HØGSKOLEN I OSLO OG AKERSHUS

Bacheloroppgave i Ingeniørfag - Data og Informasjonsteknologi

Forprosjektrapport

Fredag, 20. Januar 2017

Bachelorgruppe 10
André Hovda *(s169964)* INFORMATIK
Magnus Bärnholt *(s199221)* HINGDATA
Alexander Skjolden *(s114143)* HINGDATA
Hans Petter Osvold *(s929913)* HINGDATA

Innhold

1 Presentasjon

Oppdragsgiver:	Accenture
Prosjekttittel:	Eventure (Working title)
Oppgave:	Event App
Periode:	5.1.2017 - 25.5.2017
Gruppenummer:	10
Gruppemedlemmer:	André Hovda Magnus Bärnholt Alexander Skjolden Hans Petter Osvold
Intern veileder:	Aiko Yamashita aiko.yamashita@hioa.no
Kontaktpersoner:	Fredrik Bjørnøy fredrik.bjornoy@accenture.com Christian Møller Andersen c.moller.andersen@accenture.com Marius Torsrud marius.torsrud@accenture.com
Eksterne veiledere:	Ørjan Johansen orjan.johansen@accenture.com Ingleiv Johansen ingleiv.johansen@accenture.com

2 Dagens situasjon

2.1 Oppdragsgiver

Vår oppdragsgiver **Accenture** er et av verdens ledende konsulentselskaper. Accenture leverer tjenester innenfor rådgiving, teknologi og outsourcing. De stiller med en stor fot innenfor Norge med ca 1'100 ansatte, og kontorer på Oslo (Fornebu), Bergen og Stavanger. Globalt har de nesten 400'000 ansatte og kontorer i over 200 byer i 52 land. I Norge arbeider de for mange av Norges og verdens største virksomheter, og tilbyr kompetanse og løsninger innenfor de fleste områder og bransjer.

Vi ser på Accenture som en bedrift som tilbyr god kompetanse og ressurser for faglig og personlig utvikling. Under arbeidet med bachelorprosjektet vil vi blant annet få tilgang til nye og moderne kontorlokaler på Fornebu, kursing, deltagelse på lønningspils og Accenture-årsfest, tett oppfølging av dyktige veiledere som vet hva som skal til i en god bacheloroppgave, samt en "smak" på arbeidslivet som konsulent.

2.2 Oppgaven

Vi har fått i oppdrag av oppdragsgiver å utvikle en event app. Denne appen skal spesifikt kunne støtte events, som Santa Claus Pub-crawl [1] eventet som årlig arrangeres i desembermåned i Oslos gater. Dette er et bar-til-bar arrangement, hvor deltakerne møtes samlet på et utested for en eller flere drinker, for deretter utover kvelden, ta turen samlet til fots, videre til andre forehåndsbestemte barer. Deltakerne blir også oppfordret til å kle seg i nisse- og julekostymer. Ansatte på Accenture har i alle år stått for stor deltagelse på dette arrangementet, og ser nå en nytte i en slik app.

3 Mål og rammebetingelser

3.1 Funksjonalitet

Appen skal kunne brukes til å finne events og delta på disse. Ved deltagelse på et event skal appen tilby funksjonalitet utover hva eksisterende event apper generelt tilbyr. Dette er først og fremst funksjonalitet knyttet til Santa Claus Pub-crawl arrangementet, men funksjonaliteten skal være av en karakter hvor den kan være av nytte i tilsvarende arrangementer, og arrangementer generelt, eksempelvis konferanser, festivaler, klasseturer, vennesamlinger etc.

Funksjonaliteten som kreves av Santa Claus Pub-crawl arrangementet, og som er gitt som krav av oppdragsgiver, er følgende:

- Geo-lokasjon eller funksjon for å finne ut hvor bjellenissen og arrangør befinner seg
- Kart over hvilke puber som er på årets rute med informasjon om sted etc.
- Mulighet for notifikasjon når gruppen skal flytte på seg. Enten i form av at varsel sendes fra bjellenissen og/eller varsel i form av avstand og tid til neste sted
- Bildedeling
- Nominasjon og kåring av beste kostyme
- Støtte for flere typer events

Disse kravene anser vi som en del av kjernefunksjonaliteten til appen. Vi har i dialog med oppdragsgiver bygget videre på disse kravene, og kommet frem til en noe mer fullverdig kravspesifisering av det vi anser som kjernefunksjonaliteten til appen. Her har vi tatt hensyn til hva som forventes av oppdragsgiver, brukere av appen, samt hva gruppen anser som oppnåelig innenfor tidsrammene.

Denne kravspesifiseringen har vi utarbeidet til brukerhistorier fordelt under følgende epics": events, brukere, poster, konkurranser og bilder. Vedlegg A.3 viser en skjermdump av alle brukerhistoriene under disse "epicene".

3.2 Retningslinjer

- Gruppen skal benytte smidig arbeidsmetodikk (nærmere bestemt Scrum) og arbeide i Sprints.
- Kode og kommentarer skal skrives på engelsk
- Dokumentasjon skal skrives på norsk, og med latex
- Gruppen skal ha et fokus på clean code

- Gruppen skal være konsistent i bruk av etablerte og relevante kode konvensjoner
- Løsningen skal testes med enhetstester og andre relevante tester
- Løsningen skal ta hensyn til sikkerhet og privacy hvor det er aktuelt
- Appen skal ta hensyn til universell utforming
- Appen skal følge relevante design guidelines

3.3 Teknologier og verktøy

Gruppen står relativt fritt fram i valg av teknologier fra oppdragsgivers side, så lenge disse er drøftet med veiledere. Med det som utgangspunkt har vi kommet fram til følgende teknologier som relevante for å løse oppgaven:

- Slack [2] som primær kommunikasjonskanal innad i gruppen
- Google Drive [3] til dokumentdeling
- Overleaf [4] til rapportskriving
- Youtrack [5] for oppgavestyring
- Travis [6] for bygging, testing, og deployment
- SonarQube [7] for å sikre kodekvalitet.
- Git[1] og GitHub [8] for versjonskontroll og integrasjon med CI
- Front-End utvikles i Android etter en MVP arkitektur, med Java 8 og biblioteker som: RxJava [9], RxAndroid [10], Retrofit [11], GSON [12], Dagger 2 [13]
- Front-End utvikling for Android skal følge Material design retningslinjene [14].
- Android Studio som IDE[2] for Android utvikling
- Instrumented tester med Espresso [15] for Android
- Spring boot [16] som webserver rammeverk (API[3] Server)
- JHipster [17] med Spring Boot, Hibernate, Elasticsearch m.m
- IntelliJ som IDE for Spring utvikling
- JUnit [18], Mockito [19] for server-side testing

4 Løsninger og alternativer

4.1 Prosjektstyring og organisering

Vi har valgt å følge en smidig arbeidsmetodikk. Dette har vært svært ønskelig fra oppdragsgivers side, og samsvarer også med våres ønsker. Den smidige arbeidsmetodikken vi har valgt er Scrum-metodikken, hvor

[1] Git - versjonskontrollsystem for sporing av datafiler og koordinering av endringer på disse filene blant flere brukere
[2] Integrated Development Enviroment
[3] API (Application programming interface) er et sett av subrutine definisjoner, protokoller og verktøy for bygging av programvare

arbeid blir delt inn i sprinter. Her har vi satt lengden på hver Sprint til å vare i 2 uker. Vedlegg A.1 viser en fremdriftsplan med inndeling av sprintene.

4.1.1 YouTrack vs Jira

Ønsket innad i gruppen var å benytte Jira som "issue tracker", for å administrere og organisere den smidige og SCRUM baserte arbeidsmetodikken vår. Jira er også et verktøy som Accenture benytter internt. Kostnaden forbundet med et slikt abonnement var noe vi helst ville unngå. Vi fant derfor et alternativt program; *YouTrack*. YouTrack vil dekke de behovene vi har og er ikke noe dårlig alternativ for oss.

Et mye simplere alternativ er Trello, som tilbyr enkel tavleorganisering, men vi anser dette som en for svak løsning for et prosjekt av det omfanget dette prosjektet har.

4.1.2 Versjonskontroll

Ingen softwareprosjekter har livets rett uten en eller annen form for versjonskontroll. *Git* har, siden det spant ut av kjerne teamet til Linux, vært så og si en bransjestandard, og det vil være denne teknologien vi vil benytte. Som upstream vil mest sannsynlig *GitHub* benyttes.

4.1.3 Continuously integration

Som en del av utviklingsstakken og vektøyene knyttet til dette, vil devops og følgelig CI være en naturlig del. Dette er verktøy vi vil benytte til å kjøre tester, kodekvalitet, og ikke minst deployment. En typisk flyt i dette vil være; kode "commits"og pushestil et sentralt repositorie. Derfra går det et signal til en CI server, som vil starte prosessen den er satt opp til å kjøre. En slik prosess vil innebære at CI serveren trekker ned siste versjon av programvaren, bygger, tester og validerer koden. Inntreffer alle kriteriene som er satt til denne prosessen, vil også CI serveren kunne deploye programvaren til en produksjonsserver (eller lignende). Som CI programvare vil vi mest sannsynlig abonnere på tjenesten Travis CI", da denne tjenesten krever lite kunnskaper, og er lett tilgjengelig.

4.1.4 SonarQube

For å sikre kodekvalitet vil vi benytte oss av *SonarQube*. Et slikt verktøy vil være vesentlig for å sikre at vi møter kravene vi setter til "clean code"og bruk av kode konvensjoner. SonarQube vil også være et godt verktøy for å gi oppdragsgiver og veileder et innsyn i koden.

4.1.5 Fordeling av ansvar Back-End og Front-End

Vi har valgt å dele inn ansvaret for Back-End og Front-End i to grupper, med Alexander og Hans Petter på Back-End, og André og Magnus på Front-End. Alexander besitter tidligere erfaring i både Spring rammeverket og Android utvikling, og vil også bistå med sin ekspertise på Front-End delen. André besitter også tidligere erfaring innenfor Android-utvikling.

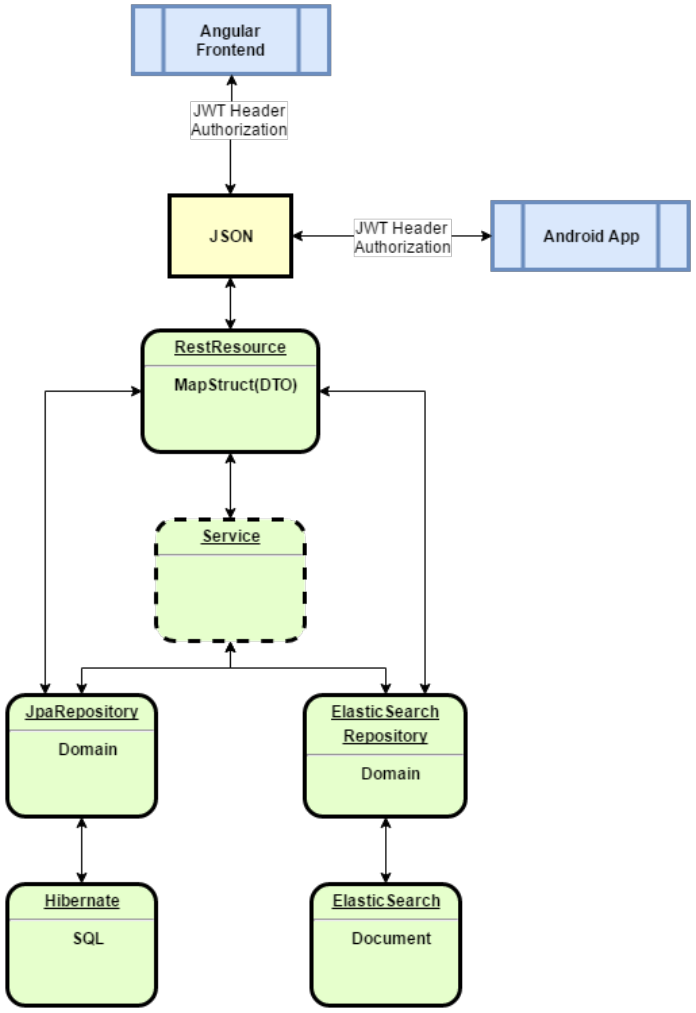
4.2 Back-End

4.2.1 JHipster

Valget for backend software falt ganske naturlig på Spring Boot. Spring Boot er et webrammeverk med Java som programmeringsspråk. Det å ha Java i frontend og backend gjør det lettere for oss å kunne jobbe på kryss av teamene. JHipster muliggjør det for oss å komme rask igang med en profesjonell Spring Boot stakk, uten at vi bruker mye tid på oppsett m.m. En løsning som JHipster gjør også backenden automatisk mer vedlikeholdbart. Figur 4.1 viser hvordan back-end stakken vil kunne bli seende ut.

4.2.2 Teknologier

Som en del av JHipsters generator natur vil vi stå relativt fritt til å velge teknologier i backend. Det faller oss likevel naturlig å benytte oss av tradisjonelle relasjonsdatabase for permanent lagring. For å løfte løsningen vil vi se på muligheten til å implementere og benyttes oss av Elasticsearch, dette for å kunne tilby bedre støtte for blandt annet lokasjonssøk.



Figur 4.1: Back-end stakk med JHipster

4.3 Front-End

4.3.1 Android vs iOS vs Hybrid

En av de større problemstillingene vi raskt sto ovenfor, gjaldt valg av platform til appen. Vi sto ovenfor valget mellom Android, iOS eller en hybrid løsning som Xamarin eller PhoneGap. Hvert valg har sine fordeler og ulemper, og vi gikk her til grundig research for å sikre oss at vi gikk videre med det valget som best dekker det oppgaven krever. Vi valgte til slutt opp med utvikling for Android platformen. Dette har sine fordeler ved at Android har den største markedsandelen internasjonalt, med 86,8% mot iOS sin 12.5% mot slutten av 2016 [20].

Samme kilden viser også en tydelig utvikling i trend i favør Android det siste året. I Norge er det derimot iOS som har et vesentlig overtak på markedsandelen [21]. Iphone brukere har også betydelig større forbruk blant "In app purchasesenn Android brukere. Dette vil ikke være relevant for oppgaven vår under vårt scope, men hvis det er et ønske i fremtiden å utvide appen med støtte for et billettsystem, vil dette være relevant. På bakgrunn av dette anser vi iOS som et veldig aktuelt alternativ. Dette alternativet viste seg dessverre å ikke være et alternativ, grunnet maskin og ressursmangler, ettersom at Xcode (som kreves for iOS utvikling) kun kan kjøres på Mac maskiner.

Både iOS og Android utvikling går under det som regnes som "native"-utvikling, og alternativet for å nå flere brukere vil være å gå for en hybrid løsning, som da vil støtte flere platformer, inkludert iOS og Android. Blant hybrid er det Phonegap og Xamarin som er de to mest aktuelle alternativene. Phonegap baserer seg på HTML5, css og javascript. Ulempene her er en mindre responsiv app, og mye av "feel'en" en native app tilbyr vil forsvinne, blant annet grunnet et generisk design og UX framfor det man får ved å følge designretningslinjene til den spesifikke platformen. Xamarin derimot skrives i C# og kompileres til native, som gir økt ytelse framfor Phonegap, men vil fortsatt ikke kunne oppfylle designretningslinjene til både iOS og Android. Dette anså vi som et strekt alternativ, frem til "research"vi gjorde viser at en betydelig del av kodebasen (antatt 30%) fortsatt vil måtte bli skrevet i native [22], og da står vi over utfordringen over å fordype oss i både Swift, Cocoa og Android.

Figur 4.2 viser en oversikt over nøkkelpunkter av fordeler og ulemper for hvert alternativ.

iOS		**Android**		**Hybrid**	
Positiv	***Negativ***	***Positiv***	***Negativ***	***Positiv***	***Negativ***
Når flest brukere i Norge (og Santa Claus Pub-crawl) Veldig nyttig erfaring Mye dokumentasjon / Stack overflow Native (Bedre kontroll og effektivitet)	Xcode og maskinvarekrav Ingen erfaring på forhånd	Erfaring på forhåd Mye dokumentasjon / Stack overflow Alle kan kode Storst markedsandel på verdensbasis Native (Bedre kontroll og effektivitet)	Færre brukere i Norge enn de andre alternativene.	Vil nå alle brukere/platformer	Ingen erfaring på forhånd Fysisk testing Vanskelig å finne "best practises" osv. Xamarin (og PhoneGap til en liten grad) vil fortsatt kreve noe koding i native

Figur 4.2: iOS vs Android vs Hybrid

Vi foretok oss også den klassiske *"Stackoverflow testen"*, hvor vi sammenlignet antall spørsmål lagt ut under hver platform på nettsiden `stackoverflow.com` (et stort online community for og av programmere for deling av programmeringsrelatert kunnskap). En slik test kan gi en indikasjon på tilgjengelige ressurser, som vil være kunne ha en stor betydning, hvis vi skulle låse oss fast i tekniske problemer, eller generelt informasjon rundt best practices, måter å unngå uante bugs, osv. Resultatene av denne testen vises i figur 4.3. Det er spesielt her hybrid alternativene viser sine største svakheter. med kun 18842 (for Xamarin) og 49618 (for PhoneGap) spørsmål, mot nesten en halv million spørsmål for iOS, og nesten en million spørsmål for Android.

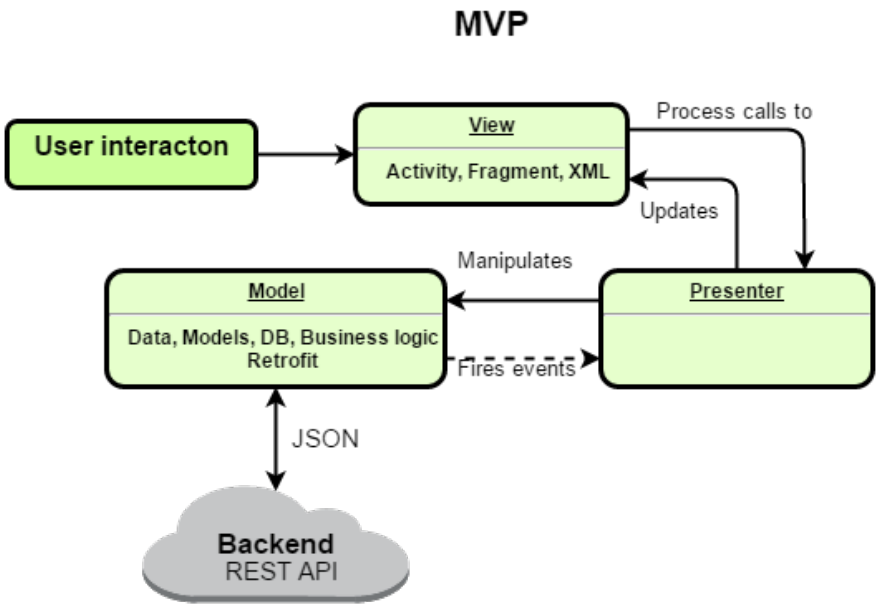
Plattform	**Stackoverflow spørsmål**
iOS	483689
Android	936624
Xamarin	18842
PhoneGap/Cordova (HTML5)	49618

Tall fra 2017-01-09

Figur 4.3: Stackoverflow test

4.3.2 MVP vs MVVM

Vi har valgt å benytte *Model View Presenter* (MVP), som arkitekturmønster for Android applikasjonen, fremfor det mer tradisjonelle *Model View Controller* (MVC) mønsteret. MVC har nemlig sine svakheter i forhold til "Seperation of Concerns", da spesielt når kompleksiteten øker. Det er her MVP viser sine styrker; en dypere "seperation of concernsgjør løsningen bedre rustet for skalering, vedlikehold og testing. Figur 4.4 gir en grov beskrivelse av hvordan MVP mønsteret vil bli implementert front-end.



Figur 4.4: MVP mønster

Et interessant alternativ er *Model View ViewModel* (MVVM); et noe nyere mønster i Android sammenheng, som skiller seg fra MVP med data-binding mellom ViewModel og View. MVVM vil også gi de samme fordelene som nevnt ovenfor for MVP, og gir en dyp "seperation og concerns".

4.3.3 RxJava og RxAndroid

Reaktiv programmering fokuserer på data flyt og endringer. RxJava og RxAndroid (RxAndroid - for å tilpasse seg til JVMen til Android) er biblioteker som tilbyr dette på en elegant måte og støtter diverse andre tredjepartsbiblioteker som Retrofit. Disse to bibliotekene til sammen vil sørge for at kommunikasjonen mot REST back-end vil virke elegant og sømløst. En slik modell åpner og for endel muligheter når det kommer til fleksibel uthenting av data fra forskjellige kilder (som f.eks et cache, database m.m).

4.3.4 Dagger 2 og Dependency Injection

Siden Android (i motsetning til Spring Boot) ikke tilbyr noen form for Dependency Injection som en del av kjernebibliotekene, vil det være naturlig for oss å finne denne funksjonaliteten andre steder - dagger er et slikt bibliotek. Med Dagger 2 har Dependency Injection på Android platformen blitt både enkelt og kraftig. Daggers versjon 2 vil sørge for å binde sammen klassene under kompilasjon, i motsetning til tidligere versjoner som gjorde dette under kjøring, og følgelig unngå et potensielt ytelsesproblem.

4.4 Retrofit og GSON

Selv om Android tilbyr større for kommunikasjon med eksterne tjenester, og serialisering i kjernebibliotekene velger vi likevel å finne denne funksjonaliteten via tredjepartsbiblioteker. Retrofit skaper et abstraksjonslag mot vår back-end tjeneste som gjør at detaljene rundt oppkoblingen forsvinner. Sammen med GSON, som er et JSON (de)serialisering bibliotek, skaper det en kodebase som er veldig kompakt, men samtidig beskrivende og funksjonell.

4.5 Material Design

Både vi og oppdragsgiver ønsker en app som gir en god brukeropplevelse. For å oppnå dette er det vesentlig at vi tar hensyn til velkjente deisgn og UX[4] prinsipper for mobil. Det vil også være en stor styrke å tilpasse design og UX etter de platformspesifkke retningslinjene, da dette vil sørge for at appen oppfører seg slik brukerne er vant til fra bruk av andre apper, samt tar hensyn til f.eks. navigasjonsbaren til Android og hvordan dette påvirker UX prinsipper og brukeroppførsel.

Ved å benytte Material design retningslinjene sørger vi for at appen automatisk oppfyller mange av disse design og UX kravene. Material er også muligens den best dokumenterte design retningslinjen til Android.n

[4] UX - User experience

Referanser

[1] T. Gundersen, "The 16th santa claus pubcrawl in oslo new invite," [Besøkt 15-Jan-2017]. [Online]. Available: https://www.facebook.com/events/1673999059556651/ (Sitert på side 4.)

[2] Slack: Technologies, Inc., "Slack: Where work happens," [Besøkt 14-Jan-2017]. [Online]. Available: https://slack.com/ (Sitert på side 5.)

[3] Google, "Google disk," [Besøkt 15-Jan-2017]. [Online]. Available: https://www.google.com/drive/ (Sitert på side 5.)

[4] Overleaf, "Overleaf: Real-time collaborative writing and publishing tools with integrated pdf preview," [Besøkt 15-Jan-2017]. [Online]. Available: https://www.overleaf.com/ (Sitert på side 5.)

[5] JetBrains, "Youtrack," [Besøkt 15-Jan-2017]. [Online]. Available: https://www.jetbrains.com/youtrack/ (Sitert på side 5.)

[6] "Travis ci," [Besøkt 15-Jan-2017]. [Online]. Available: https://travis-ci.org/ (Sitert på side 5.)

[7] "Sonarqube," [Besøkt 19-Jan-2017]. [Online]. Available: https://www.sonarqube.org/ (Sitert på side 5.)

[8] GitHub, Inc., "Github," [Besøkt 15-Jan-2017]. [Online]. Available: https://github.com/ (Sitert på side 5.)

[9] "Rxjava," [Besøkt 15-Jan-2017]. [Online]. Available: https://github.com/ReactiveX/RxJava (Sitert på side 5.)

[10] "Rxandroid," [Besøkt 15-Jan-2017]. [Online]. Available: https://github.com/ReactiveX/RxAndroid (Sitert på side 5.)

[11] "Retrofit," [Besøkt 15-Jan-2017]. [Online]. Available: https://square.github.io/retrofit/ (Sitert på side 5.)

[12] "Gson," [Besøkt 15-Jan-2017]. [Online]. Available: https://github.com/google/gson (Sitert på side 5.)

[13] "Dagger," [Besøkt 15-Jan-2017]. [Online]. Available: https://google.github.io/dagger/ (Sitert på side 5.)

[14] "Material design guidelines," [Besøkt 19-Jan-2017]. [Online]. Available: https://material.io/guidelines/ (Sitert på side 5.)

[15] "Espresso," [Besøkt 19-Jan-2017]. [Online]. Available: https://google.github.io/android-testing-support-library/docs/espresso/ (Sitert på side 5.)

[16] Pivotal Software, Inc., "Spring boot," [Besøkt 15-Jan-2017]. [Online]. Available: https://projects.spring.io/spring-boot/ (Sitert på side 5.)

[17] "Jhipster," [Besøkt 19-Jan-2017]. [Online]. Available: https://jhipster.github.io (Sitert på side 5.)

[18] "Junit," [Besøkt 19-Jan-2017]. [Online]. Available: http://junit.org/junit4/ (Sitert på side 5.)

[19] "Mockito," [Besøkt 19-Jan-2017]. [Online]. Available: http://site.mockito.org/ (Sitert på side 5.)

[20] IDC, "Idc: Smartphone os marketshare, 2016 q3," 2016, [Besøkt 19-Jan-2017]. [Online]. Available: http://www.idc.com/promo/smartphone-market-share/os;jsessionid=97ABAAA0A8CF5C0E3F52965C18305E7D (Sitert på side 8.)

[21] digi.no, "Her er dataene - mobilstatistikk for mai 2015," [Besøkt 19-Jan-2017]. [Online]. Available: http://www.digi.no/artikler/her-er-dataene-mobilstatistikk-for-mai-2015/208836 (Sitert på side 8.)

[22] "Html5 vs. phonegap vs. xamarin vs. native," [Besøkt 19-Jan-2017]. [Online]. Available: http://blah.winsmarts.com/2014-2-HTML5_vs_PhoneGap_vs__Xamarin_vs_Native.aspx (Sitert på side 8.)

A Vedlegg

A.1 Fremdriftsplan

Sprint	Dato	Oppgaver
Sprint 0 - Planlegging og tilrettelegging	05.01 - 20.01	13.1 Presentere arbeid for interne veiledere/oppdragsgiver • Presenterbar prototype i trådskisse/wireframe format 20.1 Innlevering av forprosjektrapport • Overordnet forståelse for prosjektet • Valg av platformstøtte • Valg av teknologier • Utrededning av rammebetingelser • Styringsdokumenter • Prosjektagbok • Fremdriftsplan • Risikovurdering • Oppsett av Youtrack, brukerhistorier og inndeling i Sprints • Tildeling av roller og arbeidsoppgaver
Sprint 1	21.01 - 05.02	• Oppsett av utviklingsmiljø • Testplan • Utikling starter på frontend og backend • Opprette forbindelse gjennom hele stacken (backend til frontend) • Utvikling • Testing • Dokumentering
Sprint 2	06.02 - 19.02	• Utvikling/Testing/Dokumentering
Sprint 3	20.02 - 05.03	• Forberedelse til presentasjon neste Sprint • Arbeid med midtrapport • Utvikling/Testing/Dokumentering
Sprint 4	06.03 - 19.03	8.3 Midtrapport og presentasjon • Utvikling/Testing/Dokumentering
Sprint 5	20.03 - 02.04	• Utvikling/Testing/Dokumentering
Sprint 6	03.04 - 16.04	• Utvikling/Testing/Dokumentering
Sprint 7	17.04 - 30.04	• Utvikling/Testing/Dokumentering • Kodefrys ved avslutning av sprinten.
Ferdigstilling av sluttrapport	01.05 - 25.05	25.5 Innleverng av sluttrapport • Ferdigstilling av sluttrapport
Muntlig presentasjon	26.05 - 06/09.06	6/9.6 Muntlig presentasjon Kursing i presentasjonsteknikk hos Accenture

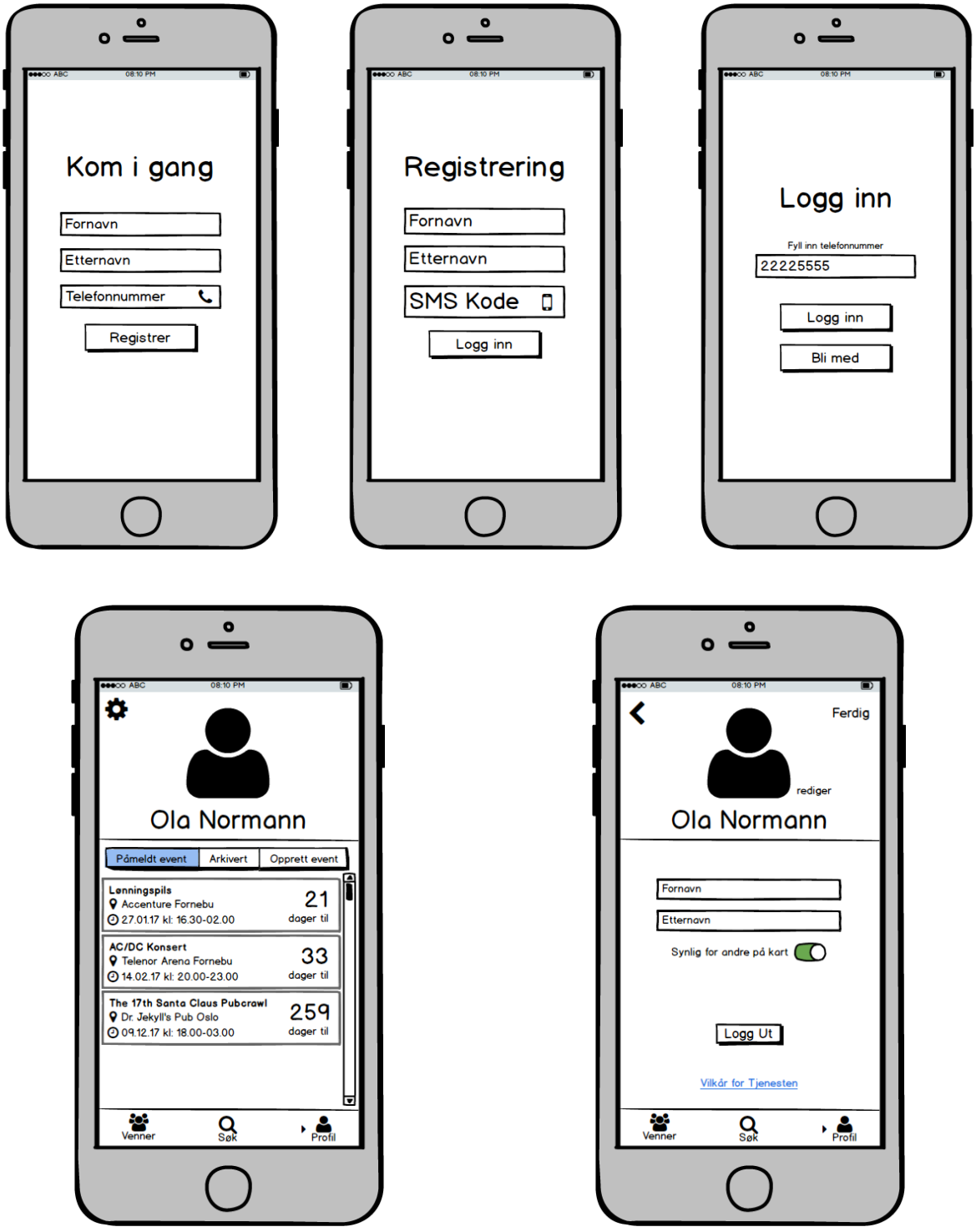
A.2 Risikoskjema

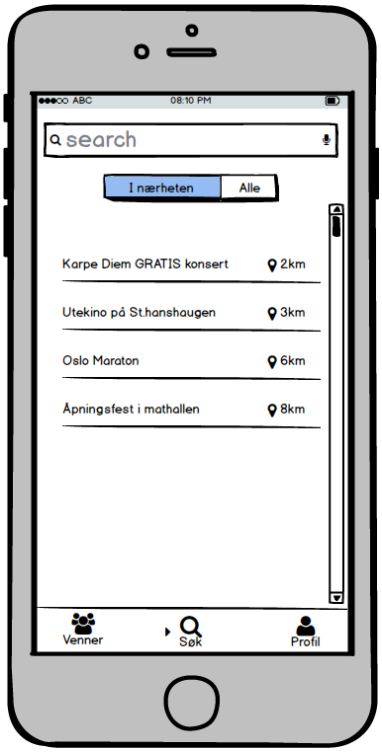
Risiko	Sannsynlighet	Konsekvens	Tiltak
Endringer underveis	Høy	Lav	Smidig arbeidsmetodikk bengreser drastisk konsekvensene av endringer undeveis i prosjektet. Som et følge av denne arbeidsmetodikken vil sannsynligheten for endringer være høy, men dette kommer uten de samme konsekvensene som i tradisjonell fossefallsmetodikk.
For lite tid / For bredt scope	Middels	Middels	Vi planlegger å være ferdige med utvikling i god tid i forveien slik at vi har et buffer av tid tilgjengelig til ferdigstilling av sluttrapporten. Dette bufferet tillater også utviklingen til å fortsette noe utover fristen vi setter hvis dette skulle være nødvendig. Smidig arbeidsmetodikk hvor utvikling, testing og dokumentering gjøres under hver Sprint vil gjøre det mye enklere å beregne det totale tidsomfanget underveis, og vi unngår en situasjon hvor vi for eksempel ender opp med for liten tid til å fullføre testing eller dokumentasjon etter utviklingsfasen. Vi har også valgt ut hva vi anser som kjernefunksjonalitet i appen, og som også etter våres estimeringer havner med god margin innenfor den tiden vi har tilgjengelig. Annen funksjonalitet vi ønsker å implentere klassifiserer vi som ekstra funksjonalitet, med en lav prioritet, og vil kunne revurderes og reprioriteres underveis hvis situasjonen tillater det, og våres opprinnelige tidsestimater gir oss ekstra spillerom.
Mindre alvorlig sykdom	Høy	Lav	Tilrettelegging for møter og kommunikassjon innad i gruppen over nett. Tilrettelegging for å kunne jobbe enklere hjemmefra
Alvorlig sykdom	Lav	Høy	Sørge for å tildele kompetanse på tvers av oppgaver i prosjektet. Vi planlegger å ha kodegjennomganger i slutten av hver uke hvor gruppene for front-end og back-end gjennomgår, og gir en oversikt over arbeidet som har blitt gjort. Sørge for god dokumentering gjennom hele prosjektløpet slik at samtlige i gruppen er oppdaterte på alle valg, drøftinger og resultaterer som er gjort. En smidig arbeidsmetodikk vil også her kunne spille til vår fordel, da forandringer og begrensing av scope vil være lettere å gjennomføre enn alternativt fossefallsmetodikken.
Tap av data	Lav	Middels	Dokumentasjonen skrives i overleaf og vil gi versjonshistorikk med mulighet for tilbakestillelse. Det vil også bli foretatt jevnlige lokale "backups". Kode vil med hyppighet bli lastet opp til Github, med versjonskontrollsystemet Git, slik at vi drastisk reduserer mengden arbeid som kan forsvinne hvis en lokal kopi skulle forsvinne, korrupteres eller lignende. Git tillater også tilbakestilling til tidligere versjoner, som vil være svært nyttig hvis arbeid ved en feil blir overskrivet eller fjernet ved en oppdatering i kodebasen.

A.3 Brukerhistorier

N	MAIN-26	Konkurranse	Jan 16
N	MAIN-21	Stemme på bilder i en konkurranse	Jan 16
N	MAIN-20	Starte konkurranse	Jan 16
N	MAIN-19	Nominere bilder til en konkurranse	Jan 16
N	MAIN-18	Opprette konkurranse til en event	Jan 16
N	MAIN-27	Bilder	Jan 16
N	MAIN-35	Slette bilder	Jan 16
N	MAIN-17	Legge til bilde til en event	Jan 16
N	MAIN-22	Tagge bilder	Jan 16
N	MAIN-2	Events	Jan 16
N	MAIN-38	Rollen event admin	Jan 16
N	MAIN-36	Rollen event creator	Jan 16
N	MAIN-14	Legge til lokasjoner på event	Jan 16
N	MAIN-37	Tildele rollen event admin	Jan 16
N	MAIN-28	Se lokasjon til eventadministrator	Jan 16
M	MAIN-23	Tagge eventer	Jan 16
N	MAIN-13	Melde på event	Jan 16
N	MAIN-12	Favoritt event	Jan 16
N	MAIN-11	Søke i eventer	Jan 16
N	MAIN-10	Se eventer	Jan 16
N	MAIN-9	Slette events	Jan 16
N	MAIN-8	Redigere events	Jan 16
N	MAIN-7	Opprette events	Jan 16
N	MAIN-25	Post	Jan 16
N	MAIN-24	Tagge poster	Jan 16
N	MAIN-34	Redigere poster	Jan 16
N	MAIN-15	Opprette poster på event	Jan 16
N	MAIN-16	Kommenter poster på en event	Jan 16
N	MAIN-3	Brukere	Jan 16
N	MAIN-1	Brukerpålogging	Jan 16
N	MAIN-6	Registrere lokasjonsdata	Jan 16
N	MAIN-33	Melde seg ut	Jan 16
N	MAIN-5	Redigere profil/konto	Jan 16
N	MAIN-4	Opprette konto	Jan 16

A.4 Trådskisser/Prototype

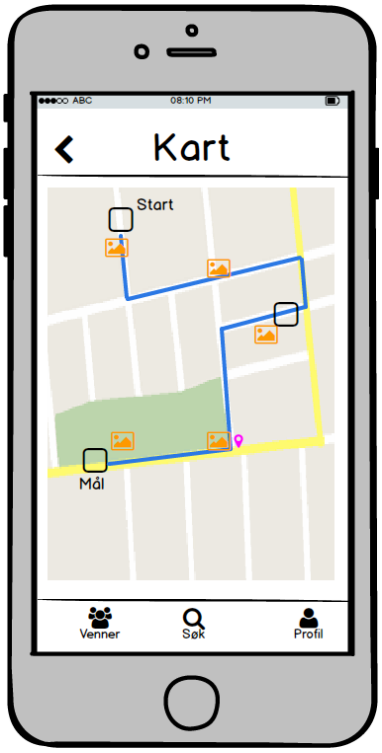


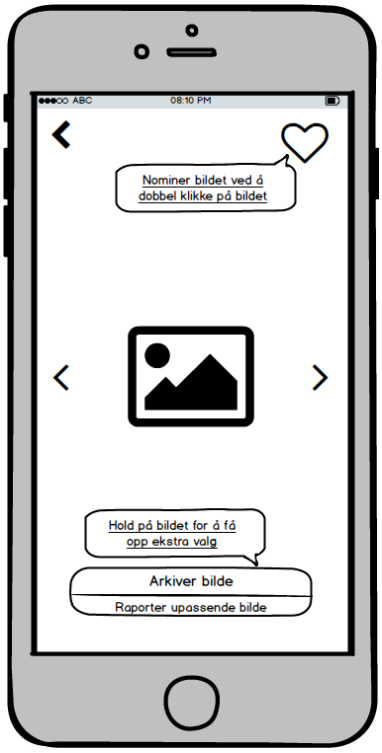

search
I nærheten
Alle
Karpe Diem GRATIS konsert 2km
Utekino på St.hanshaugen 3km
Oslo Maraton 6km
Åpningsfest i mathallen 8km
Venner
Søk
Profil


search
Venner
Søk
Profil


12. JAN
Arrangement 1
Arrangement Bilde
16.30 - 02.00
Dr. Jekyll's Pub Oslo
Skal
89
Interessert
103
Venner
7
Beskrivelse
Velkommen til PubCrawl 2017!
Bar til bar runde gjennom Oslo's beste pubber
Happy Hour fra 16.00-18.00
Konkurranse med beste kostyme
Inviter alle du kjenner å bli med på festligheten!
Venner
Søk
Profil
Innformasjon
Nyhet Feed
Konkurranse


Opprett / endre event
Arrangement navn
Kl: 17.00 - 01.01.2017
Beskrivelse
Legg til sted
1. Rådhusplassen 24b
2. Ibsensgate 14
3. Slottsparken
Venner
Søk
Profil


Kart
Start
Mål
Venner
Søk
Profil


Nominer bildet ved å dobbel klikke på bildet
Hold på bildet for å få opp ekstra valg
Arkiver bilde
Raporter upassende bilde


Nyhet Feed
Ola Normann har lagt ut 5 bilder.
18. desember
Kari Hansen har opprettet en meningsmåling.
17. desember
Stem på den det matønsket du foretrekker.
Koldtbord med 6 retter 16
Peppes Pizza 4
Lasagne 9
Emil Lillegutt
14. desember
Til alle som ikke har fått kjøpt inngangsbillett, det vil bli lagt ut 25 ekstra biletter fra og med 16. desember.
Venner
Søk
Profil


konkurranse
Ola Normann, Juletre 31
Venner
Søk
Profil