kort ble aksesseringen utenfra og I/O treg, noe som var irriterende for utviklingen. Norsk Tipping ønsket derfor at vi brukte IBM Bluemix for utviklingen, da de vurderer å bruke denne tjenesten i fremtiden og ønsket våre tilbakemeldinger. Her satt vi opp vår egen PHP løsning med en MySQL database som vist i Figur 27.



Figur 25 Grensesnittet vårt mot Raspberry Pi serveren



Figur 26 Kontakt med webserveren på lokalt nettverk



Figur 27 Vår Bluemix-applikasjon med PHP og MySQL

5.2. Utviklingsverktøy

Løsningen vår har blitt utviklet med de verktøyene vi mente egnet seg best. Siden vi koder for web er det da selvsagt at vi bruker HTML, CSS og JavaScript. Vi vurderte å bruke node.js som serverspråk istedenfor PHP. Det er et asynkront og event-basert rammeverk som er basert på JavaScript. Det gjør det enklere for mindre erfarne programmerere å bygge skalerbare nettverksapplikasjoner og hindrer dead-locking av prosesser. (6) Vi valgte til slutt

å bruke PHP som serverspråk. Grunnen til dette er at vi tidligere har kodet i PHP, det har en stor brukerbase som gjør det enklere å finne løsninger på feil eller spørsmål rundt om på nettet, og det har god kompatibilitet på tvers av versjoner. I tillegg til dette har også Facebook et eget SDK for PHP som vi har benyttet oss av for integrasjon.

Vi valgte også å gå for en MySQL-database. Grunnen til dette er at vi trengte en skalerbar og fleksibel relasjonsdatabase for å håndtere dataene våre. MySQL samarbeider godt med PHP og har mange av de samme fordelene. I tillegg til dette har vi også tidligere benyttet oss av det. Det kjører også på Windows, UNIX og Mac OS og er utprøvd og testet over flere år. (7)

For å kjøre spørringer på databasen benytter vi oss av mysqli. Her stod valget mellom dette og PDO. Begge er like sikre hvis GET/POST-dataene blir behandlet riktig, men mysqli krever mindre kode, er lettere å lese og er rundt 6 % raskere (8). Ulempen er at mysqli kun støtter MySQL driveren.

Vi har benyttet oss av den åpne tjenesten Word Online for rapportskriving. Dette har gjort det mulig for oss å skrive i samme dokument uavhengig av hvor vi sitter. De fleste av figurene våre er laget i Microsoft Visio. Databasemodellen er laget på nettsiden Draw.io mens Gantt-diagrammet er laget i Microsoft Project.

Møtereferat- og arbeidslogger har blitt utarbeidet i Google Docs.

5.2.1. IBM Bluemix

IBM Bluemix ble lansert 30.juni 2014 og er for stor til at vi kan dekke alt her. Vi vil dekke det som vi mener er nok til å forstå hvordan vi har brukt det. Tjenesten er en nettskybasert åpen plattform basert på open source teknologien Cloud Foundry. Hensikten er å kunne tilby tjenester som er klare for bruk umiddelbart og hosting-miljø som har kapasitet for intern skalering. I praksis vil dette si at om man har en webapplikasjon som er allokert 128mb minne til og det trengs mer minne trykker man på «legg til mer minne»-knappen (Figur 27) og Bluemix gjør alt for deg. Node.red er tjenestens GUI for å enkelt kunne sette opp APIer. Den er basert på open source teknologien node.js og gir brukeren mulighet til å raskt sette opp egne APIer basert på GET- og POST- HTTP-requester sammen med et hav av templates som er klare for drag and drop. Her finner man alt fra SMS-tjenester til avanserte algoritmer som gjenkjenner bilder. Det er denne delen av tjenesten vi har brukt.

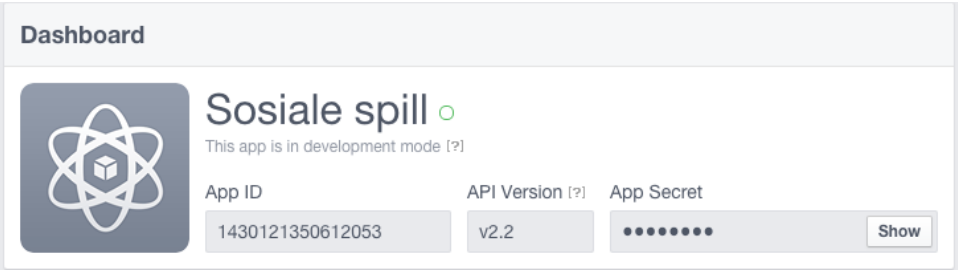
Norsk Tipping hadde allerede vurdert å teste ut tjenesten for å se om dette var noe de ville bruke i fremtiden. IBM kom på banen tidlig i prosjektet og presenterte potensielle løsninger som vi kunne bruke, som oppsett av APIer, SMS-tjenester og analyseverktøy. Norsk Tipping ønsket at vi skulle ta denne tjenesten i bruk slik at de kunne danne seg et inntrykk på om det var noe de ville fortsette med. Vi gjorde en vurdering på dette, og var begge enige om at det virket spennende og interessant, men var usikre på om det ville støtte våre kodespråk bra nok – noe det gjorde. Slik fikk vi samlet både webserveren, websiden og databasen under samme plattform. Tilgangen ga oss også mulighet for å bruke IBM Watson som er en tjeneste for kognitive analyser.

5.3. Facebook-integrasjon

For at vi kunne integrere løsningen vår måtte vi først opprette en applikasjon i Facebook. I dette kapittelet vil vi ta for oss denne prosessen og hvilke biblioteker vi benytter oss av fra Facebook.

5.3.1. Opprettelse av applikasjon i Facebook

Ved opprettelse ble vi bedt om å gi applikasjonen et navn, nettadresse og velge en kategori. Vi fikk da tildelt en applikasjons-id og et passord, som vist i Figur 28.



Figur 28 Applikasjonen på Facebook

Etter å ha fått opprettet applikasjonen på Facebook (Figur 28) var vi klare til å integrere Facebook SDK i vår løsning.

5.3.2. JavaScript SDK

Facebook sitt JavaScript SDK (versjon 2.3) bruker vi for å utføre enkle Facebook-integrasjoner (9). Kodesnutt 1 viser all koden som trengs for å opprette et kommentarfelt i

et spillelag. Av Javascript SDKet får vi tilgang til flere eksisterende knapper som, slik som send knappen for invitasjon av venner i Kodesnutt 2, som innehar alt som trengs fra Facebook. Dette gjorde mye av Facebookimplementeringen lettere. For å hente ut informasjonkapselen til en bruker trengte vi kun å sette en variabel (cookie) til sann, som vist i Kodesnutt 3.

Kodesnutt 1 Hente ut kommentarfunksjonaliteten

```
<div class="fb-comments" data-colorscheme="light"
  data-href="http://ntsosial.mybluemix.net/pages/visSpillelag.php?id=
  <?php echo   $id ?>"
  data-numposts="10" width="100%" order_by="reverse_time">
</div>
```

Kodesnutt 2 Sende invitasjon til venner

```
var url = "http://ntsosial.mybluemix.net/pages/invited.php?code=" + code;
$('#share').html('<fb:send href="' + url + '" colorscheme="light" />');
if (typeof FB !== 'undefined') {
    FB.XFBML.parse(document.getElementById('share'));
}
```

Kodesnutt 3 Hente ut Facebook-brukeren sin informasjonskapsel

```
FB.init({
appId      : '1430121350612053',
cookie     : true,  // enable cookies to allow the server to access
                    // the session
xfbml      : true,  // parse social plugins on this page
version    : 'v2.3' // use version 2.3
});
```

5.3.3. PHP SDK

Javascript SDKen var enkel å inkludere, men inneholder begrenset funksjonalitet. Vi så oss nødt til å bruke Facebook sitt PHP SDK (versjon 4.0) i tillegg (10). Det ga oss muligheter for å ha lengre økter og samtidig sikre at en bruker er logget inn (Kodesnutt 5) når en side lastes. Vi kunne enkelt hente ut informasjon om brukeren (Kodesnutt 4) ved å kjøre spørringer til Facebook sitt Graph API for å for eksempel opprette ny bruker i databasen.

Kodesnutt 4 Hente brukerens detaljer for innsetting i brukerdatabasen.

```
$me = (new FacebookRequest(
$session, 'GET', '/me'
))->execute()->getGraphObject(GraphUser::className());
```

```
$userID = $me->getId();
$userName = $me->getName();
$firstName = $me->getFirstName();
$lastName = $me->getLastName();
```

Kodesnutt 5 Verifiserer at brukeren er innlogget

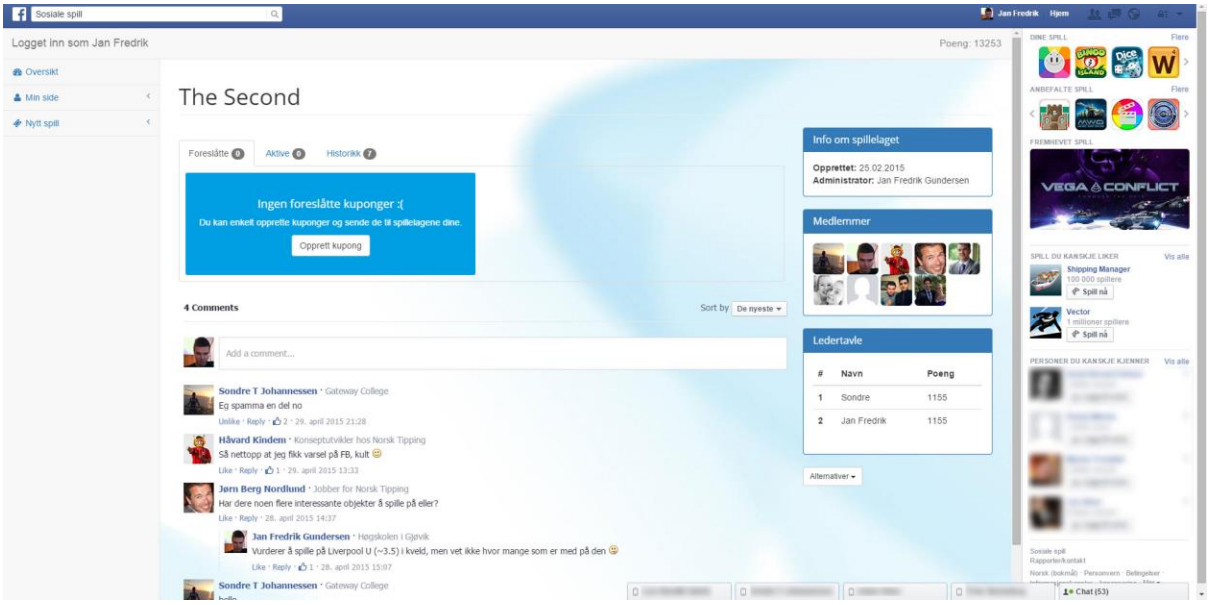
```
if ($session) {
  try {
    $accessToken = $session->getAccessToken();
    $longLivedAccessToken = $accessToken->extend();
    $_SESSION['fbToken'] = $longLivedAccessToken;
  } catch (FacebookRequestException $e) {
    // The Graph API returned an error
  } catch (\Exception $e) {
    // Some other error occurred
}

}
else {
    header("Location: ../pages/login.php");
    die();
}
```

Kodesnutt 6 Sende ut varsel til brukere ved ny kupong

```
// Henter navn på spillelag og innsender, samt ID til alle i spillelag.

$fbsql = "SELECT spillelag.name, user.firstname, user.lastname,
spillelag_medlemmer.user_id FROM spillelag JOIN spillelag_medlemmer ON
spillelag.id = spillelag_medlemmer.spillelag_id LEFT JOIN user ON
spillelag_medlemmer.user_id LEFT JOIN kupong ON kupong.oppretter_id = user.id
WHERE spillelag.id = '$data->spillelags_id' AND kupong.id= '$kupongId' GROUP BY
user_id";

if ($result = mysqli_query($conn, $fbsql)) {
    while ($row = mysqli_fetch_array($result, MYSQLI_ASSOC)) {
        $request = new FacebookRequest(
          $appSession,
          'POST',
          '/'.$row['user_id'].'/notifications',
          array (
            'href' => '/pages/visSpillelag.php?id='.$data->spillelags_id.'',
            'template' => ''.$row['firstname'].' '.$row['lastname'].' Har
             foreslått en ny kupong i '.$row['name'].'',
          )
        );
        $response = $request->execute();
    }
    mysqli_free_result($result);
}
```

Kodesnutt 6 viser hvordan vi henter ut id til hver bruker i et spillelag for å sende en varsel til hver bruker ved opprettelse av kupong i et spillelag. For at vi skulle få muligheten til å gi brukeren varsler på Facebook måtte vi aktivere Canvas i Facebook. Dette betyr kort fortalt at når brukeren klikker på varselet i Figur 29 åpner applikasjonen vår seg i Facebook-vinduet. Selve nettsiden fungerer på akkurat samme måte. Figur 30 viser hvordan applikasjonen ser ut på Facebook.

Figur 29 Varsel på Facebook



Figur 30 Ntsosial på Facebook Canvas

5.3.4. Utfordringer med Facebook-integrasjonen

1. Lagring av brukerens ID:

For å gjøre det lettere å hente ut brukerens ID og gjøre den tilgjenglig for flere filer så vi på to forskjellige lagringsmetoder, cookies (informasjonskapsler) og session.

Vi kom frem til at cookies var enkle å lagre og enkle å hente, men at dataene er lett å manipulere og mer basert for å lagre ting på en PC enn for en bruker. Sessions slettes når brukeren lukker vinduet og gjør det mye vanskligere å manipulere data. Det er også ganske lett å lagre og hente session vha. PHP, valget falt dermed på denne metoden.

2. Videresending til login.php hvis man oppdatererte før siden ble lastet:

Et problem som frustrerte oss mye var at dersom man klikket på en lenke eller oppdaterte siden før den hadde lastet helt ble man sendt til login.php igjen. Etter endel feilsøking og nye forumtråder på Stack Overflow fant vi ut at dette var fordi vår FB JavaScript SDK ikke fikk lagret `fbSession` korrekt ved oppdatering/navigering før siden var lastet. Løsningen ble da å bruke `longLivedAccessToken`, som er en utvidet tillatelse fra Facebook, og lagre denne i session.

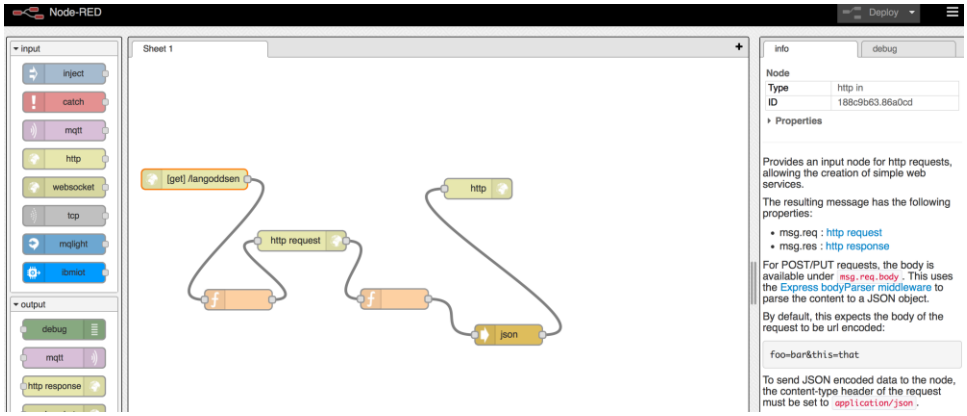
5.4. API

5.4.1. Henting av kamper fra Norsk Tipping

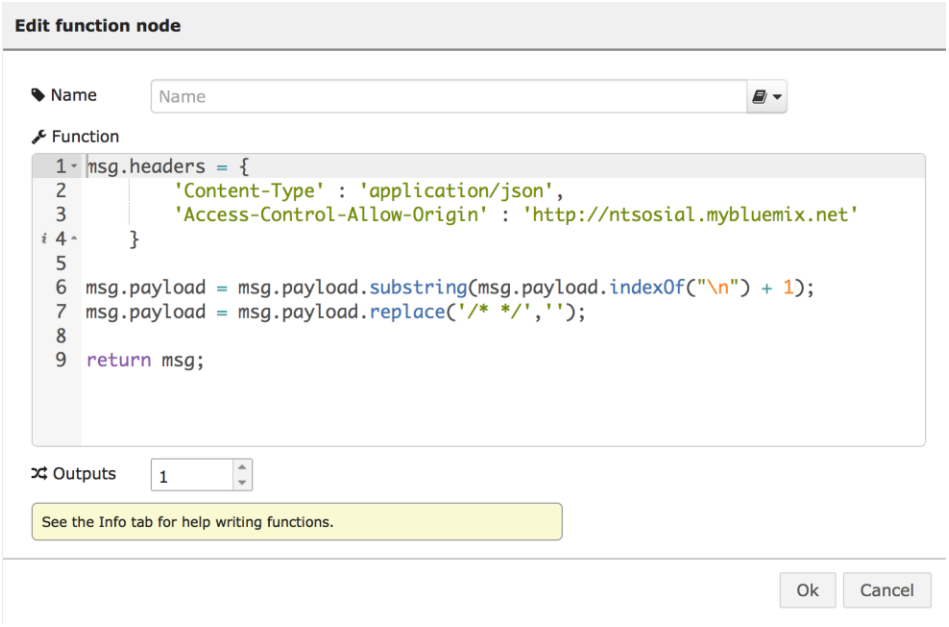
Norsk Tipping sitt API har utrolig mange funksjoner, noen eldre, noen nye og noen som er under utvikling. På grunn av blandingen av nytt og gammelt er det noen tjenester som leverer XML og noen som leverer en form for JSON. Ved hjelp av disse kan man hente ut resultat fra alle de ulike spillene og spill som kan spilles på. APIet er designet for internt bruk og det førte til noen komplikasjoner på grunn av sikkerhet som vi kommer nærmere innpå.

Informasjon om tjenesten `getGameInformation()` i APIet var gitt til oss tidlig i prosessen (11). Vi hadde egentlig tenkt å kalle denne direkte ved hjelp av jQuery, men vi hadde en del problemer med Access-Control-Allow-Origin som er en sikkerhetsmekanisme (CORS) mange nettsider og APIer bruker i dag. Denne eksisterer for å hindre «Cross Side Scripting», som er en type skadelig scripting som kan kjøres i webapplikasjoner. Sluttbrukerens nettleser kan ikke på noen måte vite hvor scriptet kommer fra og det kan bli kjørt uten noen form for godkjennelse som gjør at scriptet kan få tak i sensitiv informasjon om brukeren. (12)

Vi prøvde å sette denne headeren direkte i jQuery koden, men dette fikk vi problemer med fordi dette er en sikkerhetssjekk på server-side. Istedenfor å bruke mye tid på det gikk vi inn i Bluemix og opprettet et nytt API ved hjelp av node.RED(Figur 31). Her fikk vi satt CORS headeren og det virket med en gang. Resultatet fra `getGameInformation` var ikke et verifisert JSON-objekt fra starten av fordi første og siste linje var to utkommenterte linjer. Derfor trengte vi å fjerne den første og siste linjen av resultatet og det gjorde vi også i APIet vi lagde (Figur 32). Når dette var gjort var det klart for å ta imot live kamper.



Figur 31 Viser node.RED grensesnittet og hvordan vårt API er satt opp.



Figur 32 Viser hvordan funksjon-noden er satt opp for å manipulere resultatet fra getGameInformation.

Strukturen av resultatet vi fikk var tungvint å jobbe med og deler av grunnen til det er nok at det ikke er designet for tredjepartsbruk, men det var ingen som kunne gi oss et konkret svar på det. Etter samtale med Håvard kom det fram at web APIene har mye fra en gammel kodebase og at grunntjenesten (på tjenestelaget til Norsk-Tipping) er gammel og produserer XML. Tjenesten som brukes på web inkluderer denne tjenesten. Det medfører at alle endringer som foregår i grunntjenesten må gjenspeiles i web tjenesten. Denne jobben er veldig tidkrevende og det kan være noe av grunnen til at det brukes indexer og ikke nøkler. Et kodeeksempel på å hente ut odds ser for eksempel slik ut:

`data.events[key].eventDetails[6][0][3][2][0]`. Hadde strukturen vært god (Kodesnutt 7) kunne man for eksempel hentet ut oddsen slik: `for(var event in data.events) event.oddsAway`.

Det må nevnes at dette er basert på vår forståelse av dokumentasjonen til APIet og erfaring med bruk av APIer. Vi kunne fått et møte med de som lagde APIet hadde vi ønsket dette, men siden vi fikk satt opp en løsning, som fungerte for oss, raskt fokuserte vi heller på å utvikle annen funksjonalitet.

Kodesnutt 7 Forslag for hvordan event kunne sett ut

```
events: [{
    "id": 3213,
    "matchName": "Lag 1 - Lag 2",
    "matchDate": "01.04.2015",
    "oddsHome": 1.33,
    "oddsAway": 2.55
}]
```

5.5. Egenutviklet kode

Her vil vi gå gjennom det vi mener er de viktigste bitene av koden vi har skrevet.

5.5.1. Tilkobling til databasen

I alle filer der vi aksesserer databasen krever vi at connect.php kjøres. Denne koden er vist i Kodesnutt 8.

Kodesnutt 8 Kode for databasetilkobling

```
<?php

$servername = "x";
$username = "x";
$password = "x";
$dbname = "x";

$conn = new mysqli($servername, $username, $password, $dbname);

// Check connection
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}

$conn->set_charset("utf8");

?>
```

5.5.2. Registrering/brukersjekk

Det aller første vi kodet var sjekke om brukeren finnes i databasen. Dersom dette ikke er tilfelle må vi opprette brukeren.

I Kodesnutt 9 undersøker vi om brukeren finnes. Dersom brukeren ikke finnes opprettes det en rad i databasen som vist i Kodesnutt 10.

Kodesnutt 9 Brukersjekk

```
<?php
require 'connect.php';

$id = $_POST['id'];

// See if user exists
$sql = "SELECT * FROM user WHERE id ='$id'";

if ($result = mysqli_query($conn, $sql)) {
    $rowcount = mysqli_num_rows($result);
    echo $rowcount;

    mysqli_free_result($result);
}


$conn->close();
?>
```

Kodesnutt 10 Opprettelse av bruker i databasen

```
<?php
session_start();
require 'connect.php';

$id = $_POST['id'];
$fn = $_POST['fn'];
$ln = $_POST['ln'];

// Setter inn brukerdata
$sql = "INSERT INTO user (id, firstname, lastname, epost, poeng)
VALUES ('$id', '$fn', '$ln', 'NULL', '500')";

if ($conn->query($sql) === TRUE) {
    echo "New record created successfully";
} else {
    echo "Error: " . $sql . "<br>" . $conn->error;
}

$conn->close();
?>
```

5.5.3. Foreslå nytt spill

5.5.3.1. Prediksjon

For å gi brukeren forslag til kamper ønsket vi å bruke prediksjon. Vi tenkte først å løse dette ved å bruke IBM Watson, som er kapabel til å se sammenhenger og gjøre kognitive analyser på svært store datasett, slik at vi da kunne gi så nøyaktige spillanbefalinger som mulig. Dette ble ikke satt i gang før sent i prosessen, og grunnet dette ble det et tidspress som gjorde det vanskelig å sette seg inn i og bruke Watson. Når vi forstod at vi sannsynligvis ikke ville få tid til det så vi på andre muligheter for prediksjon sammen med Håvard fra Norsk Tipping.

Etter litt diskusjon kom vi frem til at vi kunne løse dette ved å sammenligne våre data mot Norsk Tipping sine ekte spilldata. Dette ville gi oss et godt prediksjonsgrunnlag da disse dataene inneholder 1 000 000 rader med informasjon om brukere og lagene disse har spilt på. Måten dette ble løst på var å lage et API som tar i mot en brukers mest spilte lag og returnerer forslag til andre aktuelle lag basert på koblingene i radene. Dette ble løst ved hjelp av en SQL-setning som tar utgangspunkt i brukerens mest spilte lag, finner alle andre brukere som har spilt på de samme lagene og lister alle andre lag som disse brukerne har spilt på, rangert etter popularitet. SQL setningen, som vist i Kodesnutt 11, blir generert utifra koden og returnerer et resultat som Figur 33 viser. Fra denne tabellen leser vi at av alle spillere i databasen som har spilt på Barcelona, Valencia, Sunderland, Schalke 04 og Odd, har 6851 av de spilt på Chelsea og 4078 av de spilt på Atletico Madrid. Dette gir brukeren forslag til spill som kan være interessante.

Dette er på langt nært et kognitivt prediksjonssystem som IBM Watson, men når vi har et så stort datasett vil den relative frekvensen (Figur 34 og

Figur 35) si noe om hvor sannsynlig det er at en bruker vil være interessert i å spille på det laget.

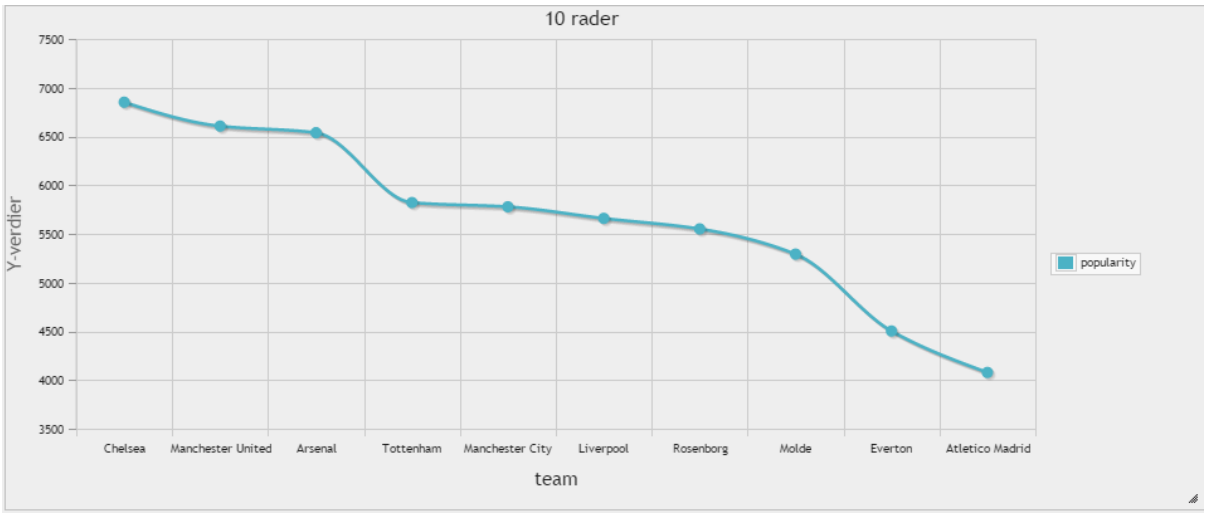
Kodesnutt 11 SQL spørring for prediksjon av lag

```
SELECT team, COUNT(*) AS popularity FROM teams
WHERE id IN (SELECT DISTINCT id FROM teams WHERE team = "Barcelona")
AND id IN (SELECT DISTINCT id FROM teams WHERE team = "Valencia")
AND id IN (SELECT DISTINCT id FROM teams WHERE team = "Sunderland")
AND id IN (SELECT DISTINCT id FROM teams WHERE team = "Schalke 04")
AND id IN (SELECT DISTINCT id FROM teams WHERE team = "Odd")
AND comp IN (SELECT DISTINCT comp FROM teams WHERE team IN("Barcelona", "Valencia", "Sunderland",
"Schalke 04", "Odd"))
```

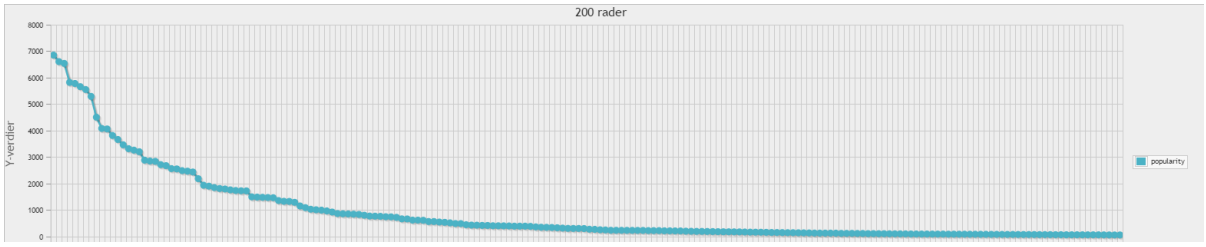
```
AND team NOT IN (SELECT DISTINCT team from teams WHERE team IN("Barcelona", "Valencia",
"Sunderland", "Schalke 04", "Odd"))
GROUP BY team ORDER BY popularity DESC LIMIT 10
```

team	popularity
Chelsea	6851
Manchester United	6606
Arsenal	6538
Tottenham	5821
Manchester City	5778
Liverpool	5659
Rosenborg	5551
Molde	5292
Everton	4503
Atletico Madrid	4078

Figur 33 Resultat fra prediksjonsspørring



Figur 34 Diagram for frekvensen av de ti største lagene



Figur 35 Diagram for hvordan frekvensen av lagene jevnes ut ved 200 lag

Dersom det basert på denne spørringen finnes noen tilsvarende lag returneres disse som JSON-objekt. Om disse lagene har kamper i oddsprogrammet til Norsk Tipping vises disse til

høyre som oddstips og brukeren kan velge å spille på disse lagene. Hvis et foreslått lag ikke har en kamp i programmet hentes neste lag fra JSON-objektet og det repeteres til det er fem foreslåtte lag.

5.5.3.2. Henting av tilgjenglige kamper og innsending av kupong

For å foreslå nytt spill må brukeren velge hvilket spill. I vår prototype er det som nevnt kun LangOddsen som er tilgjengelig. Siden brukeren blir sendt til er navngitt nykupong.php og det er her all registrering av kamper og innsending av kupong foregår. Det første vi begynte å jobbe med var representering av kampene (Figur 12) og hvordan hente de.

Først lagde vi et statisk JSON-objekt basert på APIet til Norsk-Tipping og prøvde å implementere deres representasjon av oddsen ved hjelp av deres stylesheets og fonter. Det ble brukt en del timer på det, men ble siden skrotet når vi fant ut at designet deres ikke var så responsivt som vi ønsket. (Norsk Tipping har ett for web og ett for mobil).

Utvalget av hvilke objekter man kan spille på var ikke viktig for prototypen og derfor bestemte vi oss for å bare ta fotballkamper med muligheter for å tippe hjemme, uavgjort eller borte fordi da kunne vi raskere komme oss videre med funksjonalitet. Funksjonen `updateGameInformation(data, oddsTips, favorittlag)` tar imot et JSON objekt og to arrays, og representerer de kampene vi ønsker. Den populerer også boksene til høyre (Figur 15) som inneholder oddstips og favorittlag. Inne i denne funksjonen er det en enkel if-setning som sjekker om det spesifikke event er av id 70, som er fotball, for å fortsette å appende kamper til siden. Etter studering av formatet på LangOddsen sin side på nett kopierte vi en del av informasjonen de lagret i knappene sine (H,U,B). Knappene i boksene for oddstips og favorittlag er synkronisert med knappene i tabellen. Det vil si at velger bruker en kamp under oddstips-boksen så velger bruker også denne kampen i tabellen. Slik unngår vi at like kamper kan spilles to ganger. I Kodesnutt 12 ser man hvordan knappene endte opp med å se ut.

Kodesnutt 12 Slik ser en H,U eller B knapp ut når den er ferdig generert

```
<a id="985328,HUB-
fulltid,0,1,2,3,H,U,B,170,315,365,false,false,false,,0,2,10,,1430315700000_1"
title="H (1.7)" href="#" class="btn btn-default langoddsenGainButton
betId985328,HUB-
fulltid,0,1,2,3,H,U,B,170,315,365,false,false,false,,0,2,10,,1430315700000
eventId459478 competitionId101  normal">
    <span class="badge pull-left">H</span>
    <span class="prefix ellipsis"></span>
    <span class="value ellipsis">1.7 </span>
</a>
```

Mesteparten her er autogenerert informasjon fra APIet og det ser ikke særlig pent ut grunnet strukturen til resultatet (Kodesnutt 13). Informasjon som vi bruker er:

- Id-en, i dette tilfelle 985328 og blir gjentatt under betId985328.
- _1/_2/_3 sier noe om det blir satset på hjemme, uavgjort eller borte.
- Oddsen, som her er representert som heltallene 170, 315 og 365 finner vi mellom H,U,B og false,false,false under id. Den aktuelle oddsen finner vi når vi deler disse tallene på hundre.
- langoddsenGainButton er referering til en knapp som behandler dataene vi trenger når en bruker trykker på H, U eller B.
- eventId, i dette tilfelle 459478, bruker vi for å kunne lagre gameEventId i databasen (Figur 22) slik at den aktuelle kampen (eventet) kan spores hos Norsk-Tipping
- competitionId, i dette tilfelle 101, bruker vi for å sortere kampene. CompetitionId1 er Tippeligaen og competitionId5 er Premier League for eksempel.

Kodesnutt 13 Eventobjekt fra getGameInformation()

```
[
    70,
    31,
    ["VPS Vaasa - IFK Mariehamn","VPS Vaasa - IFK Mariehamn","VPS-Marieha"],
    [],
    1430321400000,
    [2246,5058],
    [
        [
        981820,
        "HUB-fulltid",
        54,
        [
            [1,2,3],
            ["H","U","B"],
```

```
            [195,310,300],
            [false,false,false],
            []
        ],
        "0",
        2,
        10,
        [],
        1430321100000
        ]
    ]
]
```

En løkke går så igjennom alle kampene som er aktuelle og genererer en for hver kamp med klokkeslett, kampnavn og H,U,B knappene (Figur 15). Klassen `langoddsenGainButton` blir som nevnt lagt til i hver av H,U,B knappene og det gjør at hver gang en av disse knappene blir klikket kjører funksjonen vist i Kodesnutt 14.

Kodesnutt 14 Selector for langoddsenGainButton

```
$(".langoddsenGainButton").click(function()..)
```

Det første som skjer i denne funksjonen er oppretting av objektet detaljer som senere skal legges til i arrayet til den globale variabelen kupong for å til slutt bli sendt inn. Etterpå er det to linjer (Kodesnutt 15) som aktiverer btn-primary klassen hos den brukte knappen og eventuelt fjerner btn-primary hos søskenene til den knappen (U og B er søsken til H for eksempel).

Kodesnutt 15 Aktiverer btn-primary klassen og fjerner den fra søsken.

```
$(this).siblings().removeClass("btn-primary");
$(this).toggleClass("btn-primary");
```

Dette fører til at kun en knapp vil være markert som valgt i gangen. Så hentes eventId ut ifra den markerte knappen sin klasse som nevnt tidligere og det brukes til å hente ut navnet på eventet ved å søke gjennom resultatet fra Norsk-Tipping. Basert på om brukeren klikket H,U eller B blir `detaljer.hub` variabelen satt og ved hjelp av den hentes oddsen ut fra resultatet (Kodesnutt 16).

Kodesnutt 16 Henter ut odds basert på H,U eller B.

```
switch(detaljer.hub){
    case 'H': detaljer['odds'] = data.events[key].eventDetails[6][0][3][2][0];
```

```
    case 'U': detaljer['odds'] = data.events[key].eventDetails[6][0][3][2][1];
    case 'B': detaljer['odds'] = data.events[key].eventDetails[6][0][3][2][2];
}
```

Kodesnutt 17 Struktur på et detaljerobjekt

```
var detaljer{

    eventId: 1234,
    eventName: "TestLag1 – TestLag2",
    hub: "H",
    odds: 170
}
```

Videre kommer det tre sjekker hvor den første (Kodesnutt 18) sjekker om det i det hele tatt fins noen detaljerobjekt (Kodesnutt 17) i arrayet til kupong. Om det ikke finnes noen blir objektet lagt til i kupong og det kommer en varsel om at det er blitt lagt til ved hjelp av alertifyJS.

Kodesnutt 18 Legger til et detaljerobjekt om det ikke finnes.

```
if (kupong.detaljer.length === 0) {
    kupong.detaljer.push(detaljer);
    alertify.set('notifier','position', 'top-right');
    alertify.success('La til: '+ detaljer.hub + ", " + detaljer.eventName);
}
```

Den andre (Kodesnutt 19) sjekker om knappen er markert og hvis den ikke det vil det si at spillet er blitt fjernet fra brukeren og vi er nødt å fjerne det fra den globale variabelen kupong (Kodesnutt 24). Da søker vi gjennom alle detaljerobjektene i kupong og finner detaljeobjektet med matchende eventId og fjerner det. Så brukes alertifyJs til å varsle brukeren om at kampen er blitt fjernet.

Kodesnutt 19 Fjerner kampen fra kupong om ingen H,U eller B er valgt.

```
else if(!$(this).hasClass("btn-primary")) {
    $.each( kupong.detaljer, function( key, value ) {
        if(value !== undefined && value.eventId === detaljer.eventId ) {
            console.log("None selected for this event! Deleting..");
            alertify.set('notifier','position', 'top-right');
            alertify.error('Fjernet: ' + detaljer.eventName);
            itemtoRemove = value;
            kupong.detaljer.splice($.inArray(itemtoRemove, kupong.detaljer),1);
        }
    });
}
```

Den siste (Kodesnutt 20) sjekken søker først gjennom kupong sitt detaljer-array og ser om det allerede finnes kamper med samme eventId og om det gjør det blir detaljerobjektet byttet, hjelpevariabelen replaced blir satt til true og det kommer opp en varsling ved alertifyJS om at kampen er blitt byttet. Om kampen ikke finnes er det en ny kamp og detaljerobjektet blir pushet i arrayet til kupong.

Kodesnutt 20 Bytter endret kamp i kupong eller legger til ny kamp i kupong.

```
else {
    var replaced = false;
    $.each( kupong.detaljer, function( key, value ) {
        if(value.eventId === detaljer.eventId ){
            console.log("EventID equal! Deleting..");
            alertify.set('notifier','position', 'top-right');
            alertify.warning('Byttet: '+detaljer.hub + " med " + value.hub);
            kupong.detaljer[key] = detaljer;
            replaced = true;
        }
    });
    if(!replaced) {
        kupong.detaljer.push(detaljer);
        alertify.set('notifier','position', 'top-right');
        alertify.success('La til: ' + detaljer.hub + ", " + detaljer.eventName);
    }
}
```

I starten viste vi alle kampene med en gang og websiden ble veldig lang så vi følte det var nødvendig å ha med en funksjonalitet som gjør at brukerne kan velge hvor mange kamper de laster inn, og eventuelt sortere fra hvilken liga bruker vil se kamper fra. Det starter med at kun de seks første kampene blir vist på siden. Deretter kan brukeren trykke på knappen "Last inn flere kamper" og 30 nye kamper blir lastet inn. Den sjekker (Kodesnutt 21) først om størrelsen på listen er nådd og setter verdien på variabelen `oddsListe` for så å vise elementene. Etter det sjekkes det om sorteringsknappene er markert og kamper blir vist eller skjult i henhold til `competitionId`. Sorteringsfunksjonene (Kodesnutt 22) sjekker om knappen er markert og viser eller skjuler basert på `competitionId`.

Kodesnutt 21 Funksjon som kjøres når «Last inn flere kamper»– knappen klikkes.

```
$('#loadMore').click(function () {
    oddsListe = (oddsListe+90 >= $('#appendOdds').size()) ? oddsListe+90 : $('#appendOdds').size();
    // Viser alle elementene til og med oddsListe
    $('#appendOdds div:lt('+oddsListe+')').show();
    if($('#sortOvrige').hasClass("btn-default")) {
```

```
    // Beholder <div> med sorteringsknapper og skjuler resten.
       $('#appendOdds div').not(':last-child').hide();
       $('#appendOdds div').find("a.competitionId1").closest(".list-group").show();
       $('#appendOdds div').find("a.competitionId36").closest(".list-group").show();
       $('#appendOdds div').find("a.competitionId5").closest(".list-group").show();
    }
    if($('#sortPL').hasClass("btn-default"))
        $('#appendOdds div').find("a.competitionId1").closest(".list-group").hide();
    if($('#sortLA').hasClass("btn-default"))
        $('#appendOdds div').find("a.competitionId36").closest(".list-group").hide();
    if($('#sortTP').hasClass("btn-default"))
        $('#appendOdds div').find("a.competitionId5").closest(".list-group").hide();
});
```

Kodesnutt 22 Sorteringsfunksjon som sorterer kamper fra Premier League.

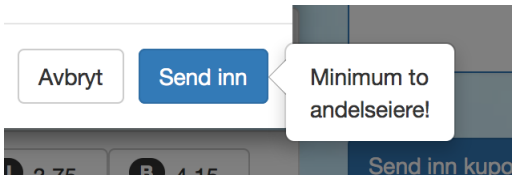
```
$('#sortPL').click(function () {
    if($(this).hasClass("btn-primary")) {
      $('#appendOdds div').find("a.competitionId1")
        .closest(".list-group").hide();
      $(this).removeClass("btn-primary");
      $(this).toggleClass("btn-default");
    }
    else {
      $('#appendOdds div').find("a.competitionId1").closest(".list-group").show();
      $(this).removeClass("btn-default");
      $(this).toggleClass("btn-primary");
    }
});
```

Når send inn kupong klikkes vises det en form (Figur 15). Representasjonen av kampene og oddsen blir hentet fra det globale kupongobjektet (Kodesnutt 24) som brukeren har manipulert. Spillelagene blir hentet via et ajax-kall til getspillelagforkupongform.php, som returnerer navn og id til lagene. Når brukeren klikker på send inn blir det først kjørt en del sjekker (Kodesnutt 23) for å unngå at feil data blir sendt til databasen som gir brukeren tilbakemelding ved en popover funksjon (Figur 36).

Kodesnutt 23 Nettlesersjekk av input

```
$("#sendInn").click(function() {
    // Om det ikke er lagt inn noen kamper
    if(kupong.detaljer.length <= 0) {
        $(this).attr('data-content', 'Ingen kamper lagt inn!');
    }
    // Det må være flere enn 1 andelseier
    else if($("#andelseiere").spinner("value") <= 1) {
        $(this).attr('data-content', 'Minimum to andelseiere!');
    }
    // Det er maks 10 andelseiere per kupong
    else if($("#andelseiere").spinner("value") >= 10) {
        $(this).attr('data-content', 'Maks 10 andelseiere!');
    }
```

```
// Innsatsen må være større enn null
else if($("#kupongverdi").spinner("value") <= 0) {
    $(this).attr('data-content', 'Innsats må være større enn null!');
}
// Maks 5000 i innsats
else if($("#kupongverdi").spinner("value") > 5000) {
    $(this).attr('data-content', 'Innsats for høy!');
}
// Om bruker har huket av for ”tilskuer” sjekkes
// poengene til brukeren mot innsats per andel i kupongen.
else if (($('#tilskuer').prop('checked') == true) &&
    (($("#kupongverdi").spinner("value")/$("#andelseiere").spinner("value"))
    > <?php echo $poeng; ?>)) {

    $(this).attr('data-content', 'Ikke nok poeng!');
}
else {…
```



Figur 36 Pop-over varsling ved feil data

Kodesnutt 24 Den globale variabelen kupong.

```
kupong.ant_deltakere = $("#andelseiere").spinner("value");
kupong.spillelags_id = $("#leverspill").val();
kupong.wager = $("#kupongverdi").spinner("value");
kupong.oppretter_id = '<?php echo $userID ?>';
kupong.betalt = ($('#tilskuer').prop('checked')) ? 0 : 1;
// I tillegg er det et array med alle kampene.
// (Kupong.detaljer[detalj1, detalj2..])

var jsonKupong = JSON.stringify(kupong); // Omgjort til lesbar JSON string.
```

Når alle data er korrekte blir resten av den globale variabelen kupong (Kodesnutt 24) sine attributter satt og kupongen blir omgjort til lesbar JSON-string. Deretter blir det kjørt et ajax-kall som POSTer `jsonKupong` til addnykupong.php (Kodesnutt 25). Først blir den totale oddsen til kupongen regnet ut og deretter kjøres det en ny sjekk hvis brukeren som sendte inn kupongen ønsker å være deltaker. En funksjon henter poengene til bruker og sammenligner med hvor mye som skal satses. Om brukeren ikke har nok poeng blir hele operasjonen avbrutt. Det opprettes så en ny `kupong` i databasen og en ny `spill_detaljer` for hver kamp som er i kupongen. Til slutt blir brukeren satt i relasjon med kupongen. Om dette ikke returnerer noen errors blir callbacket til ajax-kallet suksess. Ved suksess får

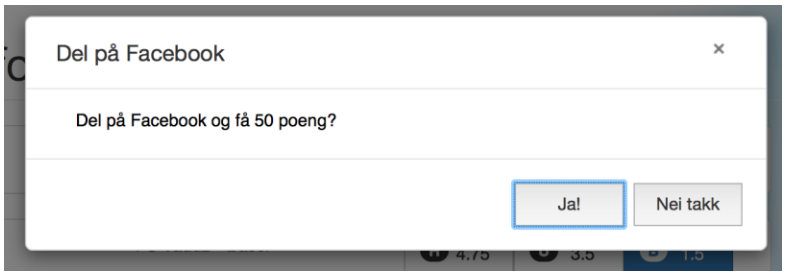
brukeren opp en ny pop-up (Figur 37) ved hjelp av alertifyJS sin confirm-funksjon (Kodesnutt 26). Her får brukeren mulighet til å dele kupongen på Facebook og få 50 ekstra poeng.

Kodesnutt 25 Koden i addnykupong.php.

```
<?php
require 'connect.php';
require 'pointshandler.php';

$data = json_decode(stripslashes($_POST['data']));
$totalOdds = 1;

  foreach($data->detaljer as $d){
      $totalOdds *= (($d->odds)/100);
  }

if ($data->betalt == 0) {
    $poeng = checkPoeng($data->oppretter_id); // Returnerer brukers poeng
    // Hvor mange poeng det skal satses
    $minuspoeng = ($data->wager)/($data->ant_deltakere);

    if ($poeng >= $minuspoeng) {
        removePoeng($minuspoeng, $data->oppretter_id);
    }
    else die();
}


// Oppretter først en kupong
$sql = "INSERT INTO kupong (id, wager, multiplier, numDraws, spill_id,
spillelags_id, ant_deltakere, submitted, oppretter_id, sattDato, totalOdds) VALUES
(NULL, '". $data->wager ."', '1', '1', '12', '". $data->spillelags_id ."', '" .
$data->ant_deltakere ."', '0', '" . $data->oppretter_id ."', NOW(), '" .
($totalOdds*100) ."')";

if ($conn->query($sql) === TRUE) {
    $kupongId = $conn->insert_id;       // Returnerer ID tilbake
    foreach($data->detaljer as $d){     // Ny spill_detaljer

    $sql2 = "INSERT INTO spill_detaljer (id, kupong_id, odds, hub, gameEventId,
kampnavn) VALUES (NULL, '". $kupongId ."', '" . $d->odds ."', '" . $d->hub ."', '"
. $d->eventId ."', '" . $d->eventName ."')";
        $conn->query($sql2);
    }

    // Bruker blir satt i relasjon med kupongen
    $sql3 = "INSERT INTO kupongdeltakere (id, kupong_id, user_id, betalt) VALUES
(NULL, ". $kupongId .", ". $data->oppretter_id .",". $data->betalt ." )";
    $conn->query($sql3);

} else {
    echo "Error: " . $sql . "<br>" . $conn->error;
}

$conn->close();
?>
```

Kodesnutt 26 Alerify sin confirm funksjon

```
alertify.confirm("Del på Facebook","Del på Facebook og få 50 poeng?", function() {

    $.ajax({
      url: '../db/pointshandler.php',
      type: 'POST',
      data: 'fbshare='+'<?php echo $userID; ?>',
      success: function(data) {
      },
      error: function(e) {
      }
    });

    var url = window.location.href;
    url = url.substring(0, url.indexOf('nyKupong'));
    url = url + "visSpillelag.php?id=" + kupong.spillelags_id;
    FB.api(
        "/me/feed",
        "POST",
        {
            "message": "Jeg foreslo en kupong av verdi " + kupong.wager + " kr.",
            "link": url,
        },
        function (response) {
            if (response && !response.error) {
            /* handle the result */
            console.log("LA TIL");
            alertify.set('notifier','position', 'top-left');
            alertify.message('Postet til Facebook.<br>50 poeng lagt til!');
            }
            else console.log(response);
        }
    );
    //POST TO FACEBOOK END

    },function(){
        window.location.href = "visSpillelag.php?id=" + kupong.spillelags_id;
    }).setting('labels',{'ok':'Ja!', 'cancel': 'Nei takk'});
```



Figur 37 AlertifyJS sin "Confirm"-funksjon

5.5.4. Vis spillelag

5.5.4.1. Brukersjekk

Det er viktig at man er medlem av spillelaget man går inn på, derfor har vi en sjekk i denne filen (Kodesnutt 27) som hindrer brukeren og aksessere dersom han/hun ikke er medlem. Denne sjekken blir utført av en SQL-spørring til databasen og returnerer enten en eller ingen rader.

Kodesnutt 27 Sjekk om brukeren er med i spillelaget

```
require '../db/FbAuth.php';
require '../db/connect.php';

$id = $_POST['id'];          // Spillelagets ID

$sql = "SELECT * FROM spillelag LEFT JOIN spillelag_medlemmer ON spillelag.id =
spillelag_medlemmer.spillelag_id WHERE spillelag_medlemmer.user_id = '$userID' AND
spillelag.id = '$id'";

if ($result = mysqli_query($conn, $sql)) {
    if (mysqli_num_rows($result) == 1) // Bruker er med

    else {
        echo "Feil. Er du medlem av spillelaget?";
        exit();
    }

    mysqli_free_result($result);
}
```

5.5.4.2. Hente detaljer om spillelaget

Etter sjekken hentes detaljer om spillelaget, som navn og opprettelsesdato. Sidens tittel settes også lik spillelagsnavnet. Vi sender en POST-forespørsel med bruker sin id og spillelagets id, og får returnert data om spillelaget (Kodesnutt 28). PHP filen getspillelagdetailed.php kjører en SQL-forespørsel med tilsendt data og returnerer resultatet som et JSON-objekt.

Kodesnutt 28 Henter info om spillelaget

```
// Funksjon: Henter detaljert info om spillelaget
function getSpillelagDetailed() {
    $.ajax({
      url: '../db/getspillelagdetailed.php',
      type: 'POST',
      data: 'id='+'<?php echo $id; ?>'+'&uid='+'<?php echo $userID?>',
      dataType: 'JSON',
      success: function(data) {
        //called when successful
```

```
        if (data != '') {
            var spillelagNavn = data.name;
            var opprettet = data.created;
            var creator_name = data.firstname + ' ' + data.lastname;
            // Setter tittel på siden og fyller opp info-bar med info
            document.title = 'Spillelag - ' + spillelagNavn;
            $('#info-bar').append("<strong>Opprettet:</strong> " + opprettet+
               "<br><strong>Administrator:</strong> " + creator_name);
            $('#spillelagsnavn').append(spillelagNavn);
        }
        else {
            // Error, ingen data.
        }
      },
      error: function(e) {
        //called when there is an error
        //console.log(e.message);
      }
    });
}
```

5.5.4.3. *Hente foreslåtte, aktive og fullførte kuponger*

I HTML-koden vår (der vi ønsker å vise frem kupongen) inkluderer vi php-filen som henter informasjonen fra databasen. Kodesnutt 29 viser hvordan vi bruker php include for å presentere spillelagsmedlemmene og ledertavlen.

Kodesnutt 29 Henter kuponger tilhørende spillelaget

```
<?php
    $_POST['spillelags_id'] = $id;      // Spillelagets ID
    $_POST['user_id'] = $userID;        // Brukers ID
    include ('../db/getkuponger.php');
?>
```

Oppgaven til *getkuponger.php* er å presentere alle kuponger tilhørende spillelaget i en karusell. Derfor inneholder denne fila en del HTML-kode (for å sette opp tabs og populere elementene i karusellen) og PHP-kode for å hente ut antall kuponger, kuponginfo og deltakere.

Kodesnutt 30 viser hvordan datene hentes fra databasen og vises frem. Hver kupong er i en `<div class=«item»>`. Figur 10 viser hvordan dette blir seende ut for brukeren.

Kodesnutt 30 Henter kuponger fra databasen

```
echo '
<div role="tabpanel" class="tab-pane" id="historikk"> <!-- Tab historikk -->
<div id="owl-historikk" class="owl-carousel">      <!-- Karusell historikk -->';
```

```php
// Henter kupong
while ($row = mysqli_fetch_array($h_result, MYSQLI_ASSOC)) {
    $kupongid = $row['id'];
    $innsats = $row['wager'];
    $deltakere = $row['ant_deltakere'];
    $laget_av = $row['firstname'];
    $win = $row['win'];
    $tot_odds = 1;

    // Lager en div for kupongen og legger inn informasjon
    echo '
        <div class="item">
        <h1>Kupong ' . $kupongid . '</h1>
        <h6>Foreslått av ' . $laget_av . '</h6>
        <p>';

    // Henter detaljer fra alle kampene tilhørende kupongen
    $detaljer_sql = "SELECT odds, hub, kampnavn, win
    FROM spill_detaljer WHERE kupong_id = '$kupongid'";
    if ($fdetaljer_result = mysqli_query($conn, $detaljer_sql)) {
        while ($row = mysqli_fetch_array($fdetaljer_result, MYSQLI_ASSOC)) {
            $odds = $row['odds']/100;
            $hub = $row['hub'];
            $kampnavn = $row['kampnavn'];
            $tot_odds *= $odds;
            $vunnet = $row['win'];

            echo '' . $kampnavn . ' (<strong>' . $hub . '</strong>)
            (<strong>' . $odds . '</strong>) ';

            if ($vunnet == '1') echo '<span class="glyphicon glyphicon-ok"
               style="color:green"></span><br>';
            else if ($vunnet == '0') echo '<span class=
               "glyphicon glyphicon-remove" style="color:red"></span><br>';
            else echo '<br>';
        }
        mysqli_free_result($fdetaljer_result);
    }

    // Printer resten av kuponginformasjonen, som innsats og odds
   echo '</p>
    <p>Totalodds: <strong>' . round($tot_odds, 2) . '</strong><br>
    Innskudd pr. andel: <strong>' . round(($innsats/$deltakere), 2) .
    ' kr</strong><br>
    Mulig gevinst: <strong>' . round((($tot_odds*$innsats)/$deltakere), 2) .
    ' kr</strong><br></p>';

    if ($win) echo '<span class="label label-success">GEVINST!</span><br>';
    else echo '<span class="label label-danger">Ingen gevinst</span><br>';

    // Henter bilde og navn på de som er med på kupongen
    $andelseiere_sql = "SELECT k.user_id, u.firstname FROM kupongdeltakere
    AS k LEFT JOIN user AS u ON u.id = k.user_id WHERE kupong_id = '$kupongid'
    AND betalt = '1'";

    if ($andelseiere_result = mysqli_query($conn, $andelseiere_sql)) {
```

```
            echo 'Andelseiere (<strong>' . mysqli_num_rows($andelseiere_result) .
            '</strong>): <br>';

            while ($andelseiere_row = mysqli_fetch_array($andelseiere_result,
            MYSQLI_ASSOC)) {
                echo '<img src="https://graph.facebook.com/' .
                $andelseiere_row['user_id'] . '/picture?
                type=square" class="img-rounded" title="' .
                $andelseiere_row['firstname'] . '"> ';
            }
        }
    echo '</div>';
}

// Skriver melding om det ikke finnes noen historiske kamper
if ($ant_historiske == 0) {
     echo "<p>Ingen historiske kuponger.</p>";
}
mysqli_free_result($h_result);
```

5.5.4.4. Delta på foreslått kupong

På alle foreslåtte kuponger legges det til to knapper med forskjellige parametere. Den ene knappen heter «Delta», og legger til brukeren som betalende. Den andre knappen heter «Se på», og legger til brukeren som tilskuer.

Siden vi ikke kan hente brukerens penger fra Norsk Tipping eller andre tilbydere er det eneste «Delta» knappen gjør å legge til brukeren i kupongdeltakere og si at brukeren har betalt. «Se på» har vi derimot kodet slik at innskuddet til kupongen blir trukket fra brukerens poengsum, som vi ser av Kodesnutt 31.

Kodesnutt 31 Delta som betalende eller tilskuer

```
if ($betalt == 0) {        // Hvis brukeren er tilskuer skal poeng trekkes fra.
    $poeng = checkPoeng($uid);  // Henter brukerens poengsaldo

    // Henter kupoengens verdi og ant. deltakere
    $sql = "SELECT wager, ant_deltakere FROM kupong WHERE id = '$kid'";

    if ($result = mysqli_query($conn, $sql)) {
        $row = mysqli_fetch_array($result, MYSQLI_ASSOC);
        // Sjekker verdien på kupongen
        $minuspoeng = ($row['wager']/$row['ant_deltakere']);

        if ($poeng >= $minuspoeng) {
            removePoeng($minuspoeng, $uid); // Trekker fra poeng
            echo "Du er nå med som tilskuer. ". $minuspoeng .
            " poeng trukket fra.";
        }
        else {
```

```
            die("Ikke nok poeng.");
        }
    }
    mysqli_free_result($result);
}
else echo "Du er nå med som betalende. ";
```

5.5.4.5. *Invitere venner*

Når man inviterer venner til spillelaget genereres det en kode (Kodesnutt 32) som har en lengde på 10 tegn, hentet ut tilfeldig fra det engelske alfabetet og tallene 0-9. Brukeren får opp denne koden, sammen med en Facebook «Send»-knapp, og en mulighet til å sende koden på SMS. Denne invitasjonskoden legges i databasen, der den blir validert og lagt inn i invite-tabellen.

Kodesnutt 32 Generering av invitasjonskode

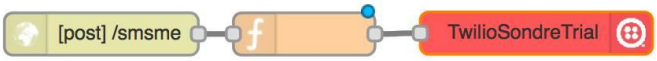
```
// Under: Genering av invitasjon/med kode
$('#inviter').on('show.bs.modal', function (e) {
    var code = generateInviteCode();
    sendInviteCodeToDb(code);
})

function generateInviteCode() {
    // Genererer invitasjonskode
    var code = "";
    var possible = "abcdefghijklmnopqrstuvwxyz0123456789";
    for( var i=0; i < 10; i++ )
        code += possible.charAt(Math.floor(Math.random() * possible.length));

    $('#invite-text').append('<br>Du kan også gi personen(e) denne koden: ' +
    code);

    return code;
}

function sendInviteCodeToDb(invitecode) {
    // Sender invitasjonskoden til databasen
    $.ajax({
      url: '../db/addinvitation.php',     // Oppretter invitasjonskoden
      type: 'POST',
      data: "user_id="+'<?php echo $userID; ?>'+"&spillelag_id="+<?php echo $id;
?>+"&code="+invitecode,
      success: function(data) {
        if (data != '') {                    // Ingen error
            console.log("Kode lagt til i DB. ID: " + data);
        }
        else {
            console.log('FEIL: Invitasjonskode ikke lagt inn i DB.');
        }

      },
```

```
      error: function(e) {
        //called when there is an error
        //console.log(e.message);
      }
    });
}
```

Dersom man velger å sende invitasjonskoden via Facebook eller SMS får mottaker en lenke som vist i Figur 38 og kan enkelt opprette en profil som blir lagt til i spillelaget.

Figur 38 Invitasjonssiden

For å sende SMS måtte vi opprette et API gjennom Node.red på Bluemix. Dette ble gjort ved hjelp av tre noder. En HTTP POST-node, en funksjonsnode og en Twilio-node som vist i Figur 40. Twilio er tilbyderen som sender SMS på våre vegne.



Figur 40 SMS API på Node Red

Det som skjer er at HTTP POST-forespørselen tar i mot teksten som skal sendes og til hvilket nummer på `url/smsme`. Funksjonsnoden sørger for at +47 blir lagt på, deretter tar

Figur 39 SMS med invitasjon

Twilio seg av SMS-utsendingen.

Det eneste vi måtte gjøre var å opprette en prøvekonto hos Twilio og legge inn kontoid og autentiseringstoken i Twilio-noden. Siden vi bruker Twilio sin prøveversjon vil meldinger kun bli sendt til et nummer, men funksjonaliteten for å sende til alle nummer er der om man oppgraderer til en betalings SMS-versjon. Figur 39 viser hvordan en melding gjennom denne tjenesten ser ut.

5.5.5. Resultatsjekk

For å kunne fastslå om kupongene har gått inn måtte vi hente resultatene fra Norsk Tipping (Kodesnutt 33). Disse hentes fra en XML-fil sortert på `gameEventId`. Ved hjelp av Cron kjøres dette skriptet hver time, der det sjekkes om noen av kampene som er ferdigspilt er spilt på i vårt system. Dersom det finnes kamper som er ferdigspilte i vårt system sammenlignes resultatet og kampen oppdateres til `win = 0` eller `win = 1` ved hjelp av sjekkene vist i

Kodesnutt 34. Om alle kampene i en kupong er ferdigspilt (Kodesnutt 35) settes også kupongen til `win = 0` eller `win = 1`, og spillerene og spillelagene får poengene sine.

Kodesnutt 33 Henting av kamper fra NT

```
<?php
$date = date('Y-m-d', time());
$url = "https://services.norsk-
tipping.no/getresults/service/results/Langoddsen/by-date/$date/";

$xmlDoc = new DOMDocument();
$xmlDoc->load($url);

$events = $xmlDoc->getElementsByTagName('Event');
$updated = false;

echo "Sjekker siste kamper.. (" . $date . ")<br>";

// Henter alle fullførte fotballkamper kamper fra Norsk Tippings XML.
foreach ($events as $event) {
    $gameEventId = $event->getElementsByTagName('MatchId')->item(0)->nodeValue;

    if ($event->getElementsByTagName('Item')->item(0)->
        getElementsByTagName('Results')->length != 0) {

        foreach($event->getElementsByTagName('Item')->item(0)->
              getElementsByTagName('Results')->
                item(0)->getElementsByTagName('Result') as $node) {
            if ($node->getAttribute("Name") == "Fulltidsresultat") {
```

```
                $hres = $event->getElementsByTagName('Item')->item(0)->
                  getElementsByTagName('Results')->item(0)->
                  getElementsByTagName('Result')->item(0)->nodeValue;

                $bres = $event->getElementsByTagName('Item')->item(1)->
                  getElementsByTagName('Results')->item(0)->
                  getElementsByTagName('Result')->item(0)->nodeValue;

                if ($hres > $bres)      $res = 'H';
                else if ($bres > $hres) $res = 'B';
                else                    $res = 'U';

                echo $gameEventId . " - " . $res . "<br>";
            }
        }
    }
}
```

Kodesnutt 34 Sjekker opp mot kamper våre brukere har spilt på

```
$sql = "SELECT id, hub FROM spill_detaljer WHERE gameEventId LIKE '$gameEventId'
AND win IS NULL";

    if ($result = mysqli_query($conn, $sql)) {
        if (mysqli_num_rows($result) > 0) {
            // Går gjennom alle og setter win = 1 eller 0
            om kampen har gått inn eller ikke
            while ($row = mysqli_fetch_array($result, MYSQLI_ASSOC)) {
                $hub = $row['hub'];
                $id = $row['id'];

                if ($hub == $res)
                    $sql2 = "UPDATE spill_detaljer SET win=1 WHERE id = '$id'";

                else
                    $sql2 = "UPDATE spill_detaljer SET win=0 WHERE id = '$id'";

                if ($conn->query($sql2) === TRUE) {
                    echo "^Record updated successfully <br>";
                    $updated = true;
                } else {
                    echo "^Error updating record: " . $conn->error . "<br>";
                }
            }
        }
    }
```

Kodesnutt 35 Kupong sjekkes og poeng deles ut

```
if ($finish) {
    echo "Kupong " . $kupong_id . " er ferdig. ";

    // Hvis kupongen er aktiv settes den til win = 1 eller 0
    if ($submitted == 1) {
      if ($no_of_wins === $ant_res) {
```

```
        echo "Alle kamper gikk inn. SQL oppdatert<br>";
        $update_win_sql = "UPDATE kupong SET win=1 WHERE id = '$kupong_id'";


          // Legge til poeng til de som er med på kupongen
          $poengsql = "SELECT user_id, k.wager, k.totalOdds, k.ant_deltakere,
          k.spillelags_id FROM kupongdeltakere AS kd LEFT JOIN kupong AS k
          ON k.id = kd.kupong_id WHERE kupong_id = '$kupong_id'";

          if ($poengresult = mysqli_query($conn, $poengsql)) {
            while ($poengrow = mysqli_fetch_array($poengresult, MYSQLI_ASSOC)){
                $user = $poengrow['user_id'];
                $poeng = ($poengrow['wager']*($poengrow['totalOdds']/100)/
                  $poengrow['ant_deltakere']);
                $sid = $poengrow['spillelags_id'];

                addPoeng($poeng, $user);    // Legger til poeng.
                echo "Gav " . $user . " " . $poeng . " poeng.<br>";

                addPoengTilSpillelag($poeng, $sid); // Legger til poeng.
                echo "Gav spillelag " . $sid . " " . $poeng . " poeng.<br>";
            }
        }

      }
        else {
            echo "Kupongen gikk ikke inn. SQL oppdatert<br>";
            $update_win_sql = "UPDATE kupong SET win=0 WHERE id = '$kupong_id'";
        }
    }
    $conn->query($update_win_sql);
}
else echo "Kupong " . $kupong_id . " er ikke ferdig. SQL ikke oppdatert<br>";
mysqli_free_result($update_win_sql);
```

5.5.6. Statistikk

For å gi brukeren tilgang på statistikk opprettet vi en egen side for dette. Vi valgte å vise data for hvilket lag som brukeren er mest aktivt i, hvilke kuponger som har fått gevinst og en vinnerstatistikk over foreslåtte kuponger. Valget falt på disse tre representasjonene fordi vi mente at dette var mest interessant for brukeren. Google Chart ble brukt for disse tre diagrammene noe som gjorde til at det var relativt lite kode å skrive for oss. Det vi trengte å gjøre var å få tak i dataen fra databasen. Dette ble gjort blant annet med funksjonen getkupongerilag.php (Kodesnutt 36) som blir brukt i pai-diagrammet for «Antall kuponger i lag» og linje-diagrammet «Kuponger med gevinst i lag», begge vist i Figur 10. Den henter ut hvor mange kuponger som bruker har vært med i hvert lag, navnet på laget for pai-diagrammet. Gevinst, dato for kupong og om den er vunnet for linje-diagrammet. Alt dette

blir pushet inn i et array som blir konvertert til et JSON-objekt før det blir returnert til ajax-kallet. I javascript-koden blir det iterert på objektet og informasjon som trengs for hvert av diagrammene blir pushet til en egen variabel (Kodesnutt 37) for det respektive diagrammet. Denne variabelen blir sendt med til funksjonen som står for å tegne diagrammet og er som nevnt hentet fra Google sin dokumentasjon.

Kodesnutt 36 Henter kuponger i lag

```
<?php
session_start();
require 'connect.php';

$id = $_POST['user_id'];

// Sjekker om riktig ID sendes med, og om det er en AJAX-request.
if($_SERVER['HTTP_X_REQUESTED_WITH'] == 'XMLHttpRequest' && $id == $_SESSION["fb-
user-id"]);
else die();

// See if user has spillelag
$sql = "SELECT c, name, ROUND((totalOdds/100 * wager)/ant_deltakere, 0) AS
gevinst, sattDato, win
FROM kupong
INNER JOIN (

SELECT COUNT( * ) AS c, name, spillelags_id
FROM kupongdeltakere
INNER JOIN kupong ON kupong.id = kupongdeltakere.kupong_id
INNER JOIN spillelag ON spillelag.id = kupong.spillelags_id
WHERE kupongdeltakere.user_id ='$id'
GROUP BY spillelags_id
)antlag ON antlag.spillelags_id = kupong.spillelags_id
ORDER BY  `antlag`.`name` ASC";

if ($result = mysqli_query($conn, $sql)) {
    // Return the number of rows in result set
    $stat = array();
    while ($row = mysqli_fetch_array($result, MYSQLI_ASSOC)) {
        array_push($stat, array( $row['name'], intval($row['c']),
$row['sattDato'], intval($row['gevinst']), intval($row['win'])));
    }

    $js_stat = json_encode($stat);
    echo $js_stat;
    //print_r($stat);

    // Free result set
    mysqli_free_result($result);
}


$conn->close();
?>
```

Kodesnutt 37 Oppretter array for pai-diagram og linje-diagram

```
success: function(data) {
    JSON.parse(data).forEach(function(item){
        var exists;
        $.each(stat, function(i, el){
            if(el[0] === item[0]){
                exists = true;   // Sjekker for duplikater
            }
        });
        // Om det ikke er duplikat
        if(!exists){
            // Pusher unikt lagnavn(item[0]) og antall kuponger [1]
           // for pai-diagram
            stat.push([item[0], item[1]]);
            // Unike lagnavn lagres i et array
            spillersSpillelagNavn.push(item[0]);

        }
        if(item[4] === 1) {
        // Om kupongen er vunnet pushes lagnavn,
       // dato for kupong og gevinsten. For bruk i linje-diagram
            stat2.push([item[0], item[2], item[3]]);

        }
    });
}
```

5.6. Gjenbrukt og modifisert kode

5.6.1. Owl Carousel

Vi så på mange forskjellige løsninger for å vise frem kupongene på en elegant måte ved hjelp av karuseller. Det vil si at et bestemt antall elementer vises frem og man kan bla gjennom disse for å se mer.

Vi gikk for Owl Carousel fordi vi syntes den virket best på alle enheter uavhengig av skjermoppløsning og fordi den så best ut. Vi bruker denne koden der vi viser frem kuponger som er foreslåtte, aktive eller historiske. For å initialisere karusellen bruker vi denne koden i Kodesnutt 38. Her har vi endret disse til hva vi mente ville gi den beste fremstillingen, basert på innholdet en kupong har. Vi endret også på fargen og kodet vårt eget oppsett for innholdet som vises i elementene.

Kodesnutt 38 Initaliseringskode for Owl

```
var owl = $("#owl-foreslatte");

owl.owlCarousel({
```

```
items : 2, // 2 elementer ved 1000px bredde eller større
itemsDesktop : [1000,2], // 2 elementer mellom 901px og 1000px
itemsDesktopSmall : [900,1], // 1 element mellom 900px og 601px
itemsTablet: [600,1], // 1 element mellom 600px og 0px

});
```

5.6.2. AlertifyJS

Når brukergrensesnittet for nytt langoddsen spill begynte så vi at det kunne vært brukervennlig å ha varslinger til bruker når valg ble tatt. Men å lage slike varslinger som vi ønsket fra bunnen av, krever mye arbeid og etter et par søk på internett dukket AlertifyJS opp (13). Basert på demoene på hjemmesiden bestemte vi oss for å gå for dette biblioteket fordi kodesnuttene var små og enkle å bruke. Av Kodesnutt 39 ser man et eksempel på en success-varsling. Den er grønn (Figur 13) og går automatisk vekk etter fem sekunder eller når den blir klikket på. Biblioteket har muligheter for mer avanserte funksjoner og muligheter for å endre etter egne preferanser. Vi brukte `confirm`, `success`, `error`, `warning` og message funksjonene. Foruten confirm er alle like enkle som Kodesnutt 39.

Kodesnutt 39 Alertify varsel

```
// success notification

// Shorthand for:
// alertify.notify( message, 'success', [wait, callback]);
alertify.success('Success message');
```

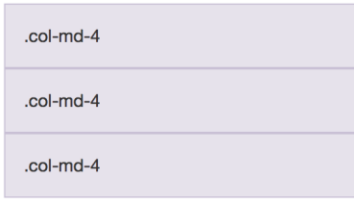
5.6.3. Bootstrap

Bootstrap er et stort bibliotek for å utvikle responsive og pene websider enkelt. Det har en rekke klasser som oppfører seg i henhold til CSS-, JavaScript- og jQuery-koden som ligger i biblioteket. Figur 41 viser et eksempel fra hjemmesiden deres (14). Det viser litt hvordan hierarkiet fungerer og at klassene er avhengige av andre for å ha riktig oppførsel. I dette eksempelet vil ikke nav-tabs fungere responsivt uten nav selv om utseende vil være korrekt. I tillegg har Bootstrap noe som de kaller for grid-layout som gir større frihet til å redigere hvordan utseende skal se ut på skjermer med ulik størrelse. Den baserer seg på at det er tolv grid-elementer til sammen i bredden og så lenge man forholder seg til det ordner bootstrap resten av layouten. Col-klassen blir brukt for å sette elementene som vist i Figur

42 og Figur 43. Koden er bare skrevet en gang og justeres automatisk fra Figur 42 til Figur 43 ved å gjøre nettleservinduet smalere.

Figur 41 Navigasjonsklassens faner i Bootstrap

Figur 42 Utseende på store skjermer



Figur 43 Utseende på små skjermer

5.6.4. Google Charts

For representasjon av data har vi brukt Google-Charts (15). Dette fordi det er et veldokumentert og populært bibliotek som er enkelt å ta i bruk. Det eneste som trengs er å laste inn en <script> tag med kilde til et JavaScript API fra Google. Etter det kan man på nettsiden velge mellom forskjellige diagram, som for eksempel pai- og søylediagram, for å få kodeeksempler på disse og det man må endre på er dataene som blir lastet inn i diagrammet. Alle diagrammene bruker en klasse, `DataTable`, for å populere diagrammet sitt som gjør at det er enkelt å bytte mellom typer diagram.

5.7. Lisensiering

I vår løsning benytter vi oss av flere forskjellige kodebiblioteker. Dette fører til at vi har lisenser å ta hensyn til. Figur 44 viser de bibliotekene vi bruker sammen med lisensieringsinformasjon.

BIBLIOTEK	LISENS
Alertifyjs	MIT (Massachusetts Institute of Technology)
Bootstrap	MIT
Facebook SDK	Apache 2.0
Google Charts	Apache 2.0
jQuery	MIT
JQuery UI	MIT
Owl Carousel	MIT

Figur 44 Oversikt over lisenser i bruk

6. Testing/kvalitetssikring

For å sørge for at løsningen vår samsvarer med kravene fra kravspesifikasjonen, som brukervennlighet, og at all kode oppfører seg som forventet etter de gitte beskrivelsene i Use Casene har vi gjennomført flere tester og kvalitetssikret løsningen vår.

6.1. Kvalitetskontroll

Etter hver nye funksjon vi har implementert har vi testet i nettleseren at ønsket resultat oppstår både med gyldig og ugyldig input. I de tilfellene det har skjedd en feil - som at ønsket resultat ikke oppstår eller at ugyldig input godtas har vi jobbet videre med funksjonen og forbedret den før vi har pushet til Git.

På slutten av oppgaven har vi gjennomført en stor test av løsningen i forskjellige nettlesere med forskjellige oppløsninger og et utvalg mobiltelefoner med iOS og Android-operativsystem. Dette for å sørge for at utseendet justerer seg selv avhengig av hvilken enhet som benyttes.

6.2. Enhetstesting

Vi også brukt rammeverket PHPUnit (16) for testing av deler av koden. Ideelt sett ville vi ha skrevet tester før vi skrev kode, men det ville tatt mye tid og gjort til at vi kanskje ikke hadde fått det resultatet vi sitter med nå. Vi ønsket likevel å utforske noen aspekter ved enhetstesting og valgte å gjøre dette på poenghåndteringsfilen. Her en rekke funksjoner som skal returnere enkle resultat basert på inputs fra programmet. Ingen av oss hadde noe erfaring med enhetstesting fra før og derfor var det mye som var nytt.

Først opprettet vi en egen fil kalt pointshandlertest.php. Her lages en klasse som arver rammeverket og dets funksjoner. Funksjonene vi har brukt er `assertTrue` og `assertEqual` og disse er presentert i Kodesnutt 40. Testen blir kjørt direkte fra terminal som vist i Kodesnutt 41 og det blir vist hvor mye minne og tid som ble brukt, samt hvor mange testfunksjoner som kjøres. Om testen ikke går igjennom får man beskjed om det i terminal som vist i Kodesnutt 41.

Kodesnutt 40 Testkoden skrevet i PHPUnit Framework

```
class PointsTest extends PHPUnit_Framework_TestCase {
    public function testActiveCoupons () {
        $this->assertTrue(getActiveCoupons('10152999217745569') == true);
    }

    public function testCheckPoeng() {
        // Bruker finnes ikke
        $this->assertTrue(checkPoeng('124213') ==  false);
        // Bruker finnes og har 500 poeng
        $this->assertEquals(checkPoeng('10152999217745569'), 500);
    }

     public function testRemovePoeng() {
        // Bruker med slutt på 69 har 500 poeng
        // Bruker med slutt på 13 finnes ikke
        $this->assertTrue(removePoeng(400 ,'124213') == false);
        $this->assertTrue(removePoeng(400 ,'10152999217745569') == true);
        $this->assertTrue(removePoeng(200 ,'10152999217745569') == true);
        $this->assertTrue(removePoeng(99 ,'10152999217745569') == true);

    }
    public function testAddPoeng() {

        $this->assertTrue(addPoeng(200,'123') == false);
        $this->assertTrue(addPoeng(200, '10152999217745569') == true);
    }
}
```

Kodesnutt 41 Tilbakemelding på enhetstesting.

```
// Suksess
Sondres-MacBook-Pro-2:test SondreTJ$ phpunit --verbose pointshandlertest.php
PHPUnit 4.6.6 by Sebastian Bergmann and contributors.
....
Time: 113 ms, Memory: 11.50M
OK (4 tests, 9 assertions)


// Feil
Sondres-MacBook-Pro-2:test SondreTJ$ phpunit --verbose pointshandlertest.php
PHPUnit 4.6.6 by Sebastian Bergmann and contributors.
..F.
Time: 129 ms, Memory: 11.50Mb

There was 1 failure:
1) test::testRemovePoeng
Failed asserting that false is true.
/Applications/XAMPP/xamppfiles/htdocs/prosjekter/test/pointshandlertest.php:24

FAILURES!
Tests: 4, Assertions: 8, Failures: 1.
```

6.3. Brukertester

Ved å utføre brukertester får vi vurdert brukervennligheten til løsningen samtidig som potensielle flaskehalser blir identifisert (17). Vi får også gode innspill til forbedringer, noe som vil være verdifullt for vår oppdragsgiver med tanke på videreføring av prototypen.

Målgruppen for brukertestingen var registrerte brukere på Facebook som tidligere har spilt LangOddsen eller har kjennskap til dette.

For å utføre testingen opprettet vi et undersøkelsesskjema i Google Docs (18). Testene ble utført mens en av oss satt ved siden av og fylte inn brukerens synspunkter i skjemaet. Vi stilte følgende spørsmål:

1. Hvordan synes du registreringsprosessen var?
2. Ble du raskt kjent med de forskjellige funksjonene?
3. Opplever du applikasjonen som responsiv?
4. Opplever du applikasjonen som enkel å forstå?
5. Har du tips til forbedring?

6.3.1. Svar

Svarene vi fikk fra 3 forskjellige brukere var:

1. Enkel. Gjort på et øyeblikk
2. Oversiktlig meny, burde kanskje vært en introguide eller noe sånt.
3. Testet på mobil og det var ingen merkbare problemer.
4. Etter å ha trykt litt rundt så var det greit å forstå.
5. Liste over spillelag å bli med i?

Test utført av: Jan Fredrik - 30. april 2015

1. Optimal
2. Ja
3. Rask
4. Ja
5. Inkludere fleir oddsspill. Typ oddsbomben etc

Test utført av: Sondre - 5. mai 2015

1. Enkel
2. Yes
3. Yes
4. Yes
5. Litt flatt design

Test utført av: Sondre – 7. mai 2015

Oppsummert mener disse brukerene at registreringen med Facebook er enkel og optimal. Bruker 2 og 3 testet på en PC mens bruker 1 testet på en mobil. Responsivheten er godkjent av alle. Forbedringene brukerene ønsket var å ha en oversikt over spillelag å bli medlem av, og å inkludere flere typer spill.

7. Avslutning

7.1. Diskusjon

Det største valget vi tok før utviklingen var hvilke språk vi skulle benytte oss av. PHP og MySQL ble valgt fordi at vi hadde erfaring med disse før og visste hvordan man skulle utvikle og kommunisere mellom de. Underveis har vi funnet mange andre språk som virker mer moderne og kanskje ville gitt oss mer bredde og utfordringer. Eksempler på dette er Node.js, Ruby, MongoDB og RedisDB. Det ville gitt andre utfordringer å brukt dette, men vi vet ikke om vi da ville kommet like langt og fått hjelp når vi hadde stått fast.

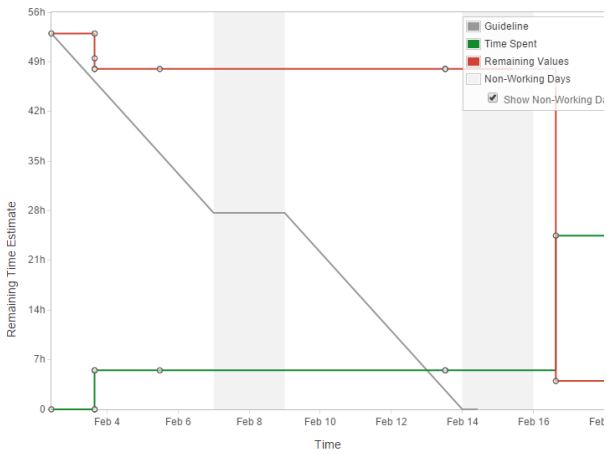
For å holde orden og få oversikt over produktkøen har vi benyttet oss av Agile-delen av JIRA. Dette gav oss og oppdragsgiver muligheten til å se hva som ble gjort og hva som skulle gjøres. Når en oppgave var fullført ble den lagt til i listen over ferdige oppgaver som vist i Figur 45. I starten var dette verktøyet veldig nytt for oss, og vi slurvet med å logge timer noe som førte til at vi fikk dårlige burn-down charts, som vist i Figur 46. Vi skjønte fort konseptet og ble flinkere til å logge timer som man ser i Figur 47.

Vi ble også bedre på å estimere tid på ulike oppgaver. I starten testet vi ut «Planning Poker» (Figur 48) for å estimere tid på oppgavene. Det hadde lite for seg da vi var to, men vi valgte likevel å prøve det. Burndown-chartene (Figur 46 og Figur 47) viser også hvordan vi ble mer treffsikre på estimeringen av oppgavene før hver sprint.

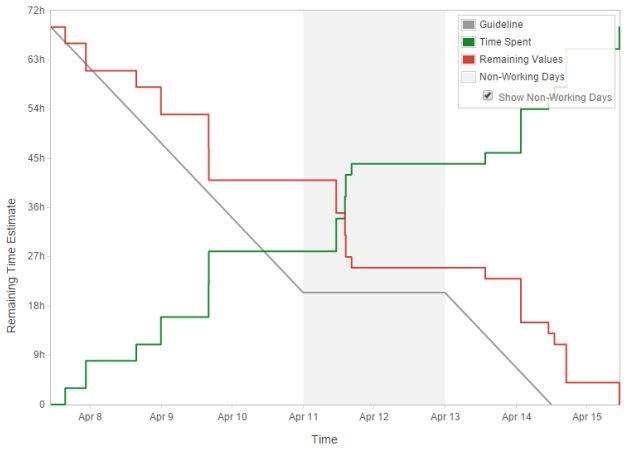
Completed Issues View in Issue Navigator

Key	Summary	Issue Type	Priority	Status	Original Time Estimate (0)
NTSOS-3 *	Invitere venner til spillag	New Feature	Major	DONE	-
NTSOS-20 *	Opprette spillelag	New Feature	Major	DONE	-

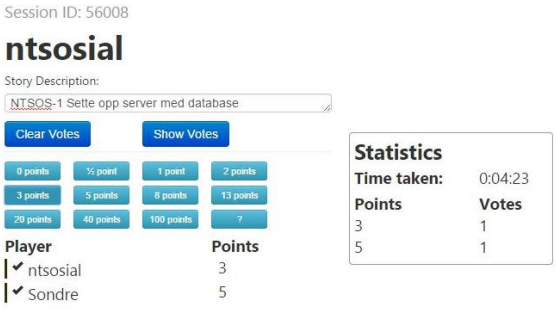
Figur 45 Oversikt over to ferdige oppgaver i JIRA



Figur 46 Burndown Chart (Sprint 1)



Figur 47 Burndown Chart (Sprint 5)



Figur 48 Testing av planning poker (Jan Fredrik sitt alias er ntsosial)

7.2. Resultater

7.2.1. Mål

Før prosjektet startet satt vi oss effektmål, resultatmål og læringsmål. Effektmålene er vanskelig å bekrefte ettersom de tar utgangspunkt i at løsningen finnes hos Norsk Tipping. Konseptet spillelag hos Norsk Tipping ble startet opp igjen, etter nedetid, ved slutten av vår utvikling og vi viste frem vårt produkt for teamet som skal utvikle konseptet. Etter presentasjonen og diskusjon kom det frem at effektmålene (se kapittel 1.6) i stor grad ville vise seg å være korrekte. Dette fordi Norsk Tipping ikke har noen form for aktiviteter på sosiale medier og at de har en utfordring med å rekruttere unge spillere med tanke på videre eksistens av selskapet.

Resultatmålene anser vi som oppnådd da vi har en fullt fungerende prototypeløsning på facebook som er i bruk. På bakgrunn av dette er også merfunksjonaliteten utarbeidet, selv om prosjektet, fra Norsk Tipping sin side, ikke er kommet så langt som først antatt.

Hele oppgaven var en stor læringsprosess som ga oss erfaring i hvordan større prosjekter jobbes med og fungerer. Vi har lært hvordan det er å forholde seg til en oppdragsgiver når man bruker scrum. Viktigheten av versjonskontroll, git, og verktøy for arbeids- og oppgavelogg, JIRA, har vært en erfaring som blir viktig å ta med seg videre. I tillegg til dette har vi også fått større innsyn i hvordan Apier, databaser og responsive web-applikasjoner blir designet og fungerer.

7.2.2. Produktet

I begynnelsen var det vanskelig for oss å forestille seg hvordan produktet ville se ut. Etter møter med Jørn og Håvard, samt utforming av kravspesifikasjon fikk vi en bedre indikator på hva vi kunne forestille oss. Vi har utviklet et produkt som står i samsvar med våre forventninger basert på tid, kunnskap og kravspesifikasjon. SMS-tjenesten (Kapittel 5.5.4) var opprinnelig ikke planlagt og satt som krav, men ble implementert helt i slutten av prosjektet. Ideen kom opp mens vi holdt på med avslutningen av rapporten. Produktet er en prototype og hensikten er at det skal vise en basisfunksjonalitet og en plattform å bygge videre på det enorme potensiale konseptet har. Selv om det er en prototype har vi fått positive tilbakemeldinger fra venner og ansatte hos Norsk Tipping, og at dette er noe flere kunne tenke seg å bruke.

7.3. Gruppeevaluering

Siden vi bare var to på gruppen ble vi enige om å dele ansvaret oss i mellom og skrive relevante grupperegler med konsekvenser i tilfelle noe skulle skje. Vi har begge vært like motiverte under prosjektet og gjort like mye. Siden vi ikke alltid har vært sammen har vi jobbet mye over nett og diskutert over Skype. Alt vi har gjort har blitt loggført i Jira, sammen med antall timer brukt.

Før hver sprint har vi hatt «planning meeting», der vi har fordelt oppgavene oss i mellom slik at arbeidsmengden ble omtrent lik og vi har testet ut «planning poker» for å estimere tid. Det har ikke vært noen komplikasjoner underveis rundt hva vi ville jobbe med, eller andre

konflikter/uoverenstemmelser. Det har så klart vært diskusjoner rundt implementeringen der vi begge har hatt egne tanker og meninger, hvor vi da har kommet frem til en løsning.

Etter hver sprint har vi møtt oppdragsgiver i Hamar eller over Skype og hatt «sprint review» og «retrospective». Dette har gitt oss gode erfaringer med prosjektarbeid og videre inspirasjon i prosjektet.

Jan Fredrik har tatt seg av prosjektets nettside og oppdatert den med nye innlegg etter hver sprint. Dette har fungert bra. Det som kunne ha vært bedre er å hatt mer formelle møter der vi skrev et grundig referat. Vi har ingen formelle referat, men vi har notert ned alle møter og lagret de underveis. Vi følte det var unødig bruk av tid å skrive lange referat da vi fikk med oss det som var aktuelt og tok med oss alle tilbakemeldinger.

7.4. Veien videre

Spillelagskonseptet ble startet opp igjen i april. Under møte den 5. mai 2015 ble løsningen vår vist frem for denne gruppen.

De var klart positive til funksjonaliteten og integrasjonen med Facebook, og at dette er noe de kan se for seg å bruke når de en gang i løpet av 2016 kommer i gang med denne delen av konseptet.

Prioriteten pr. dags dato er å få til en løsning som gjør at spillegrensene ikke påvirkes per person når man spiller i lag, men delt på antall personer i laget. For eksempel hvis tapsgrensen per person er 15.000 kroner, og en person leverer inn for et lag på 10 personer, burde tapsgrensen vært høyere siden det ikke eksisterer noen løsning for å levere inn som lag per dags dato. Dette er deres førsteprioriet, og hovedgrunnen er at de vil ivareta de spillerene som spiller «manuelt» i spillelag. Gjennomsnittsalderen for disse spillerene er såpass høy, kom det fram i møtet, at sannsynlighen for at disse vil innta Facebook/PC for å bruke en slik løsning er lav. Etter dette vil de se på en løsning som kan inneholde aspekter ved vår prototype for langsiktig satsning. Vi påpekte at skulle de kunne være konkuransedyktige i fremtiden er det viktig å rekruttere yngre spillere og møte de der de er – på digitale arenaer, slik som deres konkurrenter allerede gjør.

Vi håper vi har bidratt til inspirasjon som kan hjelpe til med utformingen når den tid kommer.

7.4.1. IBM Watson

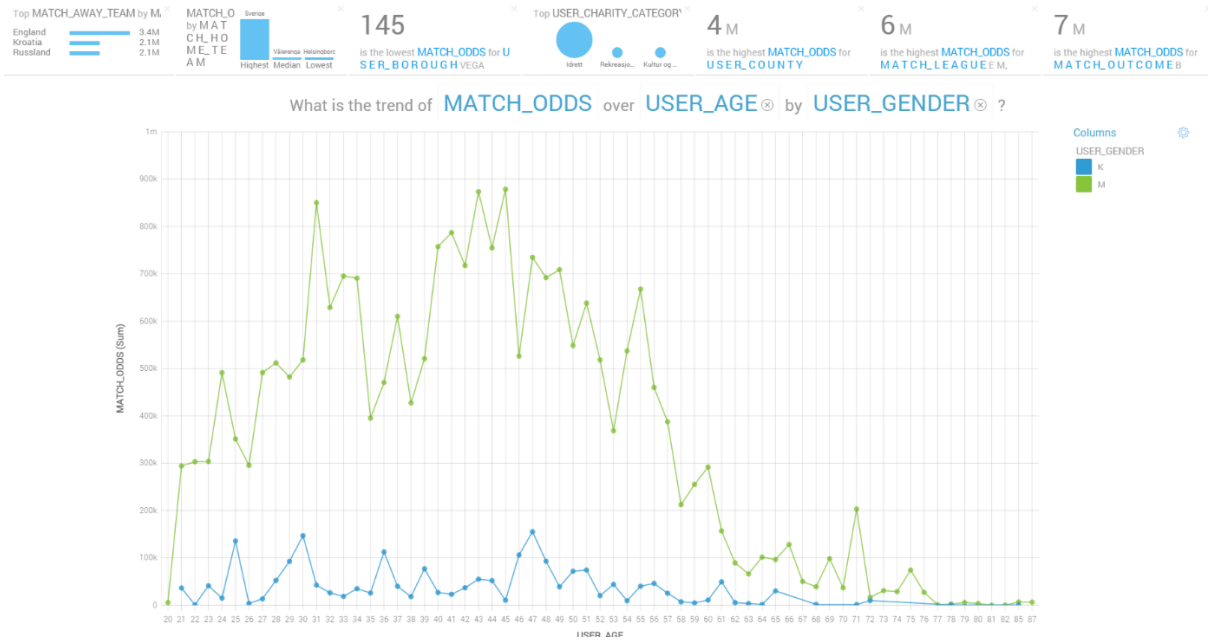
Som nevnt så vi lenge for oss å bruke IBM Watson sine analyser for å gi brukere avansert prediksjonsdata. Dette ble senere skrinlagt på grunn av tid, men også på grunn av avkastningen vi ville fått med vår data. Vi endte med å sette opp en enklere prediksjonsløsning basert på de data vi henter inn fra bruker (se kapittel 5.5.3).
I dette kapittelet vil vi gå inn på de løsningene IBM Watson kunne brukes til, tidlig i prossessen.

Etter fire års utvikling hos IBM ble Watson verdenskjent da supercomputeren vant det amerikanske gameshowet Jeopardy i 2011 (19). Sammen med lanseringen av IBM Bluemix har IBM også gjort store deler av Watson sin kunstige intelligens tilgjengelig for de som betaler for det. I korte trekk er Watson designet for å fungere som en menneskelig hjerne ved å bruke samme kognitive rammeverk som mennesker bruker: observere, tolke, evaluere og velge. Systemet er designet for å kunne ta inn all slags typer data, fra forskningsrapporter til Tweets (20).

Potensialet var stort og vi så for oss å finne relasjoner mellom en brukers alder, kjønn, lokalitet, grasrotmottaker, odds og mest spilte lag. De fire første feltene var ikke nødvendige for at prosjektet skulle fullføres og var bare aktuelle å implementere om det var tid til det, noe det ikke ble.

Mot slutten av prosjektet fikk Norsk Tipping tilgang til Watson og sammen med Håvard satt vi oss ned og testet ut tjenesten i praksis ved hjelp av data fra Norsk Tipping. Vi lastet opp en fil med en million rader og Watson ga oss informasjon om relasjonene i databasen i løpet av kort tid. I Figur 49 ser vi hva alder har å si for hvor mye odds en spiller er villig til å spille på, fordelt på kjønn. Av Figur 50 viser Watson at det kan anslås hvilken grasrotkategori (idrett, kultur og kunst, frivillighet, etc.) en spiller vil ha med 68.2 prosent sikkerhet, basert på alder og kjønn. Som nevnt ble dette testet ut på relativt kort tid og vårt inntrykk er at denne tjenesten kunne hjulpet oss tidligere i prosessen med kartlegging av applikasjonen og

prediksjoner.



Figur 49 Diagram fra Watson

Figur 50 Prediksjonsfigur fra Watson

7.5. Konklusjon

Når utviklingen nå er ferdig er vi godt fornøyde med å ha fått implementert det som ble bestemt i prosjektplanen (Vedlegg F) og enda mer, men vi føler vi at ikke har fått utnyttet vårt fulle potensiale. I ettertid er det vanskelig å definere hvordan vi kunne gjort ting annerledes basert på oppgavens utforming. Etter flere runder med Norsk Tipping har vi utviklet et «proof of concept» og vi tok utgangspunkt i at teknologien hos Norsk Tipping fungerte, fordi de allerede har planer om å utvikle det. Hadde vi ikke fått beskjed om det kunne oppgaven sett helt annerledes ut, og vi kunne fått andre erfaringer med tanke på optimalisering, sikkerhetssjekker ved transaksjoner, dypdykk i algoritmer og andre teknologiske aspekter som ligger på deres kjerneplattform.

Det har vært veldig spennende å få jobbe tett på Norsk Tipping og få innblikk i hvordan deres arbeidshverdag er. Vi har lært masse rundt utvikling og gjennomføring av større prosjekter, og denne erfaringen kommer vi til å ta med oss videre.

8. Litteraturliste

1. Statistikk sosiale medier 2010232. Metronet; [sitert 2015.04.18]; Tilgjengelig fra: https://metronet.no/statistikk-sosiale-medier-2014/.
2. norsk-tipping.no. Om selskapet. [sitert 2015.04.16]; Tilgjengelig fra: https://www.norsk-tipping.no/selskapet/om-norsk-tipping.
3. Difi. Mobile løsninger. [sitert 2015.04.29]; Tilgjengelig fra: http://uu.difi.no/veiledning/nettsider/uu-skolen/mobile-losninger.
4. Bootstrap. Get Bootstrap. [sitert 2015.05.02]; Tilgjengelig fra: getbootstrap.com.
5. Connolly TM, Begg CE. Database systems : a practical approach to design, implementation, and management. 5th ed. ed. Boston, Mass: Addison-Wesley; 2010.
6. Joyent I. NodeJS - About. [sitert 2015.05.04]; Tilgjengelig fra: https://nodejs.org/about/.
7. MySQL. Top Reasons to Use MySQL. [sitert 2015.04.22]; Tilgjengelig fra: https://www.mysql.com/why-mysql/topreasons.html.
8. Tutsplus. PDO vs. MySQLi: Which Should You Use? [sitert 2015.04.30]; Tilgjengelig fra: http://code.tutsplus.com/tutorials/pdo-vs-mysqli-which-should-you-use--net-24059.
9. Facebook. JavaScript SDK. [sitert 2015.04.19]; Tilgjengelig fra: https://developers.facebook.com/docs/javascript.
10. Facebook. PHP SDK. [sitert 2015.04.20]; Tilgjengelig fra: https://developers.facebook.com/docs/reference/php.
11. norsk-tipping.no. Dokumentasjon av Norsk Tippings åpne API. [sitert 2015.04.25]; Tilgjengelig fra: https://www.norsk-tipping.no/common-gui/web-api.jsp.
12. CgiSecurity. The Cross-Site Scripting (XSS) FAQ. [sitert 2015.04.27]; Tilgjengelig fra: http://www.cgisecurity.com/xss-faq.html.
13. AlertiFyJS. AlertifyJS. [sitert 2015.04.16]; Tilgjengelig fra: http://alertifyjs.com/.
14. Bootstrap. Components. [sitert 2015.05.02]; Tilgjengelig fra: http://getbootstrap.com/components/.
15. Google. Using Google Charts. [sitert 2015.04.26]; Tilgjengelig fra: https://google-developers.appspot.com/chart/interactive/docs/index.
16. Bergmann S. PHPUnit - The PHP testing framework. [sitert 2015.05.05]; Tilgjengelig fra: https://phpunit.de/.
17. Bouvet. Brukertesting. [sitert 2015.05.01]; Tilgjengelig fra: http://www.bouvet.no/Kommunikasjon/Brukertesting/.
18. Gundersen JF, Johannessen ST. Brukertest - Sosiale spill på Facebook. [sitert 2015.04.24]; Tilgjengelig fra: https://docs.google.com/forms/d/1AQ0AQNmSgNQKjlHZkzFUT9hsv2WEV2D8sH8WJeZadsw/viewform.
19. ZDNet. IBM's Watson wins Jeopardy practice round: Can humans hang? [sitert 2015.05.10]; Tilgjengelig fra: http://www.zdnet.com/article/ibms-watson-wins-jeopardy-practice-round-can-humans-hang/.
20. IBM. What is Watson. [sitert 2015.05.10]; Tilgjengelig fra: http://www.ibm.com/smarterplanet/us/en/ibmwatson/what-is-watson.html.

9. Vedlegg

A. Terminologiliste

B. Trivielle detaljerte use case-beskrivelser

C. Installasjonsveiledning

D. Oppgaven

E. Kontrakt

F. Prosjektplan

G. Møtelogg

H. Utviklingslogg

I. Statusoppdateringer

Vedlegg A. Terminologiliste

AJAX	Et språk som kan kommunisere med server/database fra klientsiden asynkront.
ALERTIFYJS	Et JavaScript-rammeverk for oppretting av dialogbokser og varsler.
API	Application Programming Interface. Tjeneste man benytter seg av for å hente data som ofte ligger hos ekstern aktør.
BOOTSTRAP	Rammeverk for utvikling av responsive sider i HTML, CSS og JavaScript.
BURNDOWN-CHART	En grafisk fremstilling av oppgaver som gjenstår mot tid som gjenstår.
CLOUD FOUNDRY	En open source PaaS (platform as a service) som gjør det enkelt å utnytte skyen for å hoste applikasjoner.
CORS	En HTTP mekanisme for å filtrere hvem som kan hente ut data.
CRON	Gjør at et skript kjører på en bestemt tid, f.eks hver time.
CSS	Et språk for å definere utseende på en nettside.
EXCEPTION	En uønsket hendelse i PHP.
GET/POST-REQUEST	HTTP forespørsler for sending av data fra klient til server.
GIT	Et versjonskontrollsystem.
GRASROT	Konsept fra Norsk Tipping der spilleren velger en organisasjon som får fem prosent av innsatsen.
HTML	HyperText Markup Language. Et markeringsspråk for formatering av nettsider.
IBM BLUEMIX	En cloud-PaaS for utvikling av desktop-, mobil- og webapplikasjoner.
JAVASCRIPT	Et objekt-orientert programmeringsspråk som blant annet brukes til utvikling av nettsider.
JIRA	Et prosjektstyringsverktøy med støtte for smidig systemutvikling.
JQUERY	Et JavaScript-bibliotek for å forenkle klientskripting av HTML.
JSON	JavaScript Object Notation. En tekstbasert standard for datautveksling.
LANGODDSEN	Et sportsspill fra Norsk Tipping som setter odds på kamputfall.

PHP	Hypertext Preprocessor. Et språk for utvikling av serverside-programvare.
SCRUM	En systemutviklingsmodell for utvikling av programvare.
SPRINT	En iterasjon over en gitt tid. Brukes i Scrum.
SQL	Structured Query Language. Programmeringsspråk for å behandle og manipulere data lagret i en database.
SQL-INJECTION	Å manipulere en SQL-spørring ved å skrive inn SQL-setninger i f.eks et tekstfelt som sendes til databasen.

Vedlegg B. Trivielle detaljerte use case-beskrivelser

Use Case	**Delta på spill (Betalende)**
Scope	Nettsiden
Aktør	Sluttbruker
Pre-betingelser	Foreslåtte kuponger er tilgjengelige
Post-betingelser	Bruker blir lagt til som betalende på spillet
Detaljert hendelsesforløp	1. Foreslåtte spill presenteres 2. Bruker velger å delta som betalende 3. Saldo sjekkes hos NT 4. Beløp reserveres 5. Bruker legges til som betalende
Feilsituasjoner	1) Ingen foreslåtte spill er tilgjengelig 1. Melding vises om at det ikke finnes noen foreslåtte spill. 2. Bruker får en lenke for å opprette et eget spill

Use Case	**Henter kamper**
Scope	Nettsiden, Norsk Tipping API
Aktør	Webserver
Pre-betingelser	Kontakt med Norsk Tippings kamp-API
Post-betingelser	Kampene tilgjenglig for spill vises for brukeren med H,U,B-alternativer
Detaljert hendelsesforløp	1. Kampene tilgjenglig for spill hentes 2. For hver kamp lages det en ny div som inneholder klokkeslett, kampnavn og H/U/B knapper for odds.
Feilsituasjoner	1.1) Ingen kamper blir returnert 1. Melding om «Ingen kamper tilgjengelig vises»

Vedlegg C. Readme.md (Installasjonsguiden)

```
NTSOSIAL
====================
Sosiale spill på Facebook. Bacheloroppgave våren 2015

Krav
-----
PHP 5.4 eller høyere
Apache
MySQL

Før installasjon
-----
1. Opprett en Facebook-applikasjon på developers.facebook.com

Installering
-----
1. Kopier alt i 'ntsosial' til ønsket destinasjon
2. Kjør vedlagte 'create-tables.sql' SQL-kode for å opprette tabellene
3. Endre $servername, $username, $password og $dbname i db/connect.php
4. Endre setDefaultApplication på linje 16 i db/FbAuth.php til dine FB-applikasjonsdetaljer
5. Endre appId på linje 35 i inc/fb.php til dine FB-applikasjonsdetaljer
6. Endre appId på linje 65 i pages/login.php til dine FB-applikasjonsdetaljer
7. Opprett spillelag, inviter venner og spill!

Eksterne biblioteker
-----
Bootstrap (ink. bower m/ font-awesome, jquery, jqueryui)        MIT
Facebook SDK for PHP                                            Apache 2.0
Facebook SDK for JavaScript                                     Apache 2.0
OWL Carousel (http://owlgraphic.com/owlcarousel/)               MIT
AlertifyJS (http://alertifyjs.com/)                             MIT
Google Charts (https://developers.google.com/chart/)            Apache 2.0

Kjente feil
-----


Tidligere feil
-----
Ved navigering eller oppdatering av en side før den har lastet ferdig vil man bli sendt til
autentisering igjen.
Løst ved hjelp av:
http://stackoverflow.com/questions/26074435/facebook-php-js-sdk-this-authorization-code-
has-been-used-error (23.04.15)

Laget av
-----
Jan Fredrik Gundersen
Sondre T. Johannessen
```

D. Oppgaven

Oppdragsgiver:	Norsk Tipping AS
Kontaktperson:	Jørn Berg Nordlund
Adresse:	Måsåbekkvegen 20, 2315 Hamar
Telefon:	930 48 060
Epost:	jorn-berg.nordlund@norsk-tipping.no

Facebook integrering av sosiale spill

Norsk Tipping jobber kontinuerlig for å forbedre kundeopplevelsen både gjennom et godt spilltilbud og ved å møte kundene på deres premisser. Norsk Tipping har ingen ønsker om å konkurrere med sosiale plattformer, men vil tilby løsninger for kundene der de er. Dette innebærer å muliggjøre tettere tilknytning til blant annet sosiale medier.

Norsk Tipping er i konseptfasen med å utvikle en basisløsning på sin plattform for spillelagsfunksjonalitet, som i første fase ikke vil ha tilknytning til sosiale medier. Spillelag er den digitale varianten av «kameratenes» tippelag og «venninnegjengens» lottoklubb.

Oppgaven

Utvikle en løsning på Facebook som gjør spillelagsfunksjonaliteten til Norsk Tipping tilgjengelig på denne kanalen.

Besvarelsen skal inneholde:

- Konseptvurdering for spillelagfunksjonalitet i Facebook
- Fungerende løsning i Facebook, integrert med Norsk Tippings systemer

Norsk Tipping jobber etter smidige metoder og ønsker derfor at gruppen følger samme metodikk og jobber i sprinter med leveranser i slutten av hver sprint.

E. Prosjektavtalen

HØGSKOLEN I GJØVIK

PROSJEKTAVTALE

mellom Høgskolen i Gjøvik (HiG) (utdanningsinstitusjon),

Jørn Berg Nordlund (Norsk Tipping)

(oppdragsgiver), og

Jan Fredrik Gundersen

Sondre T Johannessen

(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 08.01.15 til 15.05.15.

   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): Jan Fredrik Gundersen dato 27.01.15

Sondre T Johannessen dato 27.01.15

_____ dato _____

_____ dato _____

Oppdragsgiver (signatur): J. Nordlund dato 27.01.15

IMT Dekan/prodekan (signatur): _____ dato _____

NORSK TIPPING

HØGSKOLEN I GJØVIK

Prosjektplan

Sosiale spill på Facebook – Bacheloroppgave våren 2015

Jan Fredrik Gundersen og Sondre T. Johannessen

Innholdsfortegnelse

1. MÅL OG RAMMER

1.1. Bakgrunn

Norsk Tipping jobber kontinuerlig for å forbedre kundeopplevelsen både gjennom et godt spilltilbud og ved å møte kundene på deres premisser. Norsk Tipping har ingen ønsker om å konkurrere med sosiale plattformer, men vil tilby løsninger for kundene der de er. Dette innebærer å muliggjøre tettere tilknytning til blant annet sosiale medier.

Norsk Tipping er i konseptfasen med å utvikle en basisløsning på sin plattform for spillelagsfunksjonalitet, som i første fase ikke vil ha tilknytning til sosiale medier. Spillelag er den digitale varianten av «kameratenes» tippelag og «venninnegjengens» lottoklubb.

Oppgaven går ut på å utvikle en løsning som ved å benytte Facebook gjør spillelags-funksjonaliteten til Norsk Tipping tilgjengelig på denne kanalen.

Oppgaven fant vi på IMT-seksjonen ved HiG sitt fronterrom. Vi tok raskt kontakt med Jørn Berg Nordlund i Norsk Tipping og sendte over relevante papirer. Vi hadde deretter et Lync-møte og avtalte å ta dette prosjektet like før jul.

1.2. Prosjektmål

Effektmål:

- Større synlighet for Norsk Tipping på sosiale medier.
- Økt kundedatabase.
- Nye samhandlingsmåter med spillere.

Resultatmål:

- Merfunksjonalitet for Norsk Tipping sitt spillelagskonsept.
- En fungerende prototypeløsning for spillelag på Facebook.

Læringsmål:

- Få innsikt i hvordan et større prosjekt utarter seg og bidra til å utvikle ny programvare.
- Større innsyn i hvordan APIer blir designet og fungerer.
- Større innsyn i hvordan databaser blir designet og fungerer.

1.3. Rammer

Siden vår oppdragsgiver er Norsk Tipping så må vi forholde oss til retningslinjer satt av dem. De ønsker å ha en fungerende prototype klar når vi er ferdig med prosjektet. Det vil si at vi må

forholde oss til tidsrammen vi er blitt utdelt til bacheloroppgaven, fra start 28.januar til levering 15.mai for å ha den klar.

Spillelagsfunksjonaliteten er fremdeles i konseptfasen så det er ikke satt noen begrensinger eller foretrukket teknologi, for hvordan vi skal utvikle applikasjonen, av Norsk Tipping. Vi står fritt til å velge hvordan vi vil gå frem for å løse oppgaven fra et teknologisk perspektiv.

Vi behandler ikke sensitiv data om brukere (kortnummer, personnummer, etc.), men informasjon fra Facebook-profiler. Norsk Tipping har ikke gitt oss noen rammer på hvordan det skal bli behandlet, men vi har likevel et ansvar for å ikke misbruke tilliten til brukere.

Det må tas hensyn til Facebook sin ”Platform Policy” (https://developers.facebook.com/policy) ved utvikling av applikasjonen for at den skal bli godkjent til bruk på Facebook.

Fra Norsk Tipping sin side er det ønsket at spillelagsfunksjonaliteten skal fungere i dagens moderne nettlesere på PC og mobile enheter. Vi må derfor ta hensyn til at det vi utvikler vil virke like bra på en mobil som på en PC.

Det er ingen økonomiske rammer å ta hensyn til.

1.4. Arkitektur

Selve applikasjonen blir en grafisk HTML5 basert webside som gjør at den enkelt kan bli tatt i bruk på alle plattformer. Facebook Canvas gir mulighet for å implementere en webside som en applikasjon i Facebook. Dette gir samtidig muligheten for å bruke Facebooks egen Graph API. Det er denne APIen som gir eksterne operatører tilgang til informasjon om en Facebook-brukers profil, ved tillatelse. Dette skjer ved hjelp av GET-requester til APIen som responderer med et JSON-array basert på hva brukeren spør etter og hvilke rettigheter som er gitt. Rettighetene er gitt ved en “User Access Token” som blir definert når en bruker logger inn gjennom facebook. Denne gir programmet midlertidig sikker tilgang til Facebooks APIer.

Selve applikasjonen blir selvstendig. Det skal altså med litt moderering være mulig å bruke applikasjonen hvis Facebook skulle slutte å eksistere. Facebook vil ikke få tilgang til det som skjer av aktivitet i applikasjonen, men applikasjonen vil få muligheter til å bruke POST og GET requester mot Facebook. En spillers informasjon vil bli opprettet og lagret på en database uavhengig av Facebook, men med data som er hentet ut derfra. Facebook sin rolle vil være ved innlogging og eventuelt posting av statuser på vegne av applikasjonen.

2. OMFANG

2.1. Fagområde

Konseptet til spillelagsfunksjonalitet strekker seg over en rekke områder. Det kreves en front-end og en back-end løsning. Det trengs APIer for å hente og sende relevant data mellom kjernefunksjonalitet, spillelagsfunksjonalitet og den sosiale plattformen. Dette innebærer kompetanse om databaser og optimalisering basert på hvilke krav som blir stilt av oss, utvikling av APIer, samt koding av løsninger for en front- og back-end løsning i et foretrukket programmeringsspråk. Kommunikasjonen foregår hovedsaklig mellom en database, web-service, Norsk Tipping sine APIer og applikasjonen, med unntak kommunikasjon med Facebook når det måtte være ønskelig.

2.2. Avgrensning

Ved avgrensingen har vi tatt hensyn til vår kompetanse innenfor fagområdene, tiden som er gitt til prosjektet (frist 15.mai 2015) og Norsk Tipping sine retningslinjer om hva de skal utvikle selv. Oppgaven blir avgrenset til å designe og utvikle det som trengs for en fungerende prototype for spillelag gjennom sosiale medier. Dette innebærer rapporter om hva som kreves av utvikling og teknologi fra interne og eksterne hold.

2.3. Oppgavebeskrivelse

Det skal designes og implementeres en fungerende del (prototype) av konseptet spillelag på Facebook som vil ta for seg fotballdelen av konseptet. Programmet skal være tilgjengelig for brukere av Facebook og skal gjennom sin profil ha muligheten til å delta i et spillelag. En spiller som vil bruke applikasjonen må tillate at applikasjonen kan hente informasjon om spilleren fra Facebook som navn og venner.

Etter spilleren har godkjent og er logget inn via Facebook vil følgende funksjonaliteter være tilgjengelige:

- Mulighet for å invitere venner til programmet.
- Liste over spillelag som en spiller er medlem i.
- Mulighet for å opprette et spillelag og invitere spillere som er brukere av programmet.
- Fylle ut kupong og levere den inn individuelt og/eller til et spillelag i applikasjonen.
- Graf over kuponger levert inn sortert etter tidsrom og gevinst, med mulighet for informasjon om hver kupong. En graf for hvert spillelag; en graf for individuelle, en graf for Spillelag 1, en graf for Spillelag 2, …., en graf for Spillelag 5.

- Gå inn på et spillelag.
- Push-meldinger til Facebook. For eksempel en status: “Spiller1 har foreslått en kupong for midtuke” via Norsk Tipping Spillelag.

I et spillelag vil spillerne ha muligheter for å se aktuelle kuponger. Disse kupongene er autogenerert basert på kuponger som medlemmene har levert inn til spillelaget. Her har spillerne mulighet for å diskutere i et kommentarfelt, dedikert til hver kupong, om det skal gjøres endringer. En eller flere administratorer har rettigheter til å levere kupongen til Norsk Tipping. Et rankingsystem basert på poeng viser hvem som leder internt i laget. Det gis poeng etter antall rette i kupongen en spiller foreslo som er like med kupongen som laget leverte. Rankingsystemet kan sorteres etter kupong, uke, måned eller år.

3. PROSJEKTORGANISERING

3.1. Ansvarsforhold og roller

Gruppen består av to personer, Jan Fredrik Gundersen og Sondre T Johannessen. Begge studerer Bachelor i Ingeniørfag – Data ved Høgskolen i Gjøvik.

Oppdragsgiver
Vår oppdragsgiver er virksomhetsansvarlig for arkitektur ved Norsk Tipping, Jørn Berg Nordlund. Jørn sørger for at Norsk Tippings interesser blir ivaretatt, og sammen med Håvard Kindem (konseptutvikler hos Norsk Tipping) vil de bistå med kompetanse rundt kodedelen av prosjektet

Veileder
Vår veileder er studieprogramansvarlig for Ingeniørfag – Data, Ivar Farup (Professor of Computer Science, Phd, MSc).

Prosjektleder
Siden vi bare er to personer vil denne rollen vil være delt likt mellom Jan Fredrik og Sondre. Vi har jobbet sammen i prosjekt tidligere og har vist at vi kan ta argumentere og komme frem til reflekterte valg.

Webansvarlig
Jan Fredrik Gundersen er ansvarlig for nettsiden som også fungerer som blogg. I tillegg til å sette opp vår side vil han også være ansvarlig for å oppdatere den underveis med status og annen informasjon som sammendrag av møtereferater.

3.2. Rutiner og regler i gruppa

Jan Fredrik og Sondre har signert en avtale på at det skal avsettes minst 25-30 timer i uken til arbeid med prosjekt. Dette er er kalkulert i henhold til at 60 ECTS tilsvarer 1500-1800 timer arbeid pr. semester, som igjen tilsier at 30 studiepoeng krever 37,5 til 45 timer arbeid i uken. Bacheloroppgaven teller 20 studiepoeng.

Før og etter hver sprint så vil gruppen ha Microsoft Lync møter, eller et fysiske møter i Norsk Tipping sine lokaler på Hamar med Jørn eller Håvard om hvordan forrige sprint har gått og hvilke funksjonaliteter fra produktkøen som skal prioriteres ved neste levering.

Vi har utarbeidet følgende grupperegler:

- Det er viktig at begge yter like mye, slik at ikke en av partene sitter igjen med mye mer arbeid enn den andre.
- Arbeidet utføres til avtalt tid.
- Logg-dokumentet må oppdateres etter hver økt. Dermed får vi logget hva vi har gjort og eksakt tidsbruk.
- Møter til avtalt tid eller gir beskjed. Fravær skal meldes i god tid med begrunnelse.
- Vi er begge ansvarlig for å overholde tidsfrister fra HiG/oppdragsgiver.

Om det skulle være problemer som oppleves av en av medlemmene og det ikke ønskes konfrontasjon med det andre medlemmet så kan dette tas opp med vår veileder Ivar. Problemet skal helst bli adressert til vedkommende som skaper problemet først. Om det er gjentakende at problemer oppstår hos det ene medlemmet, som å ikke levere etter frister eller legge inn for lite arbeid, så kan medlemmet bli fjernet fra gruppen om vår veileder er enig at det utsetter en reel risiko for at prosjektet ikke blir fullført ordentlig.

4. PLANLEGGING, OPPFØLGING OG RAPPORTERING

4.1. Hovedinndeling av prosjektet

Siden vår gruppe består av to personer må vi maksimere produktiviteten. Norsk Tipping jobber etter smidige metoder og ønsker derfor at gruppen følger samme metodikk og jobber i sprinter med leveranser i slutten av hver sprint.

Vi kommer derfor til å bruke Scrum med sine karakteristiske egenskaper i dette prosjektet: Product Backlog, Sprint Backlog, sprintmøter hver dag, planleggingsmøter der ny Sprint Backlog blir utarbeidet for hver iterasjon, og vi ser på oss selv som et Scrum-team. Vi mangler en Scrum-master, men siden vi bare er to på gruppa antar vi at vi klarer oss uten dette.
På grunn av tidsrammen er det ytret et ønske fra oppdragsgiver om sprinter på to uker. Gruppen er enige i at sprinter av denne varigheten vil passe bra da vi ønsker å sikre god dialog og levere det oppdragsgiver ønsker.

4.2. Plan for statusmøter og beslutningspunkter

Vi vil ha ukentlige statusmøter med vår oppdragsgiver der vi setter oss ned før en ny sprint og diskuterer hva som er gjort og hvilke av elementene fra product backlog vi skal jobbe med til neste sprint.

Etter hver sprint vil vi ha et nytt møte med Jørn eller Håvard der vi går gjennom sprinten.

Vi vil også ha en samtale med veileder to ganger i måneden. Her vil vi ta opp fremgang, evt. manglende fremgang, nye oppgaver, forslag til endringer og annet relatert til prosjektet. Referater vil publiseres på vår hovedprosjektblogg.

5. KVALITETSSIKRING

5.1. Dokumentasjon, standardbruk og kildekode

Norsk Tipping ønsker å ha på plass en spillagsfunksjonalitet i løpet av de nærmeste årene. Det er dermed viktig at vi gir fra oss en dokumentasjon som er ryddig og god nok slik at andre

utviklere kan fortsette eller modifisere vår kode. Kildekoden må være lesbar og vil bli kommentert underveis.

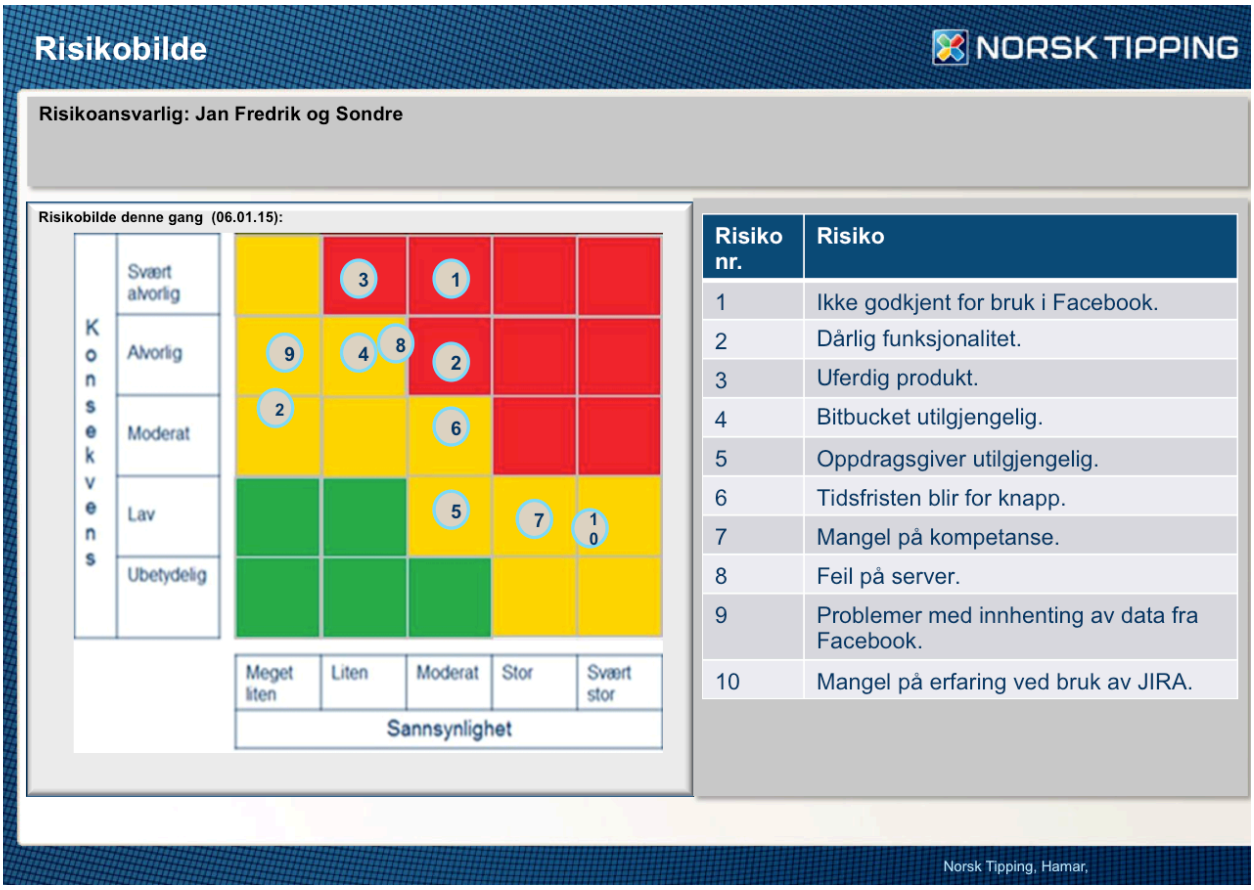
Når det er brukt kode som er skapt av andre, det være seg Stackoverflow.com eller Github.com, så skal det dokumenteres i koden. Det gjøres ved kommentarer i selve koden som sier noe om koden er modifisert eller direkte kopiert, samt en link eller beskrivelse hvor den er tatt fra. Vi kan kun bruke programkode som er open source og som kan benyttes uten at man mister rettighetene til å endre eller modifisere koden. Dette inkluderer Apache, MIT og BSD – men utelukker ikke andre.

5.2. Konfigurasjonsstyring

Gruppens medlemmer er godt kjent med og har benyttet seg av konfigurasjonsstyrings-verktøyet Git tidligere. Vi vil bruke BitBucket som kodebrønn fordi det gir oss muligheten til å ha koden privat.

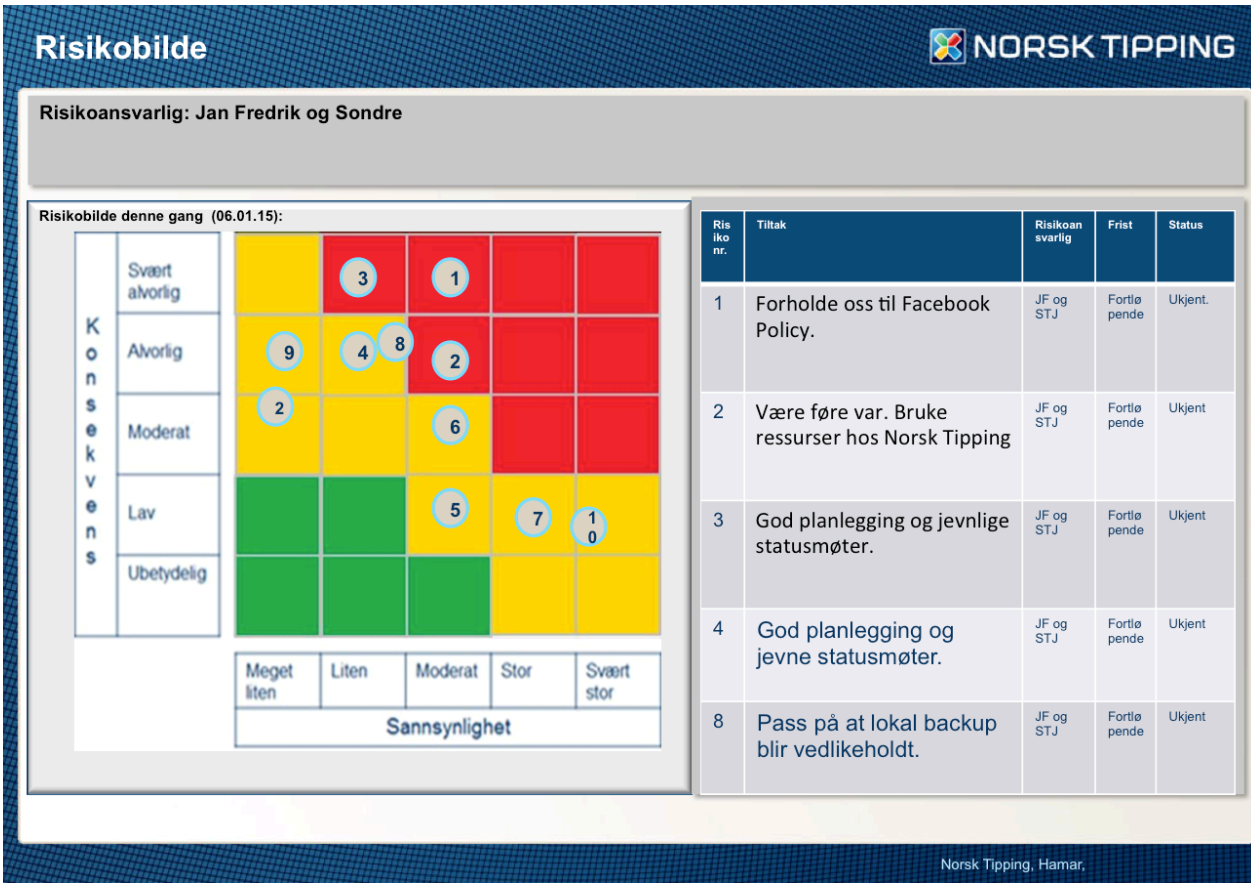
HiG stiller også med lisens til prosjektsverktøyet JIRA, som er utviklet av selskapet som har laget BitBucket. JIRA kan være nyttig å bruke slik at vår oppdragsgiver kan følge med på hvor vi er og hva vi har gjort og eventuelt legge inn ting i gjøremålslisten.

5.3. Risikoanalyse



Risiko nr.	Risiko
1	Ikke godkjent for bruk i Facebook.
2	Dårlig funksjonalitet.
3	Uferdig produkt.
4	Bitbucket utilgjengelig.
5	Oppdragsgiver utilgjengelig.
6	Tidsfristen blir for knapp.
7	Mangel på kompetanse.
8	Feil på server.
9	Problemer med innhenting av data fra Facebook.
10	Mangel på erfaring ved bruk av JIRA.

Illustrasjon 1 Risikobilde



Illustrasjon 2 Risikotiltak

Score:	**Risiko/tiltak:**
Rød	Høy risiko. Strakstiltak kreves.
Gul	Moderat risiko. Risikoreduserende tiltak skal vurderes.
Grønn	Liten risiko. Ingen risikoreduserende tiltak kreves, så fremt lov og forskrift er oppfylt.

Plan for håndtering av de viktigste risikoene:

1. Facebook har en policy for hva som kan utvikles og hvordan det skal utvikles. Norsk Tipping inngår under kategorien ”gambling”, derfor må vi ta hensyn til regler og de rettningslinjer som er satt opp rundt dette. Selvfølgelig må vi også ta hensyn til norsk lov og Norsk Tipping sine vurderinger.

2. Vi har skaffet oss en ekstra kontakt hos Norsk Tipping, slik at vi kan bruke han dersom Jørn er utilgjenglig.

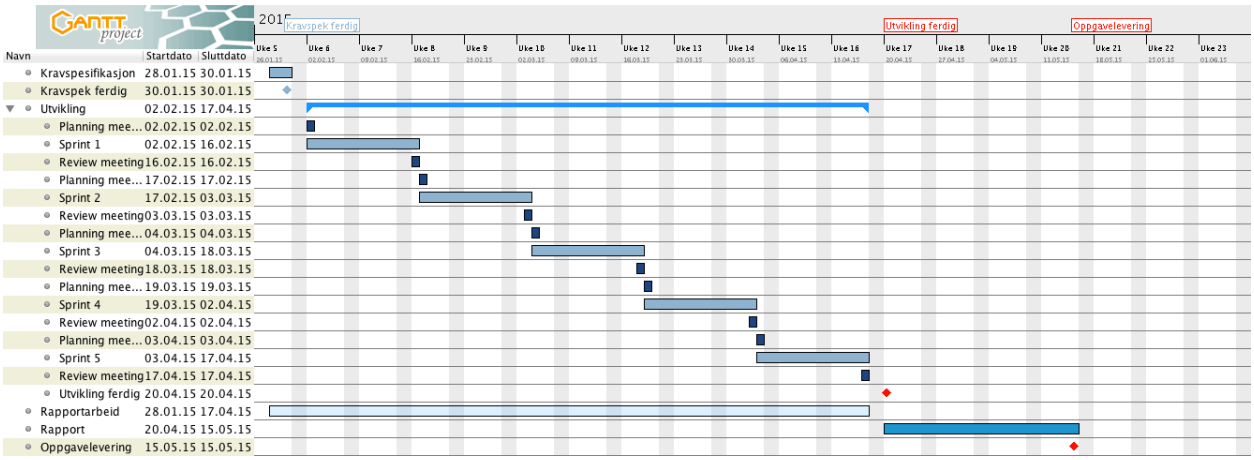
3. Ved å planlegge, blant annet ved hjelp av denne prosjektplanen, får vi et bedre overblikk over hva som må gjøres og hva vi står ovenfor. Dermed vil det være mindre sannsynlighet for at vi undervurderer prosjektets omfang eller overvurderer vår kunnskap.

Kritiske suksessfaktorer:

- Et brukervennelig grensesnitt
- Rask respons
- Stabilitet (programvaren må ikke kræsje)

6. PLAN FOR GJENNOMFØRING

Vi vil benytte oss av smidige metoder. Gantt-skjemaet vil inneholde mange sprinter og milepælene vil være markert med «diamanter». Spesifikasjonene av de forskjelige sprintene vil avgjøres i sprint planning meeting.



Klikk her for stor versjon.

Signatur gruppemedlemmer:

Jan Fredrik Gundersen

Sondre T. Johannessen

7. TERMINOLOGILISTE

- **API:** Står for “Application Programming Interface”. Betegner et grensesnitt i en programvare hvor deler av denne kan kjøres fra en annen programvare.
- **Git:** Distribuert versjonskontrollsystem.
- **Agile:** Systemutviklingsmetode. Legger vekt på god kommunikasjon med oppdragsgiver og hyppe leveranser.
- **Scrum:** Scrum er en utviklingsmetodikk. Man jobber i iterasjoner med små, men konkrete funksjonsbiter.
- **Facebook Canvas:** Vanlig HTML-webside i facebook.
- **Facebook Graph API:** For å få ut informasjon om brukere og for autentisering av innlogging. Bruk av Javascript eller PHP som back-end.
- **JSON:** Står for “JavaScript Object Notation”. JSON brukes for lagring og henting av dataobjekter.
- **Product Backlog:** Norsk: “Produktkø”. Alt som skal lages av funksjoner sortert i en liste.
- **Sprint Backlog:** De elementene man skal jobbe med den respektive sprinten.

Vedlegg G: Møtelogg

Dato	Aktør	Sammendrag
Desember 2014	Norsk Tipping	Samtaler på Lync med Jørn ang. oppgaven. Han ønsket at vi tok denne oppgaven og det gjorde vi. Sendte erklæring inn til Hilde v/ HiG
13. jan 15	Norsk Tipping	Lyncmøte med Jørn. Agile, omtrentlig ramme og sier at det skal sjekkes ut. Kapasiteten blir sjekket ut. Få på plass bedre oppgavespesifisering. Få en tidligere oppgave og sjekke "problemstillingen". Facebookbiten: Oppegående løsning, enkelt og presentere og se. Grensesnitt? Mye likt som wearables gruppa. Bruksopplevelsen. Dette tror vi kundene vil like. Teste i markedet? Agile: Starter med en min. løsning og tester. Legger på eller justerer. Tre trenere som tipper hver for seg. Man vedder på hvem som er best. (Eksempel) Mange åpne spørsmål. Starte spillelag, etablere spill og legge til medlemmer. Hvilke data må sendes til leveringskomponenten til NT? Må fungere på mobil/tab og i nettleser. Mobile first. Forventet detaljnivå? Få tak i og ta med til Jørn, for å dele opp i deloppgaver. Git? Google docs eller sharepoint?
16. jan 15	Gruppen	Vi møttes for å diskutere oppgaven og våre tanker. Undersøkte litt hvordan det kan gjøres på Facebook sin plattform. Vi besluttet å gjøre det kritiske først, som å opprette spillag, invitere til spillag og bli med i spillelag via Facebook. Vi satt også opp mal til prosjektplanen.
19. jan 15	Veileder	Hadde et møte med Ivar for å avklare ting. Vi fikk vite at vi kunne velge mellom å skrive på norsk og engelsk, at rapporten var det vi ble vurdert på og at vi måtte få på plass mest mulig før vi begynner med

		oppgaven. Ting som kodespråk, hvilke enheter som skal støttes er veldig viktig å få på plass.
20. jan 15	Norsk Tipping	Vi går for JIRA. Bloggen er oppdatert. Vi må få på et kapittel som omfatter arkitektur. Her kommer detaljerte rammer (fb canvas, fb apier). Lage rammene arkitekturmessig. Ti deltakere eks. sportspill. Hvis disse ti skal lage hver sine rekker hver uke. Flest 10-11-12. Best odds. Hvis en organisasjon vil ha et spillelag. Hva vil de ha i en sånn kontekst? Hva slags data trengs fra norsk tipping? Hva trengs i kjernesystemet? Det vil ikke ligge i spillmotoren hvem som har foreslått. Eks 15 spillere foreslår, men kun 10 er betalende eller kunder i NT. Funksjonelle rammer. Arkutekturrammer Kan ikke si alt i prosjektplanen. Hva gir fb canvas, sikkerhet , hva slags data kan ligge på Fb,hva syns brukere er greit. Data lagres hos NT eller skal eksterne kunne bruke. -Se på ulike varianter og komme med tilbakemeldinger. Innsikt som er viktig. Hvorfor den og den teknologi/arkitektur. Begrensinger.
27.jan 15	Veileder	Ivar så over prosjektplanen vår og kom med tilbakemeldinger på endringer som burde gjøres. Snakket litt om krav ved vurdering av oppgaven. Skal avtale nytt møte i løpet av morgendagen.
27.jan 15	Norsk Tipping	Fikk introduksjon i Norsk Tipping Hamar sine lokaler. Skrev under på taushetsærklering. De bruker ren Scrum. Avtalte to ukers sprinter istedenfor en.
28.jan 15	Veileder	Kort møte med Ivar for å gå gjennom prosjektplanen en siste gang. Spørte om work breakdown structure og kom frem (sammen med Ivar) til at det ikke var relevant å ha detaljer under hver sprint i Gantt-skjemaet. Dette siden Scrum ikke krever planlegging for hele perioden, men heller planlegger

		dette i forkant av hver sprint ved hjelp av “planning meeting”. Fikk tips om å skrive litt mer om open source lisenser.
2.feb 15	Gruppen	Planning meeting før første sprint. Brukte en online versjon av planning poker for å estimere tidsbruk på oppgavene og kom frem til fire oppgaver som skal fullføres de neste to ukene. Jørn var ikke med på møtet, men fikk en mail med informasjon.
16. feb 15	Gruppen	Review og planning meeting før sprint 2.
18. feb	Workshop, IBM/Norsk Tipping	Workshop med IBM ang. Bluemix
24. feb	Kurs, Norsk Tipping	Kommisjonærkurs i Hamar
2. mars	Veileder	Review og planning meeting før spring 3. Har brukt JIRA feil, med tanke på estimater. Til neste gang skal vi estimere før vi legger til i sprinten.
12. mars	Norsk Tipping	Sikkerhetsinnføring på Hamar og oppdatering rundt oppgaven
18.mars	Norsk Tipping	Tips: Ikke bruk mye tid på hvordan ting skal se ut. Moderne softwareutvikling: gå utifra en hypotese, implementer basis funksjonalitet, presenter for kunde og få feedback. (Ref: NT, Finn, Apple, FB). Vi skal prøve å få implementert Watson for å gi oss en edge på oppgava. Jørn ønsker at vi bruker bluemix, vil nok være bra for oppgava og, ettersom det ikke er mange studenter som har tilgang til alt bluemix har å tilby. Feedback: SMS med spillelagskode via bluemix. Bruk google charts: https://developers.google.com/chart/
9. april	Norsk Tipping	Gått gjennom det vi har gjort så langt. Fått positive tilbakemeldinger. Ønske av begge at vi får inn noe

		som gir oppgaven “X-faktor” som å bruke Watson til å foreslå kuponger og spillelag for hver enkelt bruker. Dette vil IBM bidra med å se på. Vi startet med å kode resultatinnhentingen, slik at vi får satt kupongene som “vunnet/tapt”. Slik at vi også får gitt brukere og spillelag poeng basert på resultatene.
15. april	Gruppa	Utviklingen av ntsosial er nå ferdig. All funksjonaliteten og kravene fra prosjektplanen er implementert. I disse dager driver vi å tester mens vi begynner å sette opp mal for rapporten. I dag hadde vi møte med Ivar på engelsk (dette er et krav) og vi fikk råd om rapportskrivingen.

Vedlegg H: Utviklingslogg

Jan Fredrik

Dato	**Timer**	**Beskrivelse**
Desember 2014	3 t	Samtaler med oppdragsgiver Jørn rundt prosjektet.
13. januar	1 t	Møte med Jørn på Lync. Fikk mer klarhet rundt oppgaven
16. januar	3 t	Møte i Oslo. Diskuterte oppgaven og startet med prosjektplanen.
19. januar	1 t	Møte med Ivar Farup. Se møtelogg.
20. januar	5 t	Kort møte med Jørn og arbeidet videre på skolen med prosjektplanen
22. januar	3 t	Jobbet med prosjektplanen
27. januar	6 t	Besøk hos Norsk Tipping. Fått adgangskort, mer info og omvisning
28. januar	5 t	Jobbet på skolen, fullført prosjektplanen
29. januar	5 t	Konfiguerert server
1. feb	2 t	Rapportskriving
3. feb	6 t	Arbeidet med server og innlogging og lagring av data til database.
Uke 8	5 d	Møte hos Norsk Tipping med IBM. Intro til Bluemix. Implementert henting av spillelag fra DB, og opprettelse + at den som opprettet blir registrert som medlem.
24. feb	9 t	Kommisjonærkurs hos NT. For å få bedre innblikk i hvordan de leverer deres produkter og hvilket forhold kundene har per dags dato.
25. feb	8 t	Implementerte “Bli med i spillelag” med invitasjonskode. Valg: En invitasjonskode kan brukes flere ganger slik at dersom man vil invitere en gruppe mennesker kan disse bruke samme kode. Invitasjonskoden forsvinner etter 48 timer.

26-28. feb	18 t	Utfordring: Det går tregt når man hele tiden må spørre Facebook om ID. Derfor ønsker vi å lagre IDen slik at den alle spørringer slipper å gå gjennom Facebook. - Lagre i session - Cookies Valgt å lagre ID i Session. FBAuth.php kjører på hver side for å sjekke om brukeren er innlogget på Facebook. Dette er sikkerhet Facebook krever, og det lar oss være trygge på at man ikke får gjort noe med andre sine brukere, og at man blir logget ut av Sosiale spill om man logger ut fra Facebook. Implementert “Inviter til spillelag”, som ligger under hvert visSpillelag.php.
3-4. mars	12 t	Jobbet med å restrukterere koden slik at vi ikke trenger å hente FB.API hver gang vi trenger brukerens ID. Bruker PHP og FB session. Var på Hamar 3 mars.
5. mars	5 t	Endret på mineSpillelag.php slik at inviteringsfunksjonen også ligger der i stedet for en egen side. Møte med Ivar. Fjernet deler av SQL spørringene fra visSpillelag.php
10. mars	6 t	Laget en egen side, invited.php - der blir invite-koden sjekket og man får instrukser om hvordan man blir med. Endet index.php sånn at den henter antall spillelag og poengsum La til Facebook send-knapp på visSpillelag.php slik at man enkelt kan sende en melding til venner/grupper på Facebook
11. mars	9 t	Foredrag med Frode om rapporten Muligheter for å slette spillelag implementert men ikke aktivert. Må diskutere litt rundt denne funksjonen. Henter medlemmer av et spillelag fra DB. Legger et fint bilde ettersom hva ID er.
12. mars	8 t	Jobbet sammen på Hamar. Hadde sikkerhetsinnføring og modellerte hvordan vi ønsket at nye kuponger og nye spill skulle

		lagres.
17. mars	6 t	Sett på hvordan vi kan representere kuponger på spillelagets side. Testet med forskjellige alternativer, men alle ble skrotet.
18. mars	7 t	Laget en kupongrepresentasjon og den er nå aktiv. Brukte også mye tid på å sørge for at man blir sendt videre til den siden man ønsket å nå dersom man er logget ut. Tidlligere ble man sendt til index.php etter login. Dette var viktig å få på plass på grunn av bugen ved at om man laster en ny side før siden man er på er ferdig lastet, blir man sendt til login på nytt av. Også nyttig når session har utløpt og man klikker på en lenke, siden man da kommer til den siden man vil etter autentisering.
19. mars	7 t	- La til tabs på visSpillelag, en for foreslåtte, en for aktive og en for historikk. - Endret fra jquerUI-dialog til modal på meld ut/inviter - Lagt til saldo i nav.
20. mars	3 t	Ryddet i koden
23. mars	4 t	Begynte å skrive pseudokode og kode SQL setninger for henting av foreslåtte kuponger innad i et spillelag
24. mars	8 t	Henter nå kuponger tilhørende spillelaget, både aktive og foreslåtte. Bruker kan bli med på kuponger som betalende eller tilskuer Man får vist hvilke kuponger man er med på Kupong blir satt til “submitted” om antall personer som har betalt er oppfylt
7. mars	6 t	Laget “Slett bruker” funksjon
8. mars	4 t	Planlegge resultatinnhenting
9. mars	8 t	Møte med Jørn. Lage resultatinnhenting
10. mars - 13. mars	12 t	Brukte lang tid med Sondre på å designe kupongtabellene og utvide databasen til å omfatte poengregistrering og lag.

Tiden før påske	7-8 d	Fått på plass endelig kupongtuseende. Restrukturering av kode.
30. mars - 5. april	-	Påskeferie
Uken etter påske	6*5 t	Jobbet med Sondre på Hamar. Fant XML-filen som inneholder resultater fra Langoddsen hos Norsk Tipping. Vi henter dataene vi trenger fra denne og kjører en CRON-jobb hver time. Denne fila går gjennom alle kamper i systemet vårt som ikke er avgjort, den setter disse til vunnet eller tapt. Om alle kamper i en kupong har gått inn, har også kupongen gått inn. Denne settes da til vunnet (evt. tapt) og bruker får poengene sine. Mye tid til feilsøking etter sprint.

Sondre

Dato	**Timer**	**Beskrivelse**
Desember 2014	3 t	Samtaler med oppdragsgiver Jørn rundt prosjektet.
13. januar	1 t	Møte med Jørn på Lync. Fikk mer klarhet rundt oppgaven
16. januar	3 t	Møte i Oslo. Diskuterte oppgaven og startet med prosjektplanen.
19. januar	5 t	Møte med Ivar Farup. Se møtelogg. Researchet bruk av Facebook applikasjoner og jobbet med prosjektplan.
20. januar	5 t	Kort møte med Jørn og arbeidet videre på skolen med prosjektplanen
27.januar	4 t	Møte med Ivar og så møte med Jørn. Arbeid med prosjektplan.
28. januar	5 t	Fullført prosjektplan på skolen.
29.januar	5 t	Utprøving og research av facebook sine SDKer, med suksessfull innlogging.

2.februar	5 t	Fikse oppsett av SSL på localhost mtp på canvas. Kastet bort mye tid på det og endte opp med å ikke bruke det. Møtelogging.
3.februar	7 t	Innlogging med facebook og inkludering av bootstrap design. Utprøving av graf.
4.februar	3 t	Research og webkoding.
5.februar	7 t	Designing av poengsystem, første uktast.
6.februar	5 t	Research og webkoding.
Uke 8	20 t	Møte med NT og IBM. Prototype på poengsystem og forklaring av dette i dokument. (Utgår)
24.feb	9 t	Kommisjonærkurs hos NT.
		Kodet representasjon av poeng for bruker og dens spillelag.
25.feb	7 t	Enkel representasjon av oddskamper vha et statisk json objekt. Skal utvides til å bli live fra norsk tipping sine apier.
27.feb - 1-mars	12 t	Jobbet med å representere aktive spillsystemer innad i lagene.
5.mars	5 t	Problem med headers det skal hentes ut liveinformasjon fra NT sin api; getGameInformation. Brukt mye tid på det. Utsettes til jeg har fått tatt en prat med håvard.
6-8.mars.	8 t	Research og generell koding. Endret en bug i scriptet som gjorde at nav bar ikke funket skikkelig.
9.mars	8 t	Møte med Jørn. Rydding i kode.
10.mars	10 t	Ymse layout. Google hangout med håvard. Hjalp meg å sette opp en api i bluemix vha node.red. Enkelt og kjapp måte å unngå NT sine sikkerhetsheaders på. Setter headeren til domenet i apien vi lagde på bluemix og vips så får vi tak i ferdigformatert info i et json objekt.
11.mars	9 t	Foredrag med Frode. Generell koding.

12.mars	8 t	Sikkerhetsinnføring hos NT. Designet hvordan nye kuponger og spill skulle lagres. Tatt med hensyn til NT sine apier for høyest mulig synkroniseringsevne.
15.mars - 24.mars	24 t	Implementering av NT sine layoutmoduler for å opprette en liste over kamper med HUB alternativer. Et stort DOMtre som er avhengig av hverandre. Det er altså ikke mulig å bare kalle på en klasse. Klassen må også være “children”-element til ett viss type “parent”-element.
15.mars - 24.mars	24 t	Kodet kupongform for å sende inn en kupong til databasen med sine spillobjekter.
15.mars - 24.mars	24 t	La til “alertify.js” biblioteket for å implementere notifications når en kamp blir valgt. layout.
25.mars	4 t	Endret kupongformen til å gi alternativet for å være med som tilskuer. Rydding i kode.
28.mars	8 t	Representering av statistikk vha google charts. Mye knoting, ikke ferdig.
30.mars - 5.april	-	Påskeferie.
7.april	8 t	Ferdigstilling av representering av statistikk vha av google charts. Line-, bar- og pie-chart er brukt.
8.april	6 t	Jobbet med getresultat med Jan Fredrik. Endret på utseendet til nykupong for å få det mer responsivt og mobilvennlig, siden det ikke var støttet av CSS til NT.
9.april	4 t	Jobbet med Jan Fredrik på NT. Liten oppdatering med Jørn. Vi har i tiden som har vært innsett at vi ikke klarer å implementere mulighetene til Watson på egenhånd. Jørn fikk fiksa slik at vi får hjelp av IBM til dette. Fhpt skjer det før oppgaven skal inn!
10.april	5 t	Parkodet pointshandler, og der den skulle implementeres, med Jan Fredrik. Innser vel begge at vi kanskje skulle kodet all php kode på denne måten fra begynnelsen, ideelt sett.

14.april	5 t	Endring av accesstoken fra FB for å gi appen tilgang til å poste på vegge OM bruker ønsker dette. Bruker får nå valget om å poste kupongen til feeden sin etter kupongen blir foreslått.
Etter 15.april	4 d	Litt endring av kode her og der. Koding av prediksjonsfunksjonalitet med Håvard og Jan. Testing av Watson med Norsk Tipping sine data.
Øvrig		Mange timer har gått til researching og testing av kode som ønskes å implementers. Mesteparten er skrotet og ikke nevnt spesifikt siden dette er ting som skjer hele tiden og ikke nødvendigvis er relevant. Det som er relevant er nevnt.

Vedlegg I: Statusoppdateringer fra bloggen

Sprint	Hva er gjort?
1	● Satt opp server med database ● Fått på plass innlogging via Facebook ● Fått lagret brukerdataene i vår database ● Designet poengsystem
2	● Opprette spillelag ● Liste over spillelag som en spiller er medlem i ● Invitere venner til spillag ● Bli medlem av invitert spillelag ● Utmelding av spillelag ● Begynt på Mine poeng og Ny kupong
3	● Utvide database til å omfatte poengregistrering og lag. ● Lagre brukerID sikkert i nettleser ● Utmelding av spillelag ● «Min side»-funksjon ● Slette spillag
4	● Tegne og skrive kode for kuponger ● Fylle ut kupong, levere privat eller til spillelag ● Restrukturere koden ● Hente kuponginfo fra DB og vise statistikk ● visSpillelag henter kuponger tilhørende spillelaget
5	● La til statistikk m/ grafer ● Resultatene for kampene hentes hver time vha. XML fra Norsk Tipping. Vår fil går gjennom og finner de kampene brukerene våre har og setter disse til vunnet eller tapt. ● Deler ut poeng om brukers kupong gikk inn. ● Poeng blir trukket fra om man vil spille som tilskuer

	● Toppliste i hvert spillelag ● Skrev Readme for prosjektet (for å kunne flytte til annen server)