NTNU
Norwegian University of
Science and Technology

Gitek PLU

Author(s)

Daniel Rosland
Eivind Ristebråten

Bachelor of Science in Engineering - Computer Science
20 ECTS
Department of Computer Science and Media Technology
Norwegian University of Science and Technology,

18.05.2016

Supervisor(s) Frode Haug

Sammendrag av Bacheloroppgaven

Tittel:	**Gitek PLU**
Dato:	18.05.2016
Deltakere:	Daniel Rosland Eivind Ristebråten
Veiledere:	Frode Haug
Oppdragsgiver:	Gitek AS
Kontaktperson:	Khai Van Ngo, kvn@gitek.no, 61178401
Antall sider:	78
Antall vedlegg:	
Tilgjengelighet:	Åpen

Sammendrag:

Price-look-up(PLU) koder brukes i butikker for enkelt å håndtere løsvektvarer som grønnsaker og frukt. Selv om de i seg selv er effektive og gir en god kobling mellom produkt og system, er de ikke veldig effektive om kasseoperatøren ikke kan dem. Oppgaven vår går ut på å utvikle et innovativt verktøy for å gjøre det lettere for butikkmedarbeidere å lære seg disse kodene. Vi skal i løpet av denne perioden utvikle både en internettportal for å opprette kurs og konkurranser med PLU-koder, og en applikasjon for mobiltelefoner for å gjøre disse oppgavene. Sammenlagt skal dette bli en helhetlig løsning for å lære seg PLU-kodene til forskjellige varer.

Summary of Graduate Project

Title:	**Gitek PLU**
Date:	18.05.2016
Authors:	Daniel Rosland Eivind Ristebråten
Supervisor:	Frode Haug
Employer:	Gitek AS
Contact Person:	Khai Van Ngo, kvn@gitek.no, 61178401
Pages:	78
Attachments:	
Availability:	Open

Abstract: Price-look-up(PLU) codes are used in shops to handle bulk goods, like vegetables and fuits. In itself, they are efficient and provide a good link between product and system, but they are not very useful if the operator does not know them. Our task is to develop an innovative tool to make it easier for employees to learn these codes. During this period, we will develop both an Internet portal to create courses and competitions, and an application for mobile phones to attend them. Put together, this will become a complete solution to learn PLU codes.

Forord

Vi vil takke Gitek AS for gode råd og veiledning, samt at de stilte både rom, utstyr og teknisk kompetanse til vår disposisjon. Vi vil også takke Frode Haug ved NTNU for god veiledning gjennom hele prosjektet.

Contents

List of Figures

Listings

1 Introduksjon

Vi har fått en oppgave fra Gitek om å lage et opplæringsverktøy for å hjelpe ansatte ved Coop Norge å lære PLU-koder. Price-Look-Up(PLU) brukes i butikker for å håndtere typisk løsvekt varer, fordi det er korte koder som gir en god tilknytning mellom vare og system.

Gitek AS leverer skreddersydde applikasjoner for bedrifter som ønsker bedre innsikt i driften gjennom dataanalyser som igjen er med på å effektivisere bedriften. Coop Norge er en av Giteks største kunder og de ønsker stadig nye applikasjoner som kan bidra til økt kundetilfredshet og effektivitet ute i butikkene.

1.1 Omfang

1.1.1 Fagområde

Price-look-up(PLU) koder brukes i butikker for å enkelt håndtere løsvekt varer som grønnsaker og frukt. Selv om de i seg selv er effektive og gir en god kobling mellom produkt og system, er de ikke veldig effektive om kasseoperatøren ikke kan dem.

PLU-koder ble tatt i bruk i 1990 i supermarkeder, og er en internasjonal standard satt av IFPS [1]. Selv om det er i utgangspunktet IFPS som bestemmer koder, har butikkjeder lov til å sette sine egne koder. Disse kodene er som regel 3-5 sifrede.

I dag eksisterer det ingen mobil-applikasjoner for opplæring av PLU-koder, og et oppslag av disse kodene er gjort via plasttavler som er satt opp ved kasseområdene. Det eksisterer et oppslagsverk for Android, men denne dekker kun IFPS sine koder.

1.1.2 Avgrensning

Vi skal lage en applikasjon som skal kunne brukes til opplæring av PLU-koder for medarbeidere i butikker, og som i tilleg skal kunne brukes som et oppslagsverk. Denne applikasjonen skal være til Android, men hvis tiden tillater det, ønsker oppdragsgiver at den også skal fungere på iOS.

Vi skal også lage et webgrensesnitt som skal brukes av administratorer. Der skal de kunne legge til og slette brukere, samt lage, endre og slette kurs. Disse kursene skal kunne bli assosiert til regioner, butikker, type vare og spesifikke brukere hvis ønskelig.

1.1.3 Oppgavebeskrivelse

Med Gitek PLU er tanken å tilby opplæring av PLU-koder til både nye og eksisterende medarbeidere på en tilgjengelig og mer engasjerende måte. For å oppnå dette skal Gitek PLU tilbys som en applikasjon på mobil, enten for iOS og/eller Android. For å gjøre det mer engasjerende er tanken å gjøre opplæring mer likt et spill. Det vil si et enkelt poengsystem og for eksempel vanskelighetsnivåer. Planen er at applikasjonen skal bli en fast del av opplæringen til nye ansatte.

For å sette opp og håndtere ulike opplæringer trengs det også et enkelt web-grensesnitt som administrator eller kursholder bruker. Denne må kunne opprette, endre og slette

kurs samt håndtere brukere.

Funksjonalitet for mobil-applikasjonen:

- Ulike moduser for å lære PLU-koder. Trykk på korrekt PLU, eller tast komplett PLU.
- Enkelt belønningssystem i form av poenger
- Vanskelighetsnivåer
- Rankingliste/scoreboard
- Oppslagsfunksjon. Søke på PLU, varenavn eller varekategori
- Applikasjonen skal fungere uten nett-tilkobling

Funksjonalitet for webgrensesnitt:

- Legge til/lage nye kurs
- Assosiere kurs til brukere, butikk, slag eller region
- Endre og slette kurs
- Oversikt over progresjon til brukere
- Importering av PLU-koder
- Endre PLU-koder

Funksjonalitet som er listet opp er ikke absolutt og viser kun hovedtrekkene. Det må forventes at det tilkommer noen i utviklingsfasen når en ser behovet. Oppdragsgiver forventer innspill fra oss på dette.

Det vil også bli behov for å designe og utvikle database, samt utvikle nødvendig tjenerfunksjonalitet for å håndtere kommunikasjon med webgrensesnitt og applikasjonen.

Oppdragsgiver vil bistå med veiledning gjennom hele prosessen, og ønsker at terskelen for å kontakte de er lav.

1.2 Målgruppe

1.2.1 Rapporten

Målgruppen for rapporten består av elever og lærere ved NTNU Gjøvik, avdeling for IMT, samt oppdragsgiver. Dette betyr at vi tar det som en selvfølge at personene i målgruppen har grunnleggende kunnskap innen programmering, databaser og systemutvikling.

1.2.2 Oppgaven

Selve oppgaven består av to deler. Det skal utvikles et webgrensesnitt, og en applikasjon for mobiltelefoner. Vi har derfor valgt å dele denne biten opp i to delere.

Webgrensesnitt

Målgruppen for webgrensesnittet består av oppdragsgiver, samt utvalgte administratorer hos Coop Norge. Dette er personer som har kjennskap til Coop Norge sitt system, og varer. De har grunnleggende erfaring med bruk av webgrensesnitt i andre løsninger. Det kan være personer i alle aldre, og med varierende kompetanse.

Applikasjon for mobil

Målgruppen for applikasjonen er stort sett yngre personer i alderen 18-30 år. Det vil bli tatt høyde for eldre personer, men fokuset er på de yngre. Gruppen består av både jenter og gutter, og vi tar det som en selvfølge at alle har erfaring med bruk av mobiltelefoner, også når det kommer til å benytte seg av andre typer applikasjoner. Applikasjonen

skal være utformet på en måte som gjør at personer med motoriske problemer, samt svaksynte, skal kunne benytte den.

1.3 Formål

1.3.1 Resultatmål

Ved prosjektets slutt skal følgende leveres:

- Et webgrensesnitt som en administrator kan bruke for å administrere kurs og oppgaver
- En mobil-applikasjon for Android hvor brukere kan øve på PLU-koder, delta på kurs og søke på varer
- Rankinglist for å se hvor mange poeng brukere har oppnådd på de forskjellige oppgavene

Utvidet funksjonalitet:

- Lage applikasjonen for iOS.

1.3.2 Effektmål

Følgende effektmål er gitt av Gitek:

- 50% mindre oppslag på PLU-kodetavle ved kasseområdet i løpet av ett år
- Fjerne papirversjoner av opplæringsmateriellet når det gjelder PLU-opplæring i løpet av ett år
- At 60% av alle ansatte benytter seg av applikasjonen utenom arbeidstid og oppsatte kurs

1.3.3 Rammer til oppgaven

Etter samtale med oppdragsgiver, er følgende krav og ønsker blitt utarbeidet:

- Mobil-applikasjonen skal fungere på minimum Android API level 16, som tilsvarer Android versjon 4.1
- Android-applikasjonen vil bli utviklet i Android-Java og XML
- Webgrensesnittet skal støttes fullt av Internet Explorer 11. Ønskelig at det også at det støtter nyeste versjon av Chrome og Firefox
- Webgrensesnittet vil bli utviklet i HTML, CSS og Javascript/jQuery
- Nødvendig tjenerfunksjonalitet vil bli utviklet i PHP og MSSQL
- All utvikling skal være ferdig og overlevert oppdragsgiver senest 18. Mai 2016.

1.4 Egen bakgrunn og kompetanse

1.4.1 Nåværende kompetanse

Fra standardiserte emner har vi begge utviklet generell kompetanse innenfor database, programmering og systemutvikling, som har vært relevant i forhold til denne oppgaven. I tillegg til dette så har vi også hatt valgemner innenfor mobil-utvikling, programvareutvikling, og spill-design. Dette har gitt oss kunnskap og erfaring innenfor Java, utvikling for Android, og spill-utvikling. Av personlige interesser så har vi begge utviklet enkle spill-prototyper.

1.4.2 Hva må læres?

Utifra formålene er det mulig å danne seg et bilde av hva slags kompetanse som vil være nødvendig for denne oppgaven. I utgangspunktet vil vi jobbe med to platformer, hvor det er forskjellige programmeringsspråk. Hvis vi også får med utvidet funksjonalitet, ender vi opp med en tredje platform. Utifra nåværende kompetanse, kan vi se at vi er nødt til å øke kompetansen innenfor de forskjellige platformene. Vi har grunnleggende kompetanse, men for å løse oppgaven må vi gå dypere på noen områder. Vi har derfor valgt å se på hva slags kompetanse vi kommer til å måtte forbedre, eller tilegne oss.

For webgrensesnittet vil det bli bli nødvendig å tilegne oss kompetanse, spesielt for CSS, Javascript og jQuery. Det vil også kunne være nødvendig å skaffe oss mer kompetanse innenfor PHP og HTML.

Hvis det blir aktuelt med en iOS-applikasjon, vil det også være nødvendig å tilegne oss kunnskap om det.

1.5 Rammer

1.5.1 Gjennomføring

Etter ønske fra arbeidsgiver, og basert på vår egen kompetanse, vil vi på dette prosjektet jobbe etter en inkrementell metode. Vi vil også ha jevnlige møter med både oppdragsgiver og veileder, slik at alle til enhver tid vet statusen til prosjektet.

1.5.2 Ansvarsforhold og roller

Da vi er en gruppe på kun to personer, vil ansvarsforholdene og rollene være delt mellom gruppens medlemmer. Eivind har derimot rollen som gruppeleder, og har derfor ansvaret for de endelige, og største avgjørelsene som angår gruppa og prosjektet.

1.5.3 Rutiner og regler i gruppa

- Forventninger
    - Alle gruppemedlemmer plikter seg til å jobbe minst 25 timer i uka hver. Disse timene inkludere også samlinger/møter hvor gruppemedlemmene jobber sammen
    - Det gies fritak fra påkrevd arbeidsinnsats på helligdager og i felles ferier. Det betyr derimot ikke at det er ulovlig å jobbe
    - Alle gruppemedlemmer skal gjøre det som blir avtalt, og til avtalt tid. Ved brudd på dette må det settes av ekstra tid for å komme ajour med arbeidet. Dette skal være utført innen utgangen av førstkommende sprint etter forsinkelsen
    - Ved ønske om ferie, må dette diskuteres internt i gruppa. Det må da tas hensyn til hvor lang tid som er igjen av prosjektet, og hvor mye jobb det er på de berørte sprintene. Hvis det blir godkjent ferie, kan det kreves opptil 50% arbeidsinnsats
- Møter
    - Alle gruppemedlemmer SKAL møte på avtalte møter, og til korrekt tid. Hvis dette ikke er mulig, skal alle gruppemedlemmer få beskjed så tidlig som mulig
    - Alle møter skal loggføres med et kort referat, tidpunkt og varighet

- Kostnader
  - Mindre kostnader for å dekke utgifter internt i gruppa, skal dekkes av gruppens medlemmer. Større kostnader som er nødvendig for å gjennomføre prosjektet diskuteres med oppdragsgiver
- Overtredelser og uenigheter
  - Ved uenighet om en avgjørelse, og det innen rimelig tid ikke er funnet en løsning, skal veileder og/eller kunde kontaktes for innspill
  - Etter gjentatte overtredelser av gruppereglene, og to skriftlige advarsler, kan gruppemedlemmer bli fjernet fra gruppa

1.6 Øvrige roller

Oppdragsgiveren vår er Gitek AS, som vil bistå med råd og veiledning. De vil også vurdere produktet underveis. Veilederen vår er Frode Haug, som vil bistå med oppgaveskriving og planlegging.

1.7 Rapportorganisering

Hele rapporten vil være på norsk, med unntak av sammendraget i starten av rapporten. Dette vil finnes på både norsk og engelsk.

1.7.1 Oppsett av rapporten

Kapittel 1

Kapitel 1 er dette kapittelet. Det inneholder en presentasjon og oversikt over selve oppgaven.

Kapittel 2

Dette kapittelet vil inneholde kravene til løsningen vår. Her inngår oppgaver som løsningen skal kunne løse, samt funksjoner løsningen skal ha. Her vil vi også beskrive generelle krav til løsningen som en helhet.

Kapittel 3

Dette kapittelet omhandler designet av løsningen vår. Det inneholder både det logiske designet av use case, databasen, og det grafiske designet av webløsningen og mobilapplikasjonen.

Kapittel 4

Dette kapittelet vil inneholde informasjon om kodingen som er blitt utført. Det vil beskrive script og plugins som er blitt benyttet, samt en presentasjon av hvordan vi har benyttet de forskjellige utviklingsspråkene.

Kapittel 5

Kapitel 5 vil handle om testing. Vi vil forklare hvordan vi har utført testingen, og hvilke elementer vi har lagt spesielt vekt på.

Kapittel 6

Dette kapittelet vil inneholder konklusjonen vår. Vi vil diskutere og argumentere for valgene vi har gjort, og hva vi kunne gjort anderledes. Vi vil også se på hva som kan gjøres

ved en eventuell videre utvikling av løsningen vår.

1.7.2 Praktisk

Vi kommer til å benytte oss av ShareLaTeX for å skrive raporten. Det er blitt utarbeidet en mal ved NTNU, som vi vil benytte oss av. Denne inneholder skrifter, logo og formateringer, slik at vi er sikre på at vi benytter oss av korrekt format.

2 Krav

Ønsket fra oppdragsgiver var at vi skulle jobbe etter en inkrementell metode. Grunnen til dette var at de var usikre på nøyaktig hvordan denne oppgaven best kunne løses, og derfor ikke hadde en fullstendig kravspesifikasjon. Vi hadde også blitt enige med oppdragsgiver at vi skulle ha jevnlige møter, og vi ble derfor enige med de om at vi skulle jobbe etter Scrum-metodikk. På grunn av dette valget vil dette kapittelet være skrevet i retrospektiv, da backloggen ble fylt ut mens vi arbeidet.

Det ble besluttet at det ville for oss være mest hensiktsmessig å ha èn ukes-sprinter. Det passet med ukentlige møter, og vi ble derfor enige om å ha møter hver fredag, slik at vi kunne gi de en oppsummering på slutten av uken. Disse møtene ble også brukt til å planlegge neste sprint. Etter møtene ville vi ha tid til å oppdatere kravspesifikasjonen, og backlog om nødvendig.

2.1 Use cases

2.1.1 Høynivå beskrivelser - Mobil-applikasjon

Innlogging / registrering av bruker

Aktør	Bruker
Hensikt	Logge inn, eller skaffe brukere, brukerkontoer.
Beskrivelse	En bruker skal kunne logge seg inn med en egen brukerkonto. Dette skal i utgangspunktet skje automatisk etter at en bruker har registrert seg. For å registrere seg må brukeren skaffe seg et brukernavn og et engangspassord fra en administrator. Dette brukernavnet og passordet brukes for å registrere brukeren første gangen. Brukeren taster dette inn i en egen meny på enheten som skal brukes, og får da tilgang til alle sine kurs.

Synkronisering fra server

Aktør	Bruker
Hensikt	Å holde brukeren oppdatert med de nyeste PLU-kodene og kursene.
Beskrivelse	For at brukeren alltid skal ha en oppdatert database med PLU-koder, og for å alltid ha alle kursene som er relevante tilgjengelig, må serveren pushe endringer til alle enheter som trenger dem. Dette gjøres i bakgrunnen, og er i utgangspunktet ikke noe brukeren trenger tenke på, med mindre det er større datamengder som skal dyttes. Brukeren skal da informeres om dette.

Synkronisering av brukerdata

Aktør	Bruker
Hensikt	Å holde med oppdatert informasjon om brukeren.
Beskrivelse	For at administratorer skal kunne se statusen til brukerene, må informasjonen fra brukerene dyttes over til serveren. Dette gjelder da hovedsaklig informasjon om hvor mange poeng brukeren har oppnådd i de forskjellige kursene. Dette vil skje i bakgrunnen når en bruker avslutter et kurs. Da vil gjeldende status lastes opp, eksempelvis at brukeren har fullført halvparten av kurset.

Delta på kurs

Aktør	Bruker
Hensikt	Å delta på kurs i regi av kursholder eller administrator.
Beskrivelse	Brukeren skal kunne delta på relevante kurs som en kursholder eller administrator har satt opp. Brukeren må være påmeldt kurset for å ha tilgang. Brukeren trykker da på knappen "Kurs og konkurranser" i hovedmenyen på applikasjonen. Brukeren vil deretter få opp en liste over sine kurs, og kan trykke på dem for å få presentert relevant informasjon om disse. Brukeren trykker så "start" eller "fortsett", og kan da starte eller fortsette på dette kurset.

Delta i konkurranse

Aktør	Bruker
Hensikt	Å delta i konkurranser i regi av kursholder eller administrator
Beskrivelse	Brukeren skal kunne delta i konkurranser som er laget av en kursholder eller administrator. Brukeren velger "Kurs og konkurranser" fra hovedmenyen. Deretter skal brukeren velge fanen "Konkurranser". Brukeren vil da få presenter alle relevante konkurranser, og kan velge en av disse. Når brukeren har valgt en konkurranse, skal det velges "Start" for starte denne

Starte trening

Aktør	Bruker
Hensikt	Å trene på alle tilgjengelige PLU-koder når som helst
Beskrivelse	En bruker kan velge å trene på PLU-koder når som helst. En bruker trenger ikke være registrert for å gjøre dette. Brukeren velger "Trening" fra hovedmenyen i applikasjonen, og velger så hva som er ønskelig å trene på. Dette kan være bestemte PLU-grupper eller alle PLU-koder. Brukeren vil så få presentert tilfeldig valgte oppgaver innenfor valgte PLU-koder

Sjekk status på kurs

Aktør	Bruker
Hensikt	Å se hvordan statusen er på et bestemt kurs
Beskrivelse	En bruker skal til enhver tid kunne se sin status innenfor alle tilgjengelige kurs. Brukeren velger "Kurs og konkurranser" fra hovedmenyen, og får så presentert en liste over tilgjengelige kurs. Brukeren velger så det kurset som det ønskes informasjon om. Brukeren vil da få en oversikt over statusen på kurset. Dette kan være om kurset er bestått eller ikke, og hva slags poengsum brukeren har

Sjekk rankingliste

Aktør	Bruker
Hensikt	Å vise brukeren hvilke poengsummer andre brukere har oppnådd.
Beskrivelse	Det skal kunne arrangeres konkurranser via applikasjonen. For å se rankingliste innenfor en konkurranse, velger brukeren "Mine konkurranser" fra hovedmenyen. Brukeren velger deretter konkurransen som skal undersøkes. Brukeren velger så "Rankingliste" for å se poengsummen til de brukerene med høyest poengsummer.

Søk etter vare

Aktør	Bruker, database
Hensikt	Søke etter en vare for å finne PLU-kode.
Beskrivelse	I hovedmenyen vil det være en søkeknapp som brukeren kan klikke på. Brukeren får da muligheten til å skrive inn tekst for å søke etter en vare. Denne teksten kan være deler av en PLU-kode, varenavn, eller varekategori. Det vil også være en widget som brukeren kan legge inn på hjemskjermen sin.

2.1.2 Høynivå beskrivelser - Web-applikasjon

Innlogging, Server

Aktør	Administrator
Hensikt	Logge seg inn på web-applikasjon.
Beskrivelse	Administrator går inn på web-applikasjonen, og skriver inn forhåndsdefinert brukernavn og passord.

Importer PLU

Aktør	Administrator, Server
Hensikt	Importering av PLU-koder.
Beskrivelse	Når administrator er innlogget vil han ha mulighet til å importere PLU-koder fra fil. Dette vil så legges inn i databasen, og bli tilgjengelig for brukere av mobil-applikasjonen.

Legg til / endre PLU

Aktør	Administrator, Server
Hensikt	Manuelt legge inn eller endre PLU.
Beskrivelse	Ved å klikke seg inn på PLU-oversikt, kan administrator få liste over alle PLU-koder med tilhørende varer. Her vil administratoren ha mulighet til å klikke på "Legg til", eller klikke inn på en vare for å endre varen. Her kan administratoren endre navn, kategori, PLU-kode eller bilde av varen. Disse endringene vil så bli synkronisert til brukere av mobil-applikasjonen.

Opprett kurs

Aktør	Administrator, Server
Hensikt	Sette opp kurs for registrerte brukere av mobil-applikasjon.
Beskrivelse	Administratorer vil ha mulighet til å opprette kurs ved å klikke inn på "Opprett kurs". Her vil administratoren kunne velge varer, enten basert på kategorier, eller ved å manuelt velge hvilke PLU-koder kurset skal inneholdet. Sistnevnte vil være spesielt aktuelt ved introduksjon av nye varer, eller for kurs beregnet på spesifikke ansatte. Når administratoren har opprettet kurset vil han få spørsmål om han vil assosiere kurset til brukere, butikker eller regioner.

Endre / slett kurs

Aktør	Administrator, Server
Hensikt	Endre detaljer på opprettede kurs, som PLU-koder.
Beskrivelse	Administratoren vil også kunne endre og slette et kurs. Dette gjøres ved å klikke inn på listen over kurs, og så velge kurset han har lyst til å endre eller slette. Hvis han vil endre er aktuelle valg er tidsfrister, tilhørende PLU-koder og assosieringer. Disse endringene vil så lastes opp til mobil-applikasjonen hos de aktuelle brukerne.

Legg til / endre / fjern medarbeider

Aktør	Administrator, Server
Hensikt	Opprette, endre eller fjerne bruker til medarbeider.
Beskrivelse	Ved å velge "Legg til medarbeider" vil en administrator manuelt kunne lage en bruker for en medarbeider. Aktuell info om medarbeider er: ID / navn, mobilnummer, e-post, og tilhørende butikk. Når brukeren er opprettet, vil det sendes en melding eller e-post til brukeren med et engangspassord, som medarbeideren vil kunne bruke for å registrere seg inne i mobil-applikasjonen. Administrator vil også kunne endre eller fjerne en medarbeider ved å enten søke seg til medarbeideren på aktuell info, eller bla seg gjennom region og butikk for å finne den aktuelle medarbeideren.

Sjekk status på kurs

Aktør	Administrator, Server
Hensikt	Sjekke overordnet status på opprettet kurs.
Beskrivelse	Når administrator har klikket seg inn på listen over kurs, vil det på hvert enkelt kurs stå angitt i prosent hvor mange brukere som har fullført kurset. Administrator kan så klikke seg inn på kurset for å få en bedre oversikt. Her vil det være en liste over alle medarbeidere og deres poeng på dette kurset.

Sjekk progresjon til medarbeider

Aktør	Administrator
Hensikt	Sjekke status på kurs for en spesifikk medarbeider.
Beskrivelse	Ved å klikke seg inn på listen over medarbeidere, vil administrator få listet opp alle medarbeiderne sine. Her vil man videre kunne klikke seg inn på en spesifikk medarbeider, og få en oversikt over alle kursene denne medarbeideren har deltatt på, samt poeng på disse kursene.

Legg til administrator

Aktør	Administrator, Server
Hensikt	Legge til administratorer som har tilgang til nevnte funksjoner.
Beskrivelse	Administratorer som har tilgang kan legge til andre administratorer. Dette gjøres ved å klikke inn på listen over administratorer, og "Legg til administrator". Dette vil fungere veldig likt som "Legg til medarbeider", men det kan her legges inn rettigheter til den nye administratoren, inkludert hvilke regioner / butikker han har tilgang til, og hvorvidt han kan opprette kurs.

2.1.3 Detaljerte beskrivelser - Mobil-applikasjon

Delta på kurs

Aktør	Bruker
Hensikt	Å delta på kurs i regi av kursholder eller administrator.
Pre-betingelse	Brukeren må være registrert i applikasjonen. Applikasjonen må være synkronisert med relevant informasjon.
Post-betingelse	Resultatet må være lagret. Lokalt hvis brukeren er uten tilgang til internett, og synkronisert mot serveren ved tilgang til internett.
Spesielle krav	Brukeren må være oppmeldt på kurset av en kursholder.
Detaljert hendelsesforløp	• Brukeren velger "Kurs og konkurranser" fra hovedmenyen. • Brukeren velger korrekt kurs fra sin liste. • Brukeren velger "Start" eller "Fortsett", alt ettersom om brukeren tidligere har startet kurset. • Brukeren svarer på oppgavene som blir presentert. • Poengsum oppdateres etter hvert spørsmål som blir besvart. • Brukeren avslutter kurset når det er ønskelig. • Status på kurset lagres, og synkroniseres hvis det er tilgang på internett, automatisk når brukeren avslutter.
Alternative scenarioer	Ingen

Starte trening

Aktør	Bruker
Hensikt	Å trene på alle tilgjengelige PLU-koder når som helst.
Pre-betingelse	Applikasjonen er blitt synkronisert minst en gang for å inneholde PLU-koder.
Post-betingelse	N/A
Spesielle krav	N/A
Detaljert hendelsesforløp	• Brukeren velger "Trening" fra hovedmenyen. • Brukeren velger ønsket kategori å trene på. Dette kan være "Alle PLU-koder" eller bestemte PLU-kodegrupper. • Brukeren velger vanskelighetsgrad. Velge korrekt PLU-kode fra et utvalg, eller taste PLU-koden uten å ha noen å utelukke. • Brukeren løser så mange oppgaver som ønskelig. • Brukeren kan avslutte når det er ønskelig. • Ingen lagring eller synkronisering er nødvendig.
Alternative scenarioer	Ingen

2.1.4 Detaljerte beskrivelser - Web-applikasjon

Legg til / endre PLU

Aktør	Administrator, Server
Hensikt	Manuelt legge inn eller endre PLU.
Pre-betingelse	Administrator må være logget inn.
Post-betingelse	Ny / endret PLU synkroniseres til mobil-applikasjonen.
Spesielle krav	Ingen

Detaljert hendelsesforløp	• Administrator velger "Varer" fra hovedmeny.
	• Administrator trykker på "Ny".
	• Administrator fyller inn:
	  ◦ Varenavn
	  ◦ Varenummer
	  ◦ Varekategori
	  ◦ PLU-kode
	• Administrator laster opp et bilde av varen ved å trykke på "Last opp", og velger et lokalt tilgjengelig bilde.
	• Administrator trykker på "Lagre".

Alternative scenarioer	• Administrator velger "Vareliste" fra hovedmeny.
	• Administrator eventuelt søker etter vare på en av følgende:
	  ◦ Varenavn
	  ◦ Varenummer
	  ◦ Varekategori
	  ◦ PLU-kode
	• Administrator velger vare, resten fortsetter som hendelsesforløp.

Opprett kurs

Aktør	Administrator
Hensikt	Sette opp kurs og konkurranser for registrerte brukere av mobil-applikasjon.
Pre-betingelse	Administrator er logget inn.
Post-betingelse	Kurs synkroniseres til assosierte brukere, hvis noen.
Spesielle krav	Minst 1 vare eksisterer

Detaljert hendelses-forløp	• Administrator velger "Kurs" fra hovedmeny.
	• Administrator trykker på "Ny".
	• Administrator fyller inn:
	  ◦ Kursnavn
	  ◦ Kurstekst
	  ◦ Vanskelighetsgrad
	  ◦ Konkurranse
	  ◦ Antall repetisjoner
	  ◦ Startdato
	  ◦ Sluttdato
	  ◦ En av følgende:
	  · Varekategori
	  · Spesifikke varer
	• Administrator trykker "Lagre".
	• Administrator får spørsmål om han vil assosiere kurset på en av følgene:
	  ◦ Region
	  ◦ Butikk
	  ◦ Spesifikke deltagere
	• Administrator velger en eller ingen og trykker "Ferdig".

Alternative scenarioer	Ingen

Legg til / endre / fjern medarbeider

Aktør	Administrator
Hensikt	Opprette, endre eller fjerne bruker til medarbeider.
Pre-betingelse	Administrator er logget inn, ved endring eller fjerning av bruker må denne allerede eksistere.
Post-betingelse	Brukernavn og engangspassord sendes til medarbeider.
Spesielle krav	Bruker med samme brukernavn kan ikke eksistere.

Detaljert hendelsesforløp	• Administrator velger "Medarbeidere" fra hovedmeny.
	• Hvis ny:
	  ◦ Administrator trykker på "Ny":
	  ◦ Administrator fyller inn følgene:
	  · Fullt navn
	  · Brukernavn
	  · Mobilnummer
	  · E-post
	  · Tilhørende butikk
	  ◦ Administrator trykker på "Lagre".
	  ◦ Tilfeldig engangspassord genereres.
	• Hvis endre:
	  ◦ Administrator søker eventuelt på medarbeider etter en av følgende:
	  · Fornavn
	  · Etternavn
	  · Brukernavn
	  · Mobilnummer
	  · E-post
	  · Butikk
	  · Region
	  ◦ Administrator velger en medarbeider, og trykker på denne.
	  ◦ Administrator trykker på "Endre".
	  ◦ Administrator oppdaterer relevant informasjon som listet over.
	  ◦ Administrator trykker eventuelt på "Genererer nytt passord", hvis nytt passord er nødvendig for medarbeider.
	  ◦ Administrator trykker på "Lagre".
	• Hvis slette:
	  ◦ Administrator velger en medarbeider.
	  ◦ Administrator trykker på "Slett".
	  ◦ Bekreftelsesvindu popper frem.
	  ◦ Administrator bekrefter sletting, eller trykker "Avbryt".

Alternative scenarioer	Ingen

Sjekk progresjon til medarbeider

Aktør	Administrator
Hensikt	Sjekke status på kurs for en spesifikk medarbeider.
Pre-betingelse	Administrator er logget inn og brukeren må være meldt opp på kurs.
Post-betingelse	Ingen
Spesielle krav	Ingen
Detaljert hendelsesforløp	• Administrator velger "Medarbeidere" fra hovedmeny. • Administrator søker eventuelt fram til medarbeider, og velger denne. • Administrator får opp medarbeiderinfo, inkludert liste over kurs med progresjon.
Alternative scenarioer	Ingen

2.2 Kvalitetsmessige og operasjonelle krav

Pålitelighet

- All funksjonalitet for både webgrensesnittet og mobil-applikasjonen skal fungere feilfritt. Unntak som krever brukerinteraksjon vil bli opplyst til brukeren, e.g. brukeren må være tilkoblet internett for å synkronisere data. All funksjonalitet vil før ferdigstilling, gjennomgå ekstensiv white-box testing for å sikre dette.
- Alle funksjoner skal utføre sine oppgaver, og ikke forårsake uønskede effekter på løsningen.
- En bruker skal aldri få informasjon eller tilgjengelighet som ikke tilhører den spesifikke brukeren.
- Mobil-applikasjonen skal alltid opprettholde full funksjonalitet uavhengig av tilkobling til internett.

Effektivitet

- Mobil-applikasjonen skal være effektiv nok til at den fungere på alle mobile enheter med minimum Android API 16.

Standardiserte krav

- Appcompat library skal brukes for Android, da dette sørger for at applikasjonen vil fungere også på eldre modeller.
- Web-applikasjonen skal støttes fullt av Internet Explorer 11, samt versjoner av Chrome 48 og Firefox 43.

Personvern og sikkerhet

Ved sletting av en bruker, både på webgrensesnittet og på mobil-applikasjonen, skal all informasjon om vedkommende bruker også fjernes fra databasen.

Brukergrensesnitt

- Mobil-applikasjonen skal designes slik at en bruker aldri behøver mer enn maks fire trykk for å komme til ønsket funksjon fra hovedmenyen .
- Material Design guidelines fra Google skal følges på Android, for å sikre en best

mulig brukeropplevelse. LINK: https://www.google.com/design/spec/material-design/introduction.html

Brukeropplæring / hjelpefunksjon

- Ved oppstart av mobil-applikasjonen for første gang:
  - Skal en veiviser forklare de viktigste funksjonene i applikasjonen.
  - Skal brukeren bli tilbudt muligheten for å prøve de forskjellige spill-modusene.

2.3 Backlog

- DB-design - Konseptuellt design av DB.
- App - Material design - Undersøke regler for material design.
- Web/server - Importering av varer fra CSV - Laste opp .csv på web, og importere dette i DB.
- Web - Legge til/slette vare - Kunne legge til og slette en vare fra DB.
- Server - Legge inn medarbeider - Når en medarbeider legges til i web-grensesnittet, skal han lagres i DB.
- Web - Legge til medarbeider - Funksjonalitet på web-grensesnitt for å legge til en medarbeider
- Server - Hente medarbeider - Hente ut medarbeider-info fra DB.
- Web - Endre / slette kurs - Det skal gå ann å endre eller slette kurs som allerede eksisterer.
- Web - Assosiere kurs til brukere, butikk, slag eller region - Kurs skal kunne assosieres slik at kun aktuelle brukere får tilgang til det.
- App - Google Cloud Messaging - Undersøking og implementering.
- App - Innloggingssystem - Logge seg inn med en brukerkonto, bruke Google sign-in.
- App - belønningsystem / poengsystem / achievements - Finne/lage algoritmer for poengsystem, finne på achievements og implementere disse.
- App - Widget for søking - Widget som kan legges på homescreen for enkel tilgang til søk på varer.
- App - Søkefunksjon - Søke etter varer, enten på navn eller PLU-kode.
- App - Rankinliste / scoreboard - Etter fullført kurs/konkurranse.
- App - Brukerstatistikk - Funksjoner for å samle statistikk om brukeren av applikasjonen.
- Server - Legge til / hente region / butikk - legge inn og hente regioner og butikker fra DB.
- Web - Bootstrap dashboard - Dashboard for å vise statistikk til administratorer.

3 Design

I dette kapittelet vil vi ta for oss designet av vår løsning. Vi vil ta for oss både høynivå og de detaljerte beskrivelsene, og se hvordan disse henger sammen. Vi vil også vise hvilke valg vi har tatt når det kommer til design av brukergrensesnitt.

3.1 Use cases

Vi har laget Use Case-diagrammer for beskrivelsene i forrige kapittel. Her gir vi en grafisk presentasjon av disse beskrivelse, og viser hvilke roller som er involverte i de forskjellige handlingene.

3.1.1 Diagram for mobil-applikasjon

I dette Use Case-diagrammet blir det vist hvilke handlinger som skal kunne utføres på mobil-applikasjonen. Det bli også vist hvilke roller og enheter som er involvert i disse handlingene.

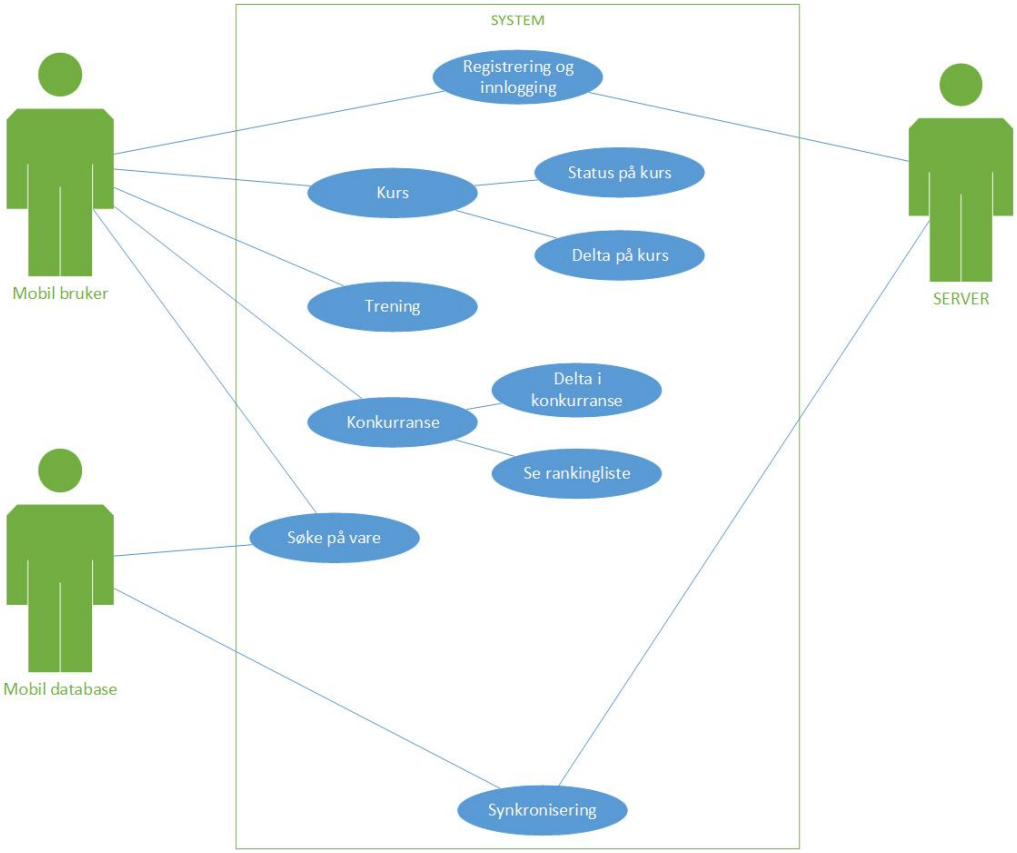


Figure 1: Diagram for mobil-applikasjon

3.1.2 Diagram for administrator

I dette Use Case-diagrammet blir det vist hvilke handlinger en administrator skal kunne utføre på webgrensesnittet. Det blir også vist hvilke roller og enheter som er involvert i disse handlingene.

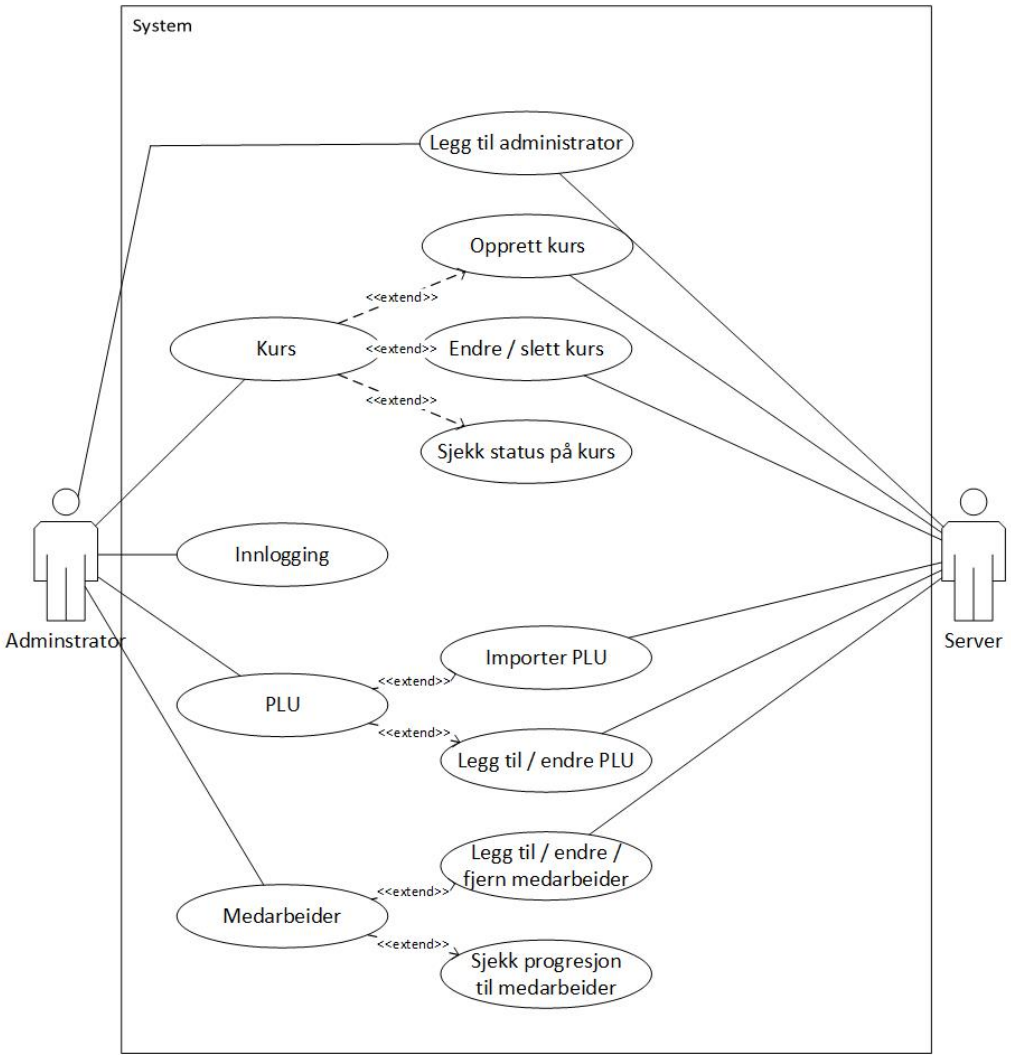


Figure 2: Diagram for administratorer

3.2 Database

Databasen ble designet ut ifra oppgavebeskrivelsen i kapittel 1.1.3. Fra beskrivelsen av funksjonalitet, kan vi se at det er nødvendig å ha tabeller for brukere, poengsystem, varer, varekategorier og kurs. Figur 3 illustrerer det ferdige database-designet. "PLUUser" er tabellen over brukere, og har kolonner for brukerinformasjon, samt userId og instanceToken. Opprinnelig skulle vi bruke Google Sign-in for å håndtere bruker-innlogging, og derfor ønsket vi å lagre user-id fra Google i denne tabellen. Google Cloud Messaging (GCM) skulle brukes for Push Notifications, og vi skulle derfor lagre Instance Token for håndtering av GCM.

"PLUItem" er varelisten, hvor hver rad inneholder en unik PLU-kode og et varenavn, og eventuelt et bilde og en beskrivelse. Den knyttes også til en "PLUType", som er varekategoriene. Et eksempel på en slik kategori er bakevarer. "PLUCourse" er tabellen som lagrer kurs, og relevant info om kurset. Denne kan knyttes enten til en varekategori eller til èn eller flere "PLUCourseItem" som deretter er knyttet til en vare. Hvis kurset er knyttet til en varekategori, vil kurset oppdateres hvis det blir lagret nye varer i denne kategorien.

Når en bruker har tatt et kurs, blir resultatene lagret i "PLUScore". Resultatene fra de enkelte oppgavene i kurset blir lagret i PLUScoreCourseItem, hvor det for hver oppgave lagres en poengsum, og tiden som ble brukt.

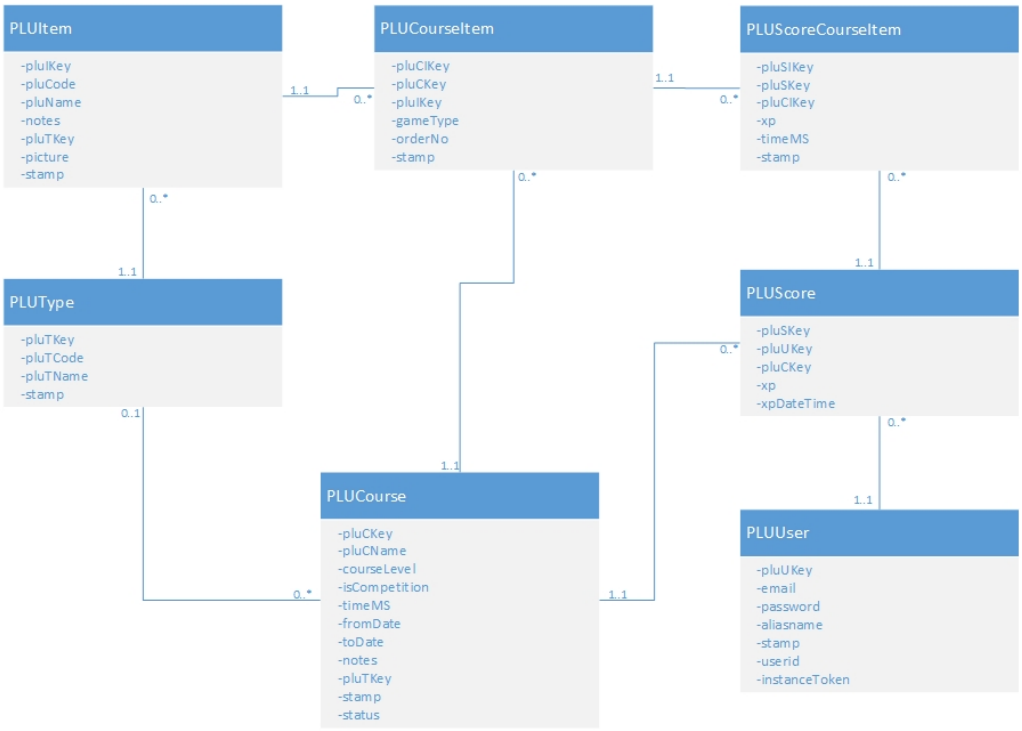


Figure 3: Endelig database-design

3.3 Brukergrensesnitt for web

Dette kapittelet tar for seg det grafiske brukergrensesnittet for web.

3.3.1 Planleggingen

Da vi planla webgrensesnittet, bestemte vi at dette skulle være enkelt og intuitivt. Vi ville holde oss til et design som gjorde at det skulle være lett å forstå hvordan løsningen skulle brukes. Vi bestemte oss for å benytte oss av Bootstrap. Dette er et system som inneholder et sett med verktøy for utvikling av responsive internettsider, og inneholder knapper, bannere og diverse moduler en kan benytte seg av. Vi vil nå vise biter av løsningen, og forklare tankegangen bak designet.

3.3.2 Fargevalg

Siden dette skal benyttes av Coop Norge, valgte vi å ha en litt mørk blå farge som basis. Vi har også benyttet oss av fargekoding på tilbakemeldinger til brukere, samt merking av

spesielle rader i enkelte tabeller. Vi vil forklarer mer om dette senere i rapporten. I alle disse tilfellene har vi benyttet oss av et sett med farger som er godt synlige, og som også fungerer for fargeblinde.

3.3.3 Hovedmeny

Vi ville ha en hovedmeny som var enkel og oversiktlig å benytte seg av. Vi har ikke benyttet oss av mange sider for brukerne, og trengte derfor ingen innviklet meny. Vi endte opp med å benytte oss av en enkel navigasjonslinje på toppen av nettsiden, hvor vi har en knapp for å komme seg til hovedsiden, samt en knapp til hver av de totalt tre andre sidene på nettsiden. Denne menyen er alltid tilgjengelig, uansett hvor brukeren befinner seg.

3.3.4 Hovedsiden

Hovedsiden skal være en enkel og oversiktlig side, hvor brukeren kan få relevant informasjon, samtidig som det skal være mulig å enkelt gå videre til andre områder av løsningen. Vi har her valgt å benytte oss av paneler. Det er ett panel for hver side i løsningen:

- Kurs
- Varer
- Ansatte

Hver av disse panelene viser informasjon om respektive område, og har mulighet til å ta brukeren direkte til det ønskede området. Vi har brukt forskjellige farger på panelene for å gjøre det enklere for brukeren å skille dem.

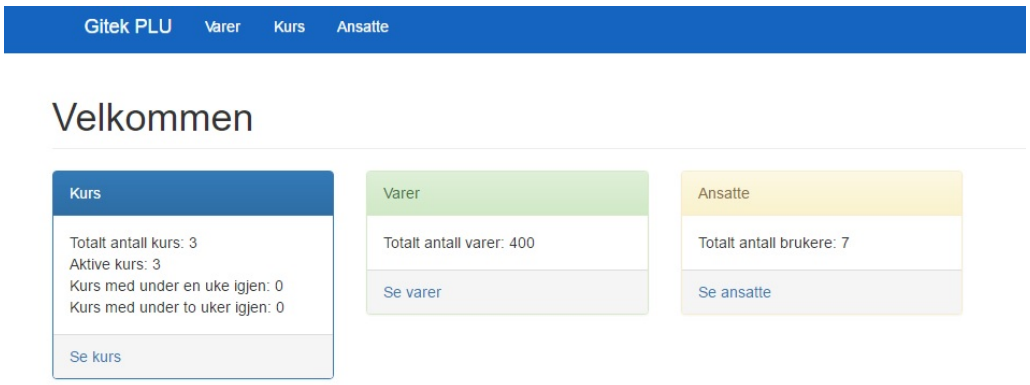


Figure 4: Hovedside med navigasjonsbar og paneler på web

3.3.5 Kursoversikt

Kursoversikten skulle være oversiktlig, og det skulle være lett å finne fram til aktuelle kurs. Her er det benyttet tabeller, en for kurs som ikke er avsluttet enda, og en for kurs som er avsluttet. Det gjør at tabellen ikke inneholder irrelevante kurs. I tilegg er alle radene i tabellen for pågående kurs fargekodet, slik at det skal gå fort å få et overblikk over hvilke kurs som bør følges opp. Begge tabellene kan også sorteres på ønskede kriterier, og er fullt søkbare for å gjøre den enda enklere for brukeren å finne det spesifikke

kurset de ønsker. I tabellen for pågående kurs er det også mulighet for å endre eller slette ønsket kurs, samt se utvidet informasjon. Alt er lett tilgjengelig for brukeren.

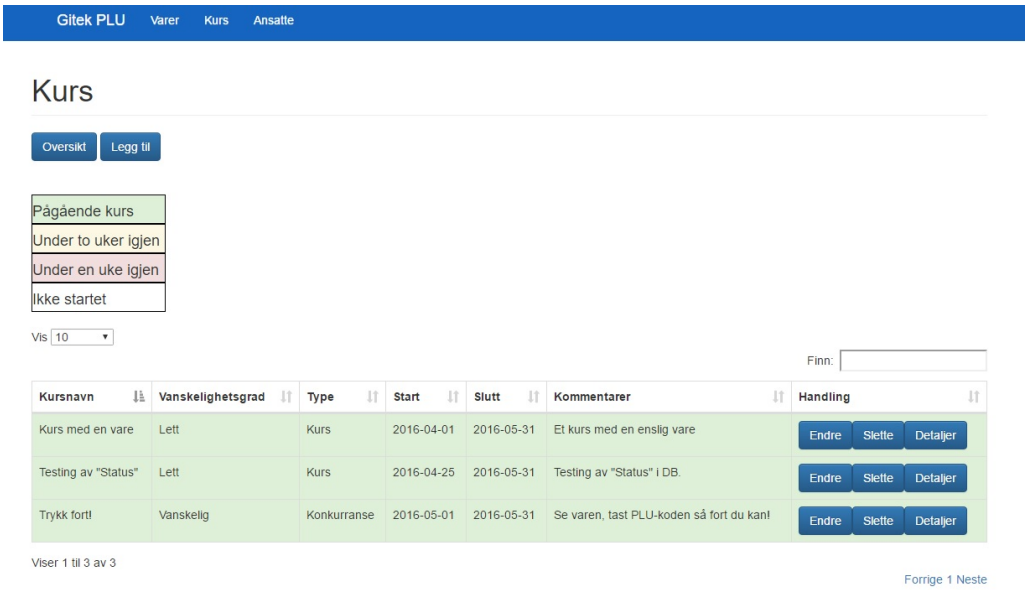


Figure 5: Kursoversikt på web

3.3.6 Legge til kurs

For å legge til kurs er det blitt benyttet en HTML form. Dette er rett og slett et skjema for internettsider, hvor brukeren fyller inn all informasjonen. Dette gjør at det er veldig rett fram hva brukeren skal gjøre. Det er brukt en modul for å velge dato, slik at brukeren slipper å manuelt taste inn datoene for kursene. Det kommer opp en egen meny for dato, og brukeren bare velger de datoene som er ønskelige. Hvis brukeren glemmer å taste inn informasjon, blir ikke kurset lagret, men brukeren blir varslet om hva som mangler. Brukeren kan da fylle inn manglende informasjon, og prøve igjen. Dette gjør at brukeren alltid er sikker på at ingen informasjon blir uteglemt.

For å legge til varer i kursene, er det igjen blitt benyttet tabeller. Det er en tabell som inneholder alle tilgjengelige kategorier, en tabell med alle tilgjengelige varer, og en tabell med alle varene som er lagt til i kurset. Brukeren kan da velge å legge til en og en vare, eller hele kategorier av varer. Brukeren får alltid tilbakemelding på alle valgene som blir gjort. Dette blir gjort for å sikre at brukeren alltid vet at handlingen ble utført. Her også er det benyttet logiske farger, med grønt når varer blir lagt til, og rødt når varer blir fjernet. Det blir også bedt om en bekreftelse før kurset blir lagret.

3.3.7 Sletting av kurs

Det skal være enkelt å slette et kurs. Dette gjøres kun ved at brukeren trykker på "Slette"-knappen i tabellen. Brukeren blir da bedt om å bekrefte at akkurat det kurset skal slettes.

3.3.8 Detaljer om kurs

Det er ønskelig at brukeren skal kunne få utvidet informasjon om alle kurs. Vi ønsket å få samlet så mye som mulig på et sted, samtidig som det skulle være oversiktlig. Vi har derfor valgt å lage lister med informasjon. Informasjonen er delt opp i flere lister for å

samle det som logisk hører sammen:

- Generell informasjon
- Informasjon om varer
- Informasjon om brukere
- Informasjon om brukere innenfor spesifikke regioner

Dette gir tilsammen en samling med data som brukeren kan bruke etter eget ønske. Her finnes informasjon om kurset med navn, vanskelighetsgrad, start- og sluttdato. Hvor mange varer som er i kurset, samt navnet på alle varene. Brukeren får også en oversikt over hvor mange deltakere som har deltatt på kurset, kallenavnet til alle sammen, hvem som har høyest poengsum og hvor mange poeng den personen har. Denne informasjonen er også tilgjengelig for hver enkelt region, i tilegg til en egen oversikt over personer som ikke er ansatte i Coop. Alle listene kan minimeres, slik at det ikke er for mye informasjon på skjermen om gangen.

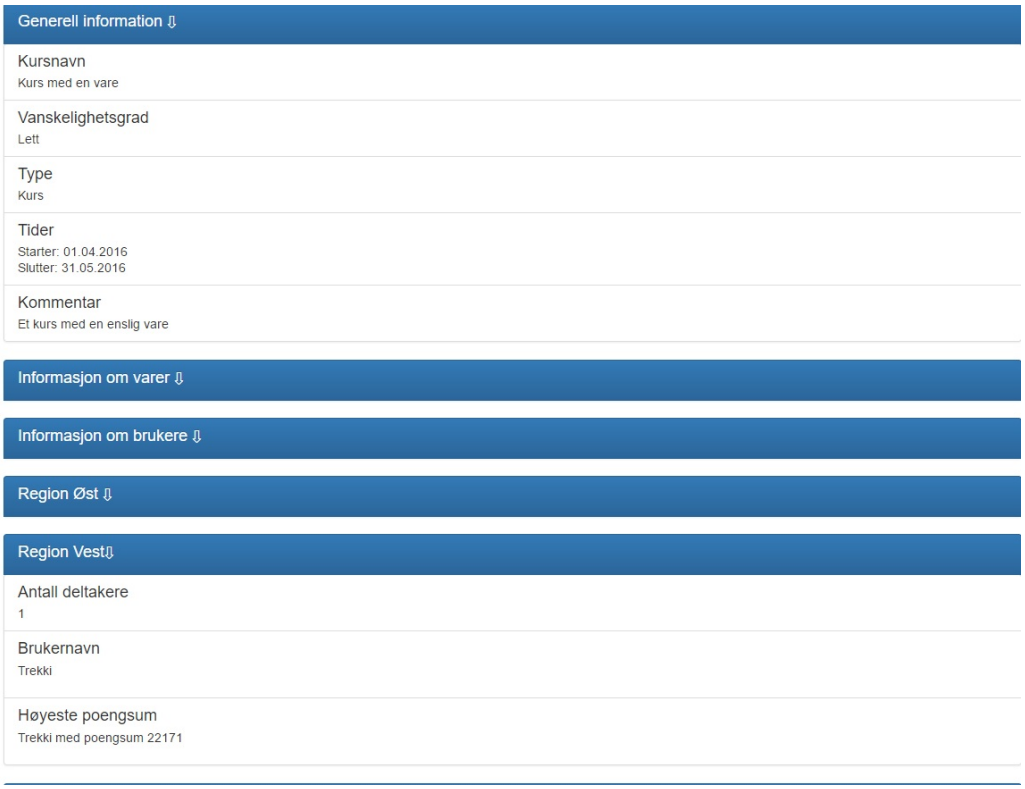


Figure 6: Detaljert kursoversikt på web

3.3.9 Brukeroversikt

Vi syntes det skulle være enkelt å få en oversikt over alle brukere. Her er det også laget en tabell. Denne inneholder alle brukere, og viser kallenavn og epostadresse. Vi har valgt å ikke benytte oss av registrering av brukere via webgrensesnittet, da det er mest logisk, og absolutt enklest for alle parter, at all registrering skjer via mobil-applikasjonen. Vi har derimot lagt inn muligheten for å slette brukere via webgrensesnittet. Dette gjøres på samme måte som for kurs. Det er en enkelt knapp i tabellen som sier "Slett". Brukeren

blir da bedt om å bekrefte valget, og brukeren blir slettet.

3.3.10 Varer

Gitek ønsker et fullverdig system for å administrere varer og PLU-koder vi webgrensesnittet. Vi ville da gi dem en god oversikt over alle varene, gi en godt synlig indikasjon på om denne varen hadde bilde eller ikke, og mulighet til å endre eller legge til nye varer. Her er det også blitt benyttet en tabell, med full søkefunksjon og sortering. Vi viser navnet på varen, PLU-koden, et miniatyrbilde hvis varen har bilde, og knapper for å endre eller slette en vare. Det er også en egen knapp for å legge til nye varer.

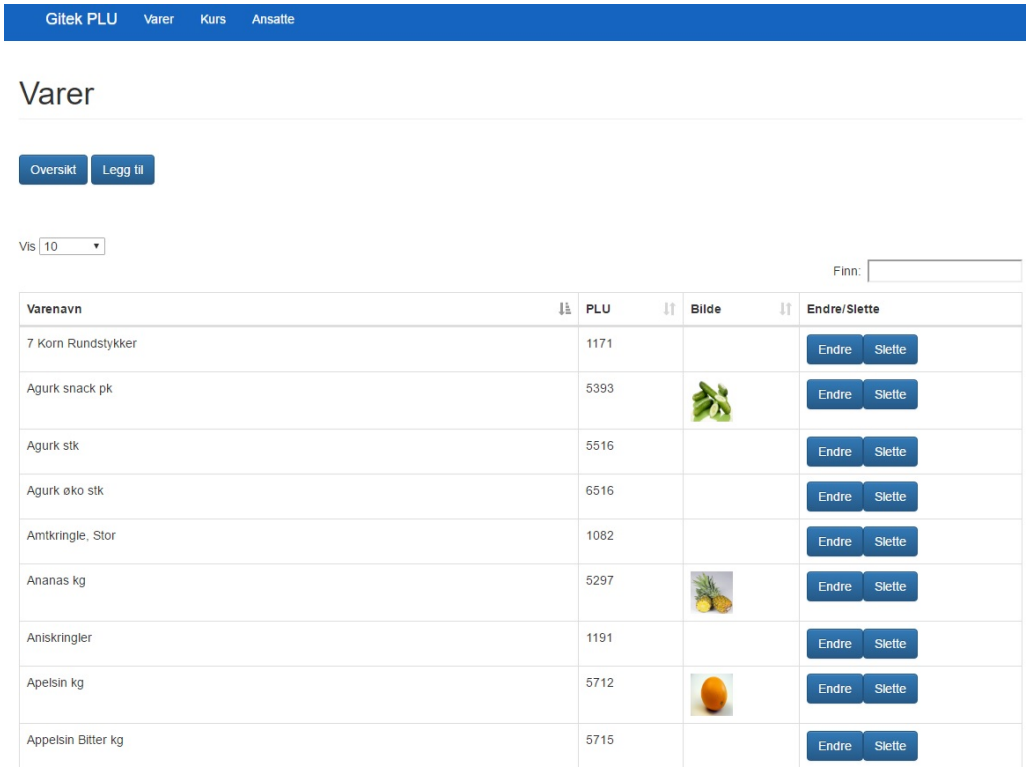


Figure 7: Vareoversikt på web

3.3.11 Legge til og endre varer

Å legge til varer skal være så enkelt som mulig. Her har vi benyttet oss av det samme systemet som vi har for å legge til kurs. Det er et skjema som brukeren fyller ut, og får tilbakemelding hvis det er noe som mangler. Det som skiller dette skjemaet fra skjemaet til kursene, er at her har brukeren også muligheter til å laste opp et bilde av varen. Ved å velge et bilde fra datamaskinen sin, vil brukeren få opp en forhåndsvisning av bildet i skjemaet. Dette gjør at brukeren får en synlig tilbakemelding på at korrekt bilde er blitt valgt. Ved å endre en vare benyttes det samme skjemaet. Den eneste forskjellen er at skjemaet da er fylt ut med informasjonen til den eksisterende varen, og brukeren bare endrer det som ønskes. Dette gir minst mulig jobb for brukeren.

3.4 Brukergrensesnitt på Android

Ved design av brukergrensesnitt på Android, valgte vi å følge Material Design for Android. I prosessen for å lage et brukervennlig grensesnitt, valgte vi å ha store tydelig knapper, med stor tekst. Dette var for å sørge for at applikasjonen også skulle kunne brukes av svaksynte. Vi bestemte oss for å bruke et blått fargetema, av samme grunn som på webgrensesnittet.

3.4.1 Hovedmeny

Vi designet hovedmenyen med tanke på at den skal være lett å forstå. Alle knappene skal ha logisk tekst, og være store slik at det ikke blir vanskelig å trykke på dem. De skal også være adskilte, slik at det blir lett for brukeren å skille knappene fra hverandre, og vanskelig å trykke feil. Vi ønsket også å ha alle knappene tilgjengelige på skjermen til enhver tid, slik at vi unngikk å ha gjemte knapper utenfor skjermbildet. Fargene er et gjennomgående tema, og vi har her, som på webløsningen, valgt en blåfarge.

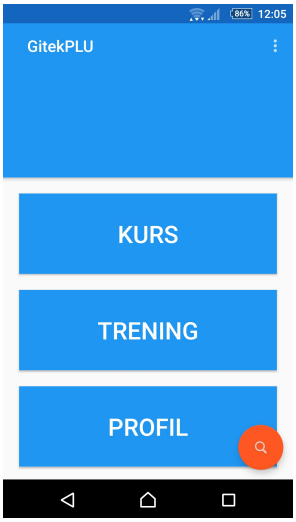


Figure 8: Hovedmeny for Android-applikasjonen

Hovedmenyen ble da oversiktlig og ren, med minst mulig distraksjon for brukeren. Brukeren har også en snarvei for et oppslagsverk, merket med et dyp oransje forstørrelsesikon nederst til høyre på skjermen. Dette gir brukeren rask tilgang til en søkefunksjon for å lete etter varer og PLU-koder.

3.4.2 Kursoversikt

Kursoversikten er delt inn i tre deler. Det er tre forskjellige faner som brukeren kan velge mellom. Disse inneholder henholdsvis pågående kurs, konkurranser, og utgåtte kurs og konkurranser. Dette gjør at brukeren slipper å bli vist unødvendig informasjon.

Etterhvert som brukeren besvarer oppgaver i kurs, vil hele knappen bli gradvis grønn. Grønnfargen starter til venstre på knappen, og går mot høyre. Det står også hvor mange prosent av kurset som er fullført. Dette gir brukeren en klar indikasjon på hvor mye som er igjen av kurset. Alle knappene er utformet på samme måte som hovedmenyen, noe som skaper en helhetlig opplevelse av applikasjonen.

Tittelen på alle kursene er plassert som tekst på knappene. Dette gjør at brukeren lett kan gjenkjenne kursene. Ved å trykke på en av knappene blir brukeren presentert med informasjon om det angitte kurset. På dette skjermbildet er det også to faner, informasjon og resultater. På fanen for informasjon vises informasjon fra administrator.

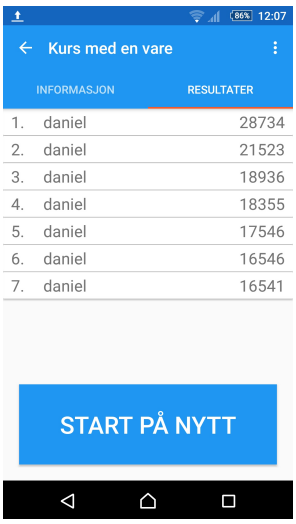


Figure 9: Resultatliste for kurs

Hvis brukeren har valgt et kurs, vil fanen for resultater vise alle poengsummene brukeren har oppnådd på kurset. Hvis brukeren har valgt en konkurranse, vil fanen for resultater vise den høyeste poengsummen for alle brukere som har deltatt på konkurransen. Denne listen er sortert i synkende rekkefølge.

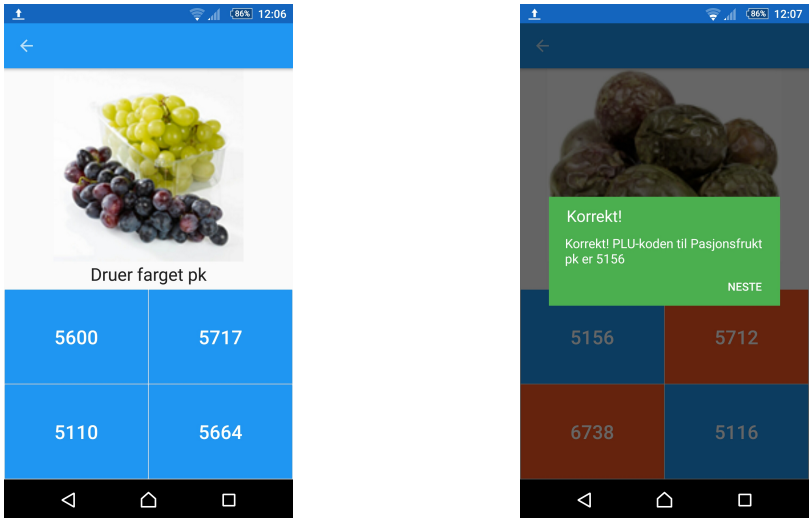
3.4.3 Treningsmodus

Treningsmodus er en måte for brukeren å trene på PLU-koder på, uten å være avhengig av at en administrator har opprettet et kurs. Brukeren kan selv velge varekategorier, vanskelighetsgrad, og antall spørsmål. Vi har forhåndsdefinert antall spørsmål til henholdsvis 5, 10, 15 og uendelig antall spørsmål. Dette blir valgt med en slidebryter. Vanskelighetsgraden velges også med en slidebryter, og mulige valg er lett, middels og vanskelig.

Når brukeren har gjort alle valgene og startet spillet, fungerer resten på samme måte som kursene. Forskjellen på kurs og trening, er at ingen poengsum blir generert ved trening.

3.4.4 Spill - Velg PLU

Vi designet første spill-modusen med utgangspunkt i at den skulle være den enkleste, og derfor passe spesielt godt for personer som ikke er kjent med PLU-kodene.



(a) Oppgaven (b) To feil og et korrekt svar

Figure 10: Spillmodus: Velg PLU

I figuren over viser vi det endelige designet. Knappene har samme stil som hovedmenyen her også, igjen for å skape et helhetlig inntrykk av applikasjonen. Det er et stort bilde av varen, slik at brukeren ikke skal ha noe problem med å gjenkjenne varen. I tillegg står navnet på varen i klartekst midt på skjermen, noe som gjør det lettere for brukeren hvis bildet skulle være utydelig.

Ved feil svar blir alternativet brukeren valgte farget med en dyp oransje farge. Hvis brukeren svarer feil på for mange forsøk, blir det gitt en melding til brukeren om dette, og korrekt svar vil bli avslørt. Ved valg av korrekt alternativ, vil brukeren bli presentert med en positiv melding om at korrekt svar ble avgitt. I motsatt fall blir det presentert en negativ melding. I denne meldingen vises også den korrekte koden. Antall forsøk blir definert utifra vanskelighetsgraden på oppgaven.

3.4.5 Spill - Velg bilde

Denne spill-modusen er en reversering av modusen som ble forklart ovenfor. Her blir brukeren presentert med PLU-koden, og oppgaven blir å velge det tilhørende bildet. Denne er ellers identisk med modusen beskrevet ovenfor.

3.4.6 Spill - Tast PLU

Denne modusen ble designet for de som allerede har lært seg mange av PLU-kodene. Presentasjonen av oppgaven er lik som ved modusen for "Velg PLU", men her må brukeren manuelt taste inn PLU-koden. Det er ingen alternativer, så brukeren er nødt til å kunne koden, eller deler av denne, for å løse oppgaven.

Avhengig av vanskelighetsgrad, får brukeren mulighet til å svare flere ganger. Når brukeren har tastet inn sitt svar, vil korrekte tall på korrekt plass bli farget grønne. Brukeren kan da velge å fjerne tallene som ikke er korrekte, for så å taste inn nye tall. Ved for mange forsøk vil brukeren bli presentert med svaret. Det er også en knapp til høyre for tallene som brukeren kan benytte for å fjerne alle inntastede tall.

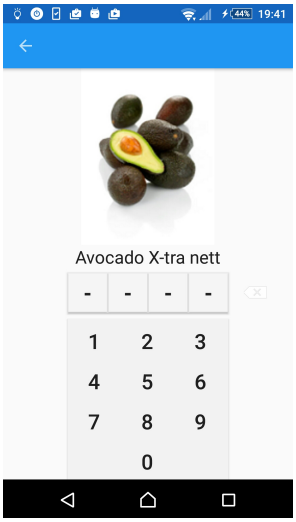


Figure 11: Spillmodus: Tast PLU

Tastaturet brukeren benytter seg av for å besvare oppgaven, er designet for å være likt tastaturet på et kassaapparat. Dette gjør det mer naturlig for brukeren å taste inn PLU-kodene.

4 Implementering

I dette kapitlet vil vi beskrive hvordan vi har gått fram for å programmere løsningene vi diskuterte i forrige kapittel. Vi vil ta for oss bitene fra designet, og forklare tilhørende kode.

Denne oppgaven krevde utvikling på både web- og mobil-platformer. Vi visste derfor at vi trengte kompetanse innenfor et bredt spekter av språk. Gjennom skolen, og av interesser utenfor skolen, hadde vi begge kompetanse innenfor flere av språkene vi visste vi måtte benytte oss av. Vi visste også at utvikling av løsningen for web og mobil måtte skje parallelt, fordi en god del funksjoner på mobil-applikasjonen er avhengig av funksjoner på web. For å teste en del funksjoner for web, var vi også avhengig av at en del av mobil-applikasjonen var ferdig. Vi avgjorde derfor at det mest praktiske for denne oppgave var å dele oppgaven mellom oss på en slik måte som gjorde at vi kunne fordype oss innenfor forskjellige emner. Det gjorde at en person hadde hovedfokus på å utvikle løsningen for web, mens den andre personen hadde hovedfokus på utvikling av mobil-applikasjonen. Siden hele gruppen har kompetanse innenfor begge feltene, kunne vi bistå hverandre ved problemer.

4.1 Web

Da vi startet utviklingen av løsningen for web, startet vi med fem forskjellige språk. Dette var HTML, CSS, JavaScript, PHP og SQL. Vi benyttet oss av HTML og CSS for å lage og manipulere layout av websiden. JavaScript ble brukt for behandling av data og objekter, og PHP sammen med SQL ble benyttet for all kommunikasjon mellom webgrensesnittet og databasen vår. Bruken av hvert språk vil bli forklart nærmere lengre ned i dette kapittelet. Vi har også benyttet oss av noen ferdige moduler og script, blandt annet Bootstrap, PNotify og DataTables. Vi kommer også til å forklare hvordan vi har benyttet oss av disse.

4.1.1 Database

Databasen vår ble først satt opp på vår konto som vi har tilgang til gjennom NTNU. Denne ble administrert via phpMyAdmin på NTNU sin portal. All kommunikasjon mellom databasen og både webgrensesnittet og mobil-applikasjonen skjedde via PHP og SQL. Vi benyttet oss av PDO-løsningen til PHP. Dette gjorde at vi kunne behandle databasen som et objekt, og alle kommandoer og funksjoner kunne da gjøres på akkurat samme måte, samme hvilken del av databasen som skulle behandles. Under viser vi noen eksempler på hvordan dette gjøres.

```
1 //Lager nytt PDO-objekt
2 $db = new PDO('mysql:host=mysql.stud.hig.no;dbname=dbName;charset=utf8','
    username','password',array(PDO::ATTR_EMULATE_PREPARES=> false,PDO::
    ATTR_ERRMODE => PDO::ERRMODE_EXCEPTION));
```

Listing 4.1: Initialisering av PDO

```
1 //Lager SQL-statement
```

```
$stmt = $db->prepare("INSERT INTO plucourse plucname, courselevel, iscompetition
    , fromdate, todate, notes) VALUES('$kursnavn','$vanskelighetsgrad', '
    $kurstype', '$startdato', '$sluttdato','$kommentar')");

//Sender statement til databasen
$stmt->execute();
```

Listing 4.2: Skriving av data med PDO

```
//Lager SQL-statement
$stmt = $db -> query('SELECT pluckey, plucname, courselevel, iscompetition,
    fromdate, todate, notes FROM plucourse ORDER BY todate ASC');

//bruker $stmt til henting av all data, lagres i $results
$results = $stmt -> fetchAll(PDO::FETCH_ASSOC);
```

Listing 4.3: Henting av data med PDO

På ukentlig møte med Gitek, Fredag 26.02, kom vi fram til at vi skulle flytte hele løsningen vår over til deres system. Siden dette systemet er i drift, var det Gitek som implementerte databasen, etter våre spesifikasjoner. De støtter ikke PHP, så all vår kode for databasekommunikasjon ble da fjernet. Gitek kommuniserer med sin database kun med JavaScript og en egen tilpasset AJAX-modul. All kommunikasjonen vår ble da omskrevet for å møte Gitek sitt krav. Under viser vi hvordan vi utfører de samme oppgavene som i eksemplene over med PHP og PDO.

```
tablename: 'plucourse',
  fields: ['plucname','courselevel','iscompetition','todate','fromdate','notes',
    'pluckey','status'],
  order: 'pluckey'
  },function(data){
    if (data.result) {
        //Behandle data som kommer i retur fra serveren
    }
  });
```

Listing 4.4: Skriving og henting av data med AJAX

Både skriving og henting av data blir gjort på samme måte når vi benytter oss av AJAX, men det blir på forhånd definert forskjellige metoder for å utføre spørringene. Disse metodene er etter ønske fra Gitek blitt utelatt fra rapporten.

4.1.2 Bootstrap

Bootstrap ble benyttet som en grunnstein under byggingen av webgrensesnittet. Dette er et sett med CSS og JavaScript filer som inneholder alt fra knapper og tekst, til paneler og modaler. Disse filene ble inkludert i HTML-dokumentene, og gav oss tilgang til alle funksjonene. Når vi skulle benytte oss av en modul fra Bootstrap i løsningen våre, måtte vi definere at klassen skulle komme derifra. Hvis vi ser på eksempelet fra forrige kapittel, hvor det ble vist en figur fra hovedsiden, var initialisering av det blå panelet for kurs slik:

```
<!--Panel for kurs-informasjon-->
<div class="col-lg-3 col-md-6">
    <div class="panel panel-primary">
        <div class="panel-heading"> Kurs</div>
    <div class="panel-body">
        <div  id="totalCourses"></div>
      <div  id="activeCourses"></div>
      <div  id="sevenDaysLeft"></div>
      <div  id="fourteenDaysLeft"></div>
    </div>
        <a href="plukurs">
```

```
            <div class="panel-footer">
                <span class="pull-left">Se kurs</span>
                <span class="pull-right"><i class="fa fa-arrow-circle-right"></i
    ></span>
                <div class="clearfix"></div>
            </div>
        </a>
    </div>
</div>
```

Listing 4.5: Initialisering av panel for kurs

Steder hvor klassen defineres til å inneholde "panel", er det klasser fra Bootstrap som blir benyttet. "panel-primary" definerer fargen på panelet, og alle fargene er tilpasset fargeblinde. Dette gjorde at Bootstrap ble spesielt nyttig sammen med JavaScript, da vi kunne vise meldinger og tilpasse innholdet på sidene våre uten å bekymre oss for fargene. Vi har benyttet Bootstrap til navigeringsbaren/headeren, paneler, modaler og knapper.

4.1.3 PNotify

PNotify er en liten modul for å vise meldinger til brukeren. Denne kan tilpasses etter nesten alle behov. Den initialiseres ved å definere en ny variabel til å være et PNotify-objekt, for så å sette egenskapene til dette objekter.

Dette ble brukt for å gi tilbakemeldinger til brukeren når det ble lagt til eller fjernet varer til kurs. PNotify ble også konfigurert til å beholde stilen til Bootstrap, for å gi et bedre helhetlig inntrykk. Under er det et enkelt eksempel på hvordan vi benyttet oss av dette.

```
var msg = new PNotify({
  title: 'Vare lagt til!',
  text: 'Lagt til "Banan kg" med PLU-kode "5730"!',
  type: 'success',
  history: {
    menu: true
  },
});
```

Listing 4.6: Lage meldinger med PNotify

Figure 12: PNotify suksess-melding

"Title" setter tittelen på beskjeden. "Text" bestemmer hva som skal være hovedteksten i meldingen, mens "type" definere fargen. Ved å sette "history: menu : true", får brukeren en meny for å vise gamle meldinger.

4.1.4 DataTables

DataTables er en plugin for JavaScript og jQuery. Den inneholder mange forskjellige tabeller, og måter å tilpasse disse på. Vi har benyttet oss av dette fordi det gav oss en veldig stor fleksibilitet når det kommer til bruk av tabeller, samtidig som tabellene blir

veldig brukervennlige. Dette gav oss enkelt muligheten til å ha søkbare tabeller, tabeller med mange sorteringsfunksjoner, og gav brukeren muligheten til tilpasse tabeller etter eget ønske.

DataTables kan initialiseres på mange måter. Fellesnevneren er at det må defineres og lages en tabell i HTML først. Deretter benytter vi oss av jQuery for å finne tabellen, og sier deretter at denne tabellene skal være av typen DataTables.

Det er også flere måter å legge til innhold i tabellene. DataTables lot oss legge til data manuelt, men vi kunne også sende et objekt eller array til funksjonen, og DataTables ville da fylle ut tabellen automatisk. Vi har flere ganger kombinert dette. I koden under laster vi ned et objekt med data om alle brukere fra databasen til Gitek via AJAX. Dette objektet blir sendt til DataTables, og vi lar tabellen blir fylt ut automatisk. Det eneste vi trenger gjøre manuelt, er å lage knappen som skal i siste kolonne for å gi administratoren mulighet til å slette brukere.

```
tablename: 'pluuser',
fields: ['email','aliasname','pluukey'],
order: 'pluukey'
},function(data){
  if (data.result) {
    $('#ansattliste').DataTable({
      data: data.value.records,
      retrieve: true,
      columns: [
      {data: 'aliasname'},
      {data: 'email'},
      {
        sortable: false,
        "render": function(data, type, full, meta){
          var clickEvent = full.pluukey;
          return '<button type="button" class="btn btn-primary" onclick="
    deleteUser('+clickEvent+')">Slett</button>'},
          "width":"13%"
      }
      ]
    });
  }
});
```

Listing 4.7: Lage tabell for brukere med DataTables

"ansattliste" er en id til ønsket tabell i HTML-koden. Vi setter så stien til objektet vi vil DataTables skal jobbe på. Vi definerer deretter kolonnene, og sier hva slags element fra objektet som skal settes inn hvor. Den siste delen av koden inneholder funksjoner som er innebygd i DataTables, og som lar oss manipulere og manuelt lage elementer i kolonnene. Denne benytter vi oss av for å lage en "slett"-knapp. Klassen "btn-primary" er en klasse fra Bootstrap, og gir oss en standard blå knapp.

4.1.5 Kode og algoritmer fra andre kilder

Det var noen ganger behov for andre funksjoner og biter av script enn det vi selv hadde kompetanse til å lage. Det var mye manipulering og sammeligning av data som ble gjort via JavaScript, og vi var derfor avhengig av å søke hjelp til dette.

Dagens dato på valgfritt format

For å fargekode rader i tabellen for kurs, og for å vise informasjon om snart avsluttede kurs, trengte vi tilgang på dagens dato. Vi ønsket også å ha muligheten til å kunne endre formatet på datoen slik vi ønsket. Vi fant en funksjon på internett som gjorde akkurat

dette, og vi tilpasset denne til eget bruk.

```
//Returnerer dagens dato, formatet er valgfritt, funnet hos http://stackoverflow
    .com/questions/1531093/how-to-get-current-date-in-javascript
function todaysDate(){
  var today = new Date();
  var dd = today.getDate();
  var mm = today.getMonth()+1;
  var yyyy = today.getFullYear();

  if(dd < 10){
    dd = '0'+dd;
  }

  if(mm < 10){
    mm = '0'+mm;
  }

  today = yyyy+'-'+mm+'-'+dd;
  return today;
}
```

Listing 4.8: Retur av dagens dato på ønsket format

Ved å endre "today = " kan vi lage det formatet vi selv ønsker på datoen som returneres.

Kollaps av hamburgermeny

I utgangspunktet blir ikke hamburgermenyen til Bootstrap lukket etter at det er foretatt et valg i menyen. Dette førte til lite brukervennlighet da brukeren måtte trykke utenfor menyen for å at den skulle bli bort. Vi fant derfor en kode for å få denne funksjonen til å virke som ønsket.

```
<!--Lukke hamburgermeny ved at det blir gjort et valg i menyen, hjelp hos https
    ://github.com/twbs/bootstrap/issues/9013#issuecomment-39698247-->
    <script type="text/javascript">
      $(document).on('click','.navbar-collapse.in',function(e) {
        if( $(e.target).is('a') ) {
          $(this).collapse('hide');
        }
      });
    </script>
```

Listing 4.9: Lukking av hamburgermeny ved valg i menyen

Denne koden ble plassert i alle HTML-dokumentene, for å sørge for at brukeren alltid har en konsistent opplevelse av menyen.

Konvertering av et bilde til en base64-string

Alle bilder av varer blir lagret som et blob-objekt i databasen, derfor må alle bilder bli konvertert til tekst før de kan lagres. Dette er ikke noe som skjer automatisk, så det måtte lages en funksjon for det. Etter mye problemer fikk vi til slutt hjelp fra Gitek til dette.

```
//Skrevet med hjelp fra Khai ved Gitek, konverterer et bilde til base64 string
function readURL(input, callback) {
  var reader = new FileReader();
  reader.onload = function (e) {
    $('#imgPreview')
          .attr('src', e.target.result)
          .width(200)
          .height(200);
    callback(reader.result)
  };
  reader.readAsDataURL(input.files[0]);
```

```
}
```

Listing 4.10: Konverting av bilder til bas64-string

Denne funksjonen tar imot et bilde fra HTML-formen. Dette bildet blir så lest inn, og funksjonen returnere en lang string som inneholder bildet i tekstformat ved hjelp av callback.

4.1.6 Egen kode og algoritmer

Det ble også laget en del egne funksjoner og algoritmer for å utføre operasjoner. Vi vil presentere noen av disse i dette del-kapitlet.

Endre størrelse på bildet

I tabellen med alle varene, er det en egen kolonne med bilde av alle varene. Dette bildet skulle være lite til vanlig, men det var ønskelig at det bildet ble større, og lettere å se, når holdt musepekeren over det. Dette ble løst med to funksjon som ble knyttet til hvert bildet.

```
//Endrer et bilde til 196 pixler
function makeImgBig(img){
  img.style.height = "196px";
  img.style.width = "196px";
}

//Endrer et bilde til 48 pixler
function makeImgSmall(img){
  img.style.height = "48px";
  img.style.width = "48px";
}
```

Listing 4.11: Dynamisk endring av bildestørrelser

Den første funksjonen forstørrer bildet når musepekeren kommer i kontakt med det, mens den andre funksjonen setter bildet tilbake til en mindre størrelse når musepekeren går ut igjen.

Farging av viktige rader

For at det skal være lettest mulig for brukeren å se om det er noen kurs som snart blir avsluttet, ble det programmert en løsning for å sette farge på radene i kursoversikten. Denne benytter seg av funksjonen som returnerer dagens dato, og finner ut om kurset blir avsluttet i løpet av en gitt tidsperiode fra i dag.

```
if((Date.parse(value.todate)  > Date.parse(todaysDate())) && (Date.parse(value.
    fromdate)  < Date.parse(todaysDate()))){
  if((Math.abs(Date.parse(value.todate) - Date.parse(todaysDate())) / 86400000)
    < 7){ // Kurs avsluttes i lopet av 7 dager
      $(newRow).css({"background-color":"#f2dede"});
    }else if((Math.abs(Date.parse(value.todate) - Date.parse(todaysDate())) /
    86400000) < 15) { // Kurs avsluttes i lopet av 15 dager
      $(newRow).css({"background-color":"#fcf8e3"});
    }else {
      $(newRow).css({"background-color":"#dff0d8"});
    }
}
```

Listing 4.12: Fargekoding av rader

På forhånd av denne funksjonen har vi akkurat laget raden som skal farges. Denne er lagret i variabelen "newRow". Variabelen "value.todate" er datoen for kursets slutt, og

"value.fromdate" datoen for kursets start. Disse kommer fra databasen, og blir sammenlignet med dagens dato. Hvis dagens dato er mellom start- og slutt-datoen, vet vi at kurset er påbegynt. Da skal raden farges grønn. Vi regner ut differansen mellom datoene, og deler denne på 86400000. Det er 86400000 millisekunder i et døgn, derfor må denne operasjonen utføres før vi kan regne ut hvor mange dager det er igjen av kurset. Avhengig av resultatet blir denne raden farget gul eller rød hvis det er mindre enn henholdsvis to eller en uke igjen av kurset. Alle fargene er tilpasset fargeblinde.

4.2 Android-applikasjon

Ved utvikling av applikasjonen for Android, var det tre språk som var aktuelle å bruke: Java, XML og SQL. Java er grunnpilaren i Android-utvikling, og ble brukt for programmering av all funksjonalitet. XML ble brukt for utviklingen av det grafiske brukergrensesnittet, samt konstanter (e.g., farger, størrelser og strenger).

4.2.1 Database

Databasen for Android-applikasjonen ble i første omgang satt opp med SQLite, som er en enkel database som ikke krever konfigurasjon eller en server. Derfor er den typisk godt egnet for bruk på mobile enheter. Underveis i utviklingen derimot, fant vi et nytt database-system, kalt Realm. Etter diskusjon innad i gruppen, ble vi enige om å bytte til Realm, da vi mente dette ville drastisk kunne korte ned på utviklingstiden på iOS-applikasjonen. For å holde applikasjonene enkle å videreutvikle, ble det også bestemt at vi skulle bytte også på Android.

Realm skiller seg fra SQLite ved at du ikke håndterer databasen via SQL-setninger, men du håndterer innholdet som objekter. Dette er noe som drastisk forenkler databasen, og er noe som sparte oss mye tid når vi skulle utvikle iOS-applikasjonen. Det kostet oss derimot noe tid på Android-applikasjonen, da store deler av denne måtte skrives om. Under viser vi noen eksempler på forskjeller hos SQLite og Realm.

```
db.execSQL(CREATE_COURSES);

String CREATE_COURSES =
    "CREATE TABLE IF NOT EXISTS " + TABLE_COURSES + " (" +
    COURSES_ID + " INTEGER PRIMARY KEY, " +
    COURSES_NAME + " TEXT, " +
    COURSES_DESCRIPTION + " TEXT, " +
    COURSES_COURSELEVEL + " INTEGER, " +
    COURSES_COMPETITION + " BOOLEAN, " +
    COURSES_STARTDATE + " DATE);";
```

Listing 4.13: Oppretting av tabell i SQLite

```
public class PLUCourse extends RealmObject {
    @PrimaryKey private int pluCKey;
    @Required private String pluCName;
    private RealmList<PLUCourseItem> pluCourseItems = new RealmList<>();
    private PLUType pluType;
    private String notes;
    private int courseLevel;
    private boolean isCompetition = false;
    private Date fromDate;
    /* ... */
}
```

Listing 4.14: Oppretting av tabell i Realm

```
final Cursor c = db.query(true, Courses.TABLE_COURSES,
    new String[]{Courses.COURSES_ID, Courses.COURSES_NAME},
    Courses.COURSES_COMPETITION + "=" + competition,
    null, null, null, null, null, null);
```

Listing 4.15: Henting av konkurranser med SQLite

```
RealmResults<PLUCourse> pluCourses = realm.where(PLUCourse.class)
    .equalTo("isCompetition", competition).findAll();
```

Listing 4.16: Henting av konkurranser med Realm

```
SQLiteDatabase db = MainActivity.sHelper.getWritableDatabase();

ContentValues cV = new ContentValues();
cV.put(Courses.COURSES_ID, id);
cV.put(Courses.COURSES_NAME, name);
cV.put(Courses.COURSES_COURSELEVEL, courseLevel);
cV.put(Courses.COURSES_COMPETITION, competition);
cV.put(Courses.COURSES_DESCRIPTION, description);
/* ... */

if (db.insert(Courses.TABLE_COURSES, null, cV) > 0) {
    db.close();
}
```

Listing 4.17: Lagre kurs med SQLite

```
Realm realm = Realm.getDefaultInstance();

PLUCourse course = new PLUCourse();
course.setPluCKey(id);
course.setPluCName(name);
course.setNotes(description);
course.setCourseLevel(courseLevel);
course.setIsCompetition(competition);
/* ... */

realm.beginTransaction();
realm.copyToRealmOrUpdate(course);
realm.commitTransaction();
realm.close();
```

Listing 4.18: Lagre kurs med Realm

4.2.2 RecyclerView

For visning av lister i Android, har vi valgt å benytte oss av RecyclerView. Dette er en oppdatert ListView, som både er mer effektiv og mer tilpassingsvennlig enn sistnevnte. For å bruke RecyclerView er det derimot nødvendig å bruke en eller flere adapter(e), for å håndtere innholdet som skal vises. For vår del var det nødvendig med fire adaptere; for vareliste, kursliste, treningskategorier og poengliste. Disse adapterne ble basert på SimpleCursorRecyclerAdapter[2]

```
private final LayoutInflater mInflater;
    private RealmResults<PLUType> mTypes;
    public static Context context;

    public TrainingCategoryRealmAdapter(Context context, RealmResults<PLUType>
    pluTypes) {
        this.context = context;
        mInflater = LayoutInflater.from(context);
        mTypes = pluTypes;
    }

    @Override
    public CategoryViewHolder onCreateViewHolder(ViewGroup parent, int viewType)
     {
        final View itemView = mInflater.inflate(R.layout.category_list_item,
    parent, false);
        return new CategoryViewHolder(itemView);
    }

    @Override
    public int getItemCount() {
        return mTypes.size();
    }

    @Override
    public void onBindViewHolder(CategoryViewHolder holder, int position) {
        final PLUType pluType = mTypes.get(position);
        holder.bind(pluType);
    }

    class CategoryViewHolder extends RecyclerView.ViewHolder {

        private int typeId;
        private OnCategorySelectListener mListener;
        private TextView categoryTextView;

        public CategoryViewHolder(final View itemView) {
            super(itemView);
            categoryTextView = (TextView) itemView.findViewById(R.id.
    category_list_textView);

            mListener = (OnCategorySelectListener) TrainingCategoryRealmAdapter.
    context;

            itemView.setOnClickListener(new View.OnClickListener() {
                @Override
                public void onClick(View v) {
                    mListener.onCategorySelected(typeId);
                }
            });
        }

        public void bind(PLUType pluType) {
            this.typeId = pluType.getPluTKey();
            categoryTextView.setText(pluType.getPluTName());
        }
    }

    public interface OnCategorySelectListener {
        void onCategorySelected(int typeId);
    }
}
```

Listing 4.19: Implementasjon av TrainingCategoryRealmAdapter

4.2.3 Brukerhåndtering

Som tidligere nevnt, var den opprinnelige planen å bruke Google Sign-in for å håndtere innlogging av brukere. Vi fikk derimot noen problemer med implementeringen av dette, og resultatet ble at innlogging kun fungerte sporadisk. Etter noe feilsøking på dette problemet, var vi nødt til å ta et møte med oppdragsgiver for å diskutere prioriteringer. Etter diskutering, ble vi enige om å prioritere iOS-applikasjon framfor Google Sign-in. Dette førte til at vi valgte å utvikle en enkel brukerhåndtering selv.

Brukerhåndteringen vår består av tre deler. Først registerer brukeren seg, med brukernavn, epost, passord, bekreftelse av passord og hvilken region i Coop brukeren hører til, hvis noen. Dette sendes til Giteks database, og lagres der dersom ingen andre brukere med den eposten eller brukernavnet eksisterer.

Når brukeren er opprettet, taster brukeren inn brukernavn og passord. Passordet her tastes inn i et tekstfelt som tar imot en tekst-type "textPassword". Dette sørger for at passordet sensureres, slik at ingen kan se hva som tastes inn. Når brukeren trykker "Logg inn", sendes informasjonen til databasen til Gitek, som forsøker å logge inn brukeren. Hvis den klarer det, sendes brukerid tilbake til applikasjonen, som da bekrefter innloggingen. Både registrering og innlogging ble gjort i separate ASyncTask, implementert med hjelp av dokumentasjonen fra Android Developers[3].

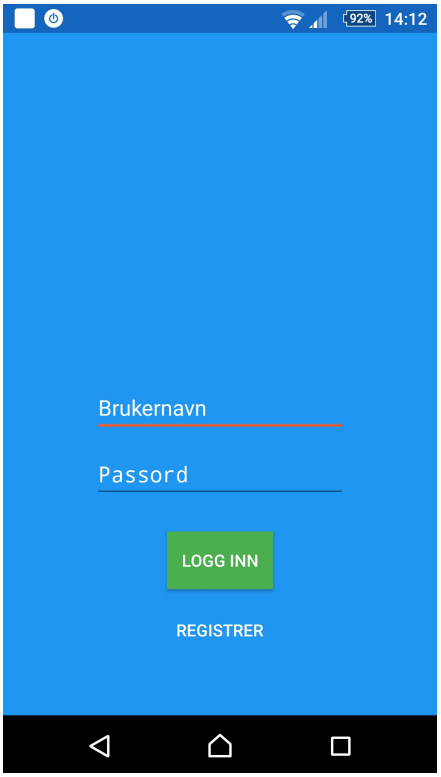


Figure 13: Enkel brukerinnlogging

4.2.4 Spill

Implementeringen av spillene ble utført ved å opprette en Activity[4] (GameActivity) for å håndtere alt som er felles for spillene(e.g., antall sjanser, antall runder, etc.), og en

Fragment[5] for hver spillmodus som håndterer alt som er spesifikt for den spillmodusen(e.g., brukergrensesnittet). Ved å dele det opp på denne måten, gjorde vi det veldig enkelt for en utvikler å legge til ytteligere spill-moduser, om det skulle være ønskelig.

4.2.5 Poenglister

Etter diskusjon med oppdragsgiver ble vi enige om å ha to forskjellige poenglister. En for kurs, hvor du kun kan se dine egne poengsummer, og en for konkurranser, hvor vi viser den høyeste poengsummen til hver bruker for den konkurransen. Disse listene blir vist fram ved hjelp av RecyclerView og en tilhørende adapter.

Poenglisten for kurs var den enkleste å implementere. Vi trengte kun en enkel spørring til databasen, hvor vi hentet ut alle poengsummer for det kurset og den brukeren, og deretter sortere den på poengsummene.

```
RealmResults<PLUScore> courseScores = realm.where(PLUScore.class)
    .equalTo("pluCourse.pluCKey", pluCourse.getPluCKey())
    .equalTo("pluUser.pluUKey", MainActivity.pluukey)
    .findAllSorted("score", Sort.DESCENDING);
```

Listing 4.20: Hente poengsummer for kurs

For å fylle poenglisten for konkurranser trengte vi en mer innviklet løsning. Her måtte vi først finne alle brukere og lagre de i en RealmResults. Deretter gikk vi gjennom alle brukerne, og fant den høyeste poengsummen i den konkurransen, hvis de hadde deltatt, og lagret den i en ArrayList. Til slutt sorterte vi den listen på poengsummene ved å bruke Collections[6].

```
RealmResults<PLUUser> users = realm.where(PLUUser.class).distinct("pluUKey");
List<PLUScore> competitionScores = new ArrayList<>();
for (int i = 0; i < users.size(); i++) {
    RealmResults<PLUScore> scores = realm.where(PLUScore.class)
            .equalTo("pluUser.pluUKey", users.get(i).getPluUKey())
            .equalTo("pluCourse.pluCKey", pluCourse.getPluCKey())
            .findAllSorted("score", Sort.DESCENDING);
    if (scores.size() > 0) competitionScores.add(scores.first());
}
Collections.sort(competitionScores, new Comparator<PLUScore>() {
    @Override
    public int compare(PLUScore lhs, PLUScore rhs) {
        return rhs.getScore() - lhs.getScore();
    }
});
```

Listing 4.21: Hente poengsummer for konkurranser

4.3 iOS-applikasjon

Ved utvikling av iOS-applikasjon, var det to språk som var aktuelle å bruke: Swift og Objective-C. Siden Swift er det nye språket fra Apple, valgte vi å gå for dette.

iOS-applikasjon ble planlagt som utvidet funksjonalitet, men underveis i utviklingen ble det ønsket fra oppdragsgiver at vi skulle prøve å utvikle den. Vi ble enige om å prøve dette, men valgte å ha Android-applikasjon som første prioritet. Dette førte til at iOS-applikasjonen ikke er fullført i samme grad som Android-applikasjonen.

Siden ingen av oss hadde noe særlig erfaring med iOS, valgte vi i første omgang å prøve forskjellige applikasjonen på iOS. De mest interessante var de som var på både Android og iOS, slik at vi kunne se forskjellene i brukergrensesnittet selv. Dette førte til

at vi i stedet for å implementere en hovedmeny slik vi har på Android, valgte vi heller å implementere et grensesnitt med faner.

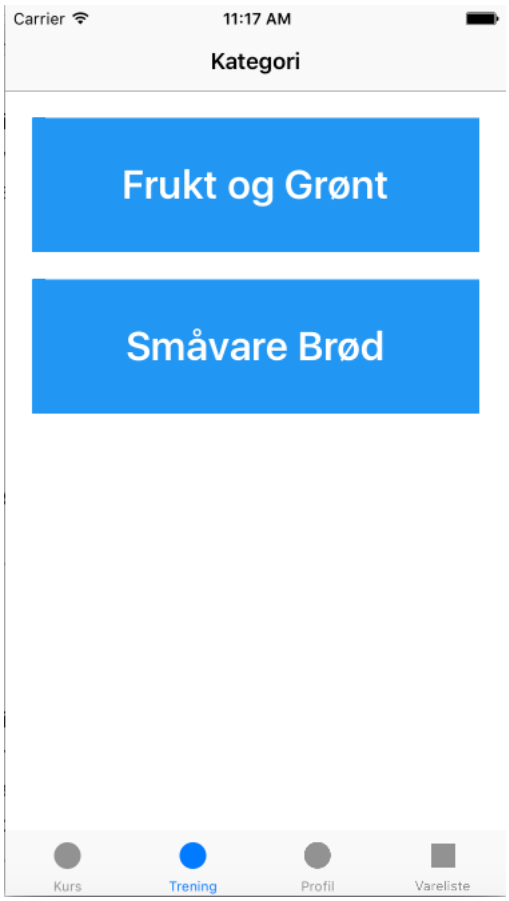


Figure 14: Varekategorier på iOS

4.3.1 Database

Databasen for iOS-applikasjonen ble utviklet med samme system som Android; Realm. Med unntak av syntax, er den implementasjonen nesten identisk med Android.

```
class PLUScore: Object {

    //MARK: Properties
    dynamic var pluSKey: Int = 0;
    dynamic var pluUser: PLUUser?;
    dynamic var pluCourse: PLUCourse?;
    dynamic var score: Int = 0;
    dynamic var stamp: NSDate?;
    var pluScoreItems: [PLUScoreCourseItem] {
        return linkingObjects(PLUScoreCourseItem.self, forProperty: "pluScore");
    }
    /* ... */
}
```

Listing 4.22: Opprette tabell Realm - Swift

```
let realm = try! Realm();

let newCourseItem = PLUCourseItem();
newCourseItem.orderNo = i;
newCourseItem.pluCourse = newCourse;
newCourseItem.pluItem = dummyPluItems[i];
newCourseItems.append(newCourseItem);

try! realm.write {
    realm.add(newCourseItem);
}
```

Listing 4.23: Lagre en rad i databasen - Swift

4.3.2 Nedlasting av database

Implementering av nedlastingsfunksjonaliteten var noe vi hadde store problemer med på iOS, og dette ble derfor ikke ferdig. Med det sagt, så har vi i siste versjonen prøvd Alamofire[7] for nedlasting. Dette er et bibliotek som sparer utvikleren for mye tid, og er noe vi kommer til å bruke videre.

```
let url = NSURL(string: urlString)!;
let request = NSURLRequest(URL: url);

let task = NSURLSession.sharedSession().dataTaskWithRequest(request) {
    (data, response, error) in
    if error == nil {
        if let data = data {
            let realm = try! Realm();
            let parsedResult: NSDictionary!

            do {
                parsedResult = try NSJSONSerialization.JSONObjectWithData(data,
    options: .AllowFragments) as! NSDictionary;
                        print("types: \(parsedResult)");
            } catch {
                print("Kunne ikke laste ned: '\(data)'");
                return;
            }
        }
    }
}
task.resume();
```

Listing 4.24: Swift - Laste ned uten Alamofire

```
Alamofire.request(.GET, "\(ajaxUrl)Ajaxmodule=db&ajax=GetRecords", parameters: [
    "tablename": "plutype"])
    .responseJSON { response in
        print(response.request)  // original URL request
        print(response.response) // URL response
        print(response.data)     // server data
        print(response.result)   // result of response serialization

        if let JSON = response.result.value {
            print("JSON: \(JSON)")
        }
}
```

Listing 4.25: Swift - Laste ned med Alamofire

5 Testing og tilbakemeldinger

5.1 Generelt

Det ble utført enhetstesting, komponenttesting og systemtesting av web-løsningen. Enhetstestingen innebar testing av enkelte funksjoner under utviklingen, for å sikre at disse gav korrekte resultater og returnerte korrekt data. Komponenttestingen gjaldt større operasjoner, og inneholdt gjerne flere funksjoner på en gang. Flere av disse funksjonene hadde også vært gjennom enhetstesting på forhånd. Systemtesten ble gjort av hele systemet som en helhet. Dette inkluderte da web-løsningen, mobil-applikasjonen og databasen.

Testing av mobil-applikasjonene ble også delt opp i tre deler: enhetstesting, komponenttesting og systemtesting. All ny funksjonalitet gikk gjennom denne testrutinen hver uke, slik at vi best mulig kunne sørge for at applikasjonene fungerte korrekt. Dette var tidkrevende, og et bedre planlagt system for testing ville sørget for kjappere og mer effektiv testing.

På grunn av tidsklemmer hadde vi ingen dedikerte brukertester. Siden dette systemet i utgangspunktet ble utviklet for Coop-ansatte, var vi avhengige av at eventuell brukertesting ville foregått med de. Selv om det er mange forskjellige butikker som bruker PLU-koder, implementerte vi PLU-koder som er spesifikke for Coop, og vi ville derfor ha kunnet forvirret andre butikkmedarbeidere med feil koder. Brukertesting ville dog ha kunnet hjulpet oss med å oppdage flere feil kjappere, samt at vi kunne få en tilbakemelding på hvordan brukeropplevelsen generelt er. Dette er noe vi forventer og håper oppdragsgiver vil utføre en gang etter siste innlevering.

5.2 Enhetstesting

Med enhetstesting mener vi testing, kontroll, og korrigering av resultatet fra en enkelt funksjon eller metode.

5.2.1 Web

Enhetstesting av webløsningen var noe som ble et kontinuerlig arbeid. Funksjonen "console.log(variabel/object)" var noe som ble flittig brukt, da dette gir en utskrift i utviklervinduet til Google Chrome. Denne utskriften inneholder akkurat hva variabelen eller objektet inneholder i det punktet i koden som utskriften blir kalt. Dette forstyrrer ikke eksisterende kode eller funksjoner. Det vil si at koden ble kjørt som normalt, men vi kan hele tiden se hva som skjer. Dette gjorde også at vi kunne se innholdet etterhvert som koden ble kjørt, og ikke bare sluttresultatet.

En annen fordel med å benytte seg av "console.log()", er muligheten til å traversere både array og objekter. Dette gav oss muligheten til å se hele innholdet i objektene. Vi kunne på denne måten se når, og hvordan, et objekt eller array ble laget og endret. Dette var spesielt nyttig ved henting av data fra databasen, da resultatet alltid kommer som et objekt som inneholder flere andre objekter. Vi kunne da se akkurat hvor den informasjonen vi var ute etter befant seg.

Vi benyttet oss også alltid av "console.log()" ved oppretting av funksjoner som skulle sende informasjon til databasen. På denne måten kunne vi alltid kontrollere at alt var korrekt, før vi risikerte å sende dårlig data til serveren. Vi kunne kontrollere at objekter som inneholdt informasjon om varer og kurs var korrekte, og vi kunne kontrollere av konvertering av bilder gav oss korrekte tekststringer.

Bortsett fra ovenfornevnte funksjoner, ble det også benyttet visuell inspeksjon. Web-grensesnittet består av mye grafikk, og dette ble fortløpende testet og tilpasset etterhvert som funksjoner ble laget.

5.2.2 Mobil-applikasjon

Før utviklingen startet, hadde vi diskusjoner om hvordan vi skulle teste applikasjonen. Tidlig i diskusjonen kom det opp automatiserte tester, eksempelvis via JUnit tester på Android. Selv om dette var noe vi kunne tenke oss, men dette ble ikke utført siden vi var usikre på nøyaktig hvordan vi burde satt opp disse testene. Av den grunnen antok vi at dette var noe som kom til å ta for lang tid å implementere på en god nok måte. I ettertid ser vi at dette var noe vi burde tatt oss tid til, og er definitivt noe vi vil bruke i senere prosjekter.

Uten automatisert testing, ble vi nødt til å sette opp vårt eget system. Siden vi hadde valgt bort automatisert testing på grunn av tiden vi antok det ville ta, var det essensielt å velge en form for testing som ville ta kortest mulig tid å implementere. For vår del falt valget på å bruke System.out.println() og System.err.println() på Android, og print() på iOS for å skrive ut til skjerm. Når man bruker disse funksjonene er man avhengig av å ha smarttelefonen tilkoblet PC/MAC med Android Studio/Xcode for å se resultatet. Dette ble derfor white-box testing, noe som er veldig vanlig i enhetstesting.

Selv om vi brukte println() og print() til å skrive ut mange former for resultater, hvor mange av disse tilfellene enkelt kunne blitt testet via automatiserte tester, var det noen tilfeller hvor disse var spesielt nyttige. En av disse tilfellene var ved opplasting og nedlasting til database. Her var det veldig praktisk å skrive ut både JSON-objektet vi sendte til server og det vi mottok som svar, for å se om dette fungerte korrekt. Resultatene fra dette ble så kontrollert mot en intern testfunksjon hos oppdragsgiver for å se om vi fikk det samme resultatet.

Et annet tilfelle hvor print() viste seg å være praktisk var ved utregning av poengsummer i konkurranser og kurs. I disse utregningene blir tiden brukeren bruker på oppgavene regnet om til en poengsum som blir lagt til resultater hvor brukeren har svart korrekt. På grunn av at resultater fra hver enkelt oppgave lagres lokalt i database, var det her praktisk å skrive ut alle resultater, samt tiden som var brukt for hver oppgave for å se om dette var korrekt. Testing av denne funksjonen viste at tiden ble telt ned dobbelt, noe som førte til at brukeren kunne få en negativ poengsum.

```
for (int i = 0; i < score.getPluScoreItems().size(); i++) {
    JSONObject itemObject = new JSONObject();
    /* ... */
    PLUScoreCourseItem item = score.getPluScoreItems().get(i);

    // Setting up JSON-object
    itemObject.put("tablename", "pluscorecourseitem");
    /* ... */

    conn.connect();

    is = conn.getInputStream();
    String string = convertInputStreamToString(is);

    // Print out output, then input:
    System.out.println(itemObject.toString());
    System.out.println(string);
}
```

Listing 5.1: Eksempel på testing av JSON

5.3 Komponenttesting

Med komponenttesting mener vi testing, kontroll, og korrigering av resultatet fra en større del av løsningen, hvor flere funksjoner eller metoder jobber sammen for å gi et ønsket resultat.

5.3.1 Web

Da vi ønsket å beholde webgrensesnittet så lett og oversiktlig som mulig for brukeren, ble det oppdelt i:

- Oversikt over varer, kurs og ansatte
- Oppretting av varer og kurs
- Endring av varer og kurs
- Sletting av varer, kurs og ansatte
- Detaljert oversikt over kurs

Dette gjorde at hver enkelt seksjon av løsningen ble dedikert til en enkelt operasjon. Dette betød at det ble naturlig å dele komponenttestingen opp i tilsvarende deler.

Alle funksjoner for oversikt, oppretting, endring og sletting ble opprettet for varer først. Disse ble da testet grundig, for å se at helheten virket slik den var planlagt. Dette innebar at all kommunikasjon til og fra databasen skulle virke, tabeller skulle vise korrekt informasjon, kontroller av former og skjema skulle feile på ukorrekt input, og kun slipper gjennom det som var korrekt data, datoformat skulle vises korrekt, og brukeren skulle ikke ha mulighet til å ødelegge eller påvirke noe som ikke var designe for å kunne bli påvirket. Alle responsive funksjoner ble også testet.

Da alt dette var testet og kontrollert, ble det videreført til andre deler av løsningen. Det vil si at når alle funksjoner for varene fungerte som planlagt, ble de videreført til kurs. Da var vi sikre på at det grunnleggende var i orden, og vi kunne tilpasse dette til det nye webområdet. Hele operasjonen ble så repetert, før det ble videreført til resterende seksjoner.

Detaljert oversikt over kurs er en ganske unik funksjon, da det er kun kurs som benytter seg av denne. Denne benytter seg av veldig mye kode, da det er mye informasjon som

skal hentes fra mange tabeller i databasen. Dette gjorde at vi delte opp websiden i flere logiske områder, og fullførte et og et område. Dette gjorde at testing av alle områdene kunne utføres uavhengig av hverandre.

5.3.2 Mobil-applikasjon

Komponentene i mobil-applikasjonen kan grovt deles opp i:

- Spilltype: Velg PLU
- Spilltype: Skriv PLU
- Spilltype: Velg bilde
- Menysystem
- Søk av varer
- Internettkommunikasjon
- Kurs og konkurranser
- Trening

Hver av disse komponentene gikk gjennom flere typer testing før de ble regnet som ferdige, i hvert fall fra et funksjonelt ståsted. I første omgang ble disse testet av utvikler, som sjekket om dette fungerte og så ut som planlagt. Dette skjedde først ved inspeksjon av koden, og så ved faktisk testing av applikasjonen, enten via emulator eller på mobiltelefon. Siden utvikleren har tilgang på koden, var dette white-box testing.

Etter at utvikleren var ferdig med sin testing, og eventuell feil-oppretting, ble andre gruppemedlemmer satt til å teste komponenten. De hadde ikke tilgang på koden, og det var derfor en black-box testing. Ved å teste på denne måten, kunne vi på en bedre måte oppdage uventede feil. Etter at utvikler hadde rettet opp i tidligere feil, kunne det skje at det dukket opp en feil et annet sted i komponenten, som utvikler i forveien hadde testet og godkjent. Ved å la andre teste funksjonaliteten ble disse feilene oppdaget. Et eksempel på dette er når utvikler hadde oppdaget en feil i spill via kurs, som innebar et glemt kall på en funksjon. Når kallet på funksjonen ble implementert, ble det glemt en sjekk på om det var kurs eller trening. I forveien hadde utvikleren testet spill via trening, og godkjent denne. Mangel på denne sjekken ble oppdaget når andre gruppemedlemmer fikk testet funksjonen. Det hendte også at vi brukte familiemedlemmer og venner til å prøve applikasjonen, noe som også dekket dette behovet.

Siste ledd i testingen var med oppdragsgiver. På slutten av hver uke hadde vi møte med de, og da viste vi frem nye funksjoner og komponenter. Her fikk de muligheten til å prøve, samt komme med kommentarer på ønskede endringer. Disse var som oftest en endring i brukergrensesnittet.

5.4 Systemtesting

Vi har valgt å definere systemtesting som en test hvor behandling av data gjennom web-grensesnittet var beregnet å ha en påvirkning på mobil-applikasjonen.

Systemtesting var en stor del av testingen vår. Løsningen vår består av en ekstern database, en web-applikasjon, og to mobil-applikasjoner på hver sin platform. For at løsningen skal fungere i sin helhet, var vi avhengig av å teste hele systemet. Da kunne vi bekrefte alle delene fungerte, også når de ble satt i et og samme system.

5.4.1 Varer

Mobil-applikasjonen er helt avhengig av varer, da treningsmodusen og alle kursene består av varer. Webløsningen har muligheten til å administrere disse varene, inkludert legge til bilder av varer. Det var derfor naturlig å foreta systemtester når webløsningen fikk funksjoner for å kunne utføre noen av disse oppgavene. Denne testingen foregikk i fire steg:

1. Vi brukte webløsningen for å legge til en vare.
2. Sjekk av varelisten på webløsningen for å se om den registrerte den nye varen.
3. Oppstart av mobil-applikasjonen, slik at den startet nedlasting av nye varer.
4. Sjekk varelisten på mobil-applikasjonen for å se om den nye varen var der.

5.4.2 Kurs

Da det ble mulighet for å opprette kurs med webløsningen, og denne funksjonen var testet, ble det utført systemtester for å kontrollere at dette virket sammen med mobil-applikasjonen. Her var det mange variabler som ble testet.

Systemtesting av kurs ble gjort flere ganger. Grunnen til dette er at kurs en en veldig stor del av løsningen, og har mange funksjoner. Det ble testet at brukeren av mobil-applikasjonen fikk tilgang til kursene som ble oppprettet av webløsningen. En bruker skulle derimot ikke kunne starte et kurs som ikke hadde passert sin startdato. Ved endring av kurs måtte det testes at mobil-applikasjonen ble oppdatert. Det samme gjaldt ved sletting av et kurs. Det er også definerte forskjeller på kurs og konkurranser, noe som gjorde at mobil-applikasjonen skulle behandle kurset anderledes hvis det var definert som en konkurranse av webløsningen.

5.4.3 Brukere

Brukere er noe som skal opprettes av mobil-applikasjonen, og vises i webløsningen. Dette valgte vi å teste ved at vi opprettet en bruker, og sjekket om denne ble lagt inn i databasen, og deretter om vi kunne få tak i denne brukeren med web-applikasjonen. Under denne testingen fant vi ut at opprettingen av brukere ikke fungerte, selv om vi kunne bekrefte at vi klarte å koble oss opp mot databasen med mobil-applikasjonen. Vi valgte derfor å skrive resultatet vi fikk fra databasen til konsoll. Det viste seg at vi ikke hadde tilgang til å opprette brukere. Vi tok opp dette med oppdragsgiver under neste møte, hvor dette viste seg å være en sikkerhetsfunksjon hos de. Dette fikk vi en løsning på samme dag som møte, og vi kunne da bekrefte at oppretting av brukere fungerte.

5.4.4 Poengsummer

Etterhvert som brukere deltar på kurs, skal en poengsum beregnes. Denne poengsummen skal vises både i mobil-applikasjonen, og i webløsningen. Det ble derfor utført flere tester hvor forskjellige brukere deltok på kurs via mobil-applikasjonen, og resultatet av denne deltakelsen ble vist i webløsningen. Det ble da kontrollert at antall deltakelser, og poengsummer, stemte med det som ble utført.

5.4.5 Eksterne hjelpemidler

Vi hadde tilgang til et godt hjelpemiddel da det kom til systemtesting og verifisering av data. Dette var Gitek sitt interne system for behandling av AJAX. Dette var en portal vi kunne benytte oss av for å lese og skrive data direkte til og fra databasen. Denne ble

flittig brukt for å verifisere at både mobil-applikasjonen og webløsningen leste og sendte korrekt data.

6 Konklusjon

Dette kapittelet vil inneholde en oversikt over det ferdige prosjektet, og litt om hva som var bra og dårlig. Det vil også inneholde en evaluering av gruppens arbeid.

6.1 Valg

Vi har foretatt mange valg under utviklingen. Det har ofte vært flere alternativer til hvordan en oppgave kunne løses, eller til hvordan grensesnittet skulle se ut. Mange av valgene har vi tatt på egenhånd, mens andre valg er blitt tatt i samråd med oppdragsgiveren.

Vi valgte å bruke å bruke tabeller i stedet for lister for å vise informasjon på webløsningen. Dette gjorde at vi kunne vise oversiktene på en mer interaktiv måte, og vi hadde flere muligheter når det kom til tilpasninger. Vi benyttet også en plugin for dette, noe som gjorde at tabellen allerede kom med en del funksjoner som vi ønsket.

Valget med å benytte oss av Bootstrap var for å gjøre det lettere for oss å gi brukeren en mer konsistent opplevelse av løsningen, samtidig som Bootstrap kom med mange verktøy og muligheter vi fant nyttige. Vi kunne laget alt fra bunnen av selv, men det ville gitt oss mye mer jobb. Det ville ikke bare blitt vanskeligere å få de funksjonene vi ønsket, det ville også blitt vanskeligere å få en lik og moderne stil på hele det grafiske grensesnittet.

I utgangspunktet hadde vi ikke forhåndsvisning av bilder når en bruker skulle legge til et bilde for en vare på web. Vi fant ut at dette burde implementeres, for å hjelpe brukeren med å verifisere at korrekt bilde var blitt valgt. Vi valgte derfor å gjøre dette for å bedre brukervennligheten.

I stedet for å ha en lang liste med alle muligheter og informasjon synlig på en side i webløsningen, valgte vi å gjemme det brukeren ikke trengte der og da. Vi benyttet oss i stedet for knapper for å gjemme og vise relevant informasjon. Da slipper brukeren å gå igjennom en lang internettside for å finne informasjonen som er relevant, og får i stedet presentert akkurat det som trengs.

Vi startet med å kun ha muligheten til å legge til brukere på webløsningen. Dette fungerte fint under testing, da vi hadde behov for en rask måte å lage brukere på akkurat da. Dette ble endret mot slutten, da det ble klart at dette ville bli en altfor stor jobb for administratorer å holde orden på når det begynner å bli mange brukere. Det skal heller ikke være veldig mange administratorer i dette systemet, og det er derfor mye mer effektivt å la brukerne lage brukerkontoer fra mobil-applikasjonen.

Vi gjorde også et ganske stort valg da vi sa ja til å flytte hele systemet vårt over til Gitek sitt system. Vi implementerte en egen database, og vår egen metode for å kommunisere med den via PHP. Det kom et punkt i oppgaven, hvor vi var avhengig av å ha tilgang til en del data, blant annet bilder av varer. Dette var noe som allerede befant seg på systemet til Gitek, og planen var at løsningen vår skulle flyttes til dette systemet uansett en gang i fremtiden. Det var ingen krav fra Gitek om at vi skulle stå for denne flyttingen. Vi valgte allikevel å foreta denne flyttingen, da vi bedre kunne tilpasse applikasjonene vår til deres

system, selv om vi var klar over at dette kom til å medføre en god del ekstra arbeid.

Det største valget vi gjorde på mobil-applikasjonen var å bytte database-system fra SQLite til Realm. Dette førte til veldig store omskrivninger av applikasjonen, da all grunnleggende funksjonalitet involverer databasen. Selv om vi er fornøyde med valget, var dette likevel noe som kostet mye tid. Med det sagt, så vet vi ikke hvor lang tid, hvis noe, vi kunne spart på å beholde SQLite, da dette var vesentlig mer tidkrevende å implementere.

Ved listing av kurs var det viktig for oss at brukere beholdt tilgangen til utgåtte kurs, slik at de kunne se resultatene. Vi hadde store diskusjoner på hvordan dette skulle løses, og endte opp med å ha tre lister for kurs: Kurs, konkurranser, og utgåtte. På denne måten kunne vi vise brukeren alle kurs, uten at det ble uoversiktlig. Dette viste seg å være raskt å implementere, så vi er godt fornøyde med dette valget.

Vi valgte å designe flere forskjellige spillmoduser, for å gjøre mobil-applikasjonen mer varierende, og derfor mer spennende for brukeren. Av denne grunnen valgte vi også å implementere flere vanskelighetsgrader, slik at brukeren kan utfordre seg selv slik de vil. Eventuelt så kan administrator velge å opprette vanskelige konkurranser, slik at man virkelig må kunne PLU-kodene for å kunne hevde seg i konkurransen. Vi mener dette var et godt valg, og oppdragsgiver har også vært positiv i denne avgjørelsen.

6.2 Kritikk

Ingen av gruppemedlemmene har erfaring når det kommer til så store prosjekter. Vi har hatt fag på skolen for å forberede oss på det, og det har vært nyttig. Allikevel skjer det ting som gir litt ekstra jobb.

Bruk av navn på variabler og funksjoner for web-applikasjonen var lite konsistent. Det var hovedsaklig én utvikler på det feltet, og allikevel ble det ikke fulgt en standard da det kom til navnekonvensjoner. Dette førte til at utvikleren måtte gå igjennom mye kode mot slutten, og rydde opp i denne. Alle navn ble da endret, og fulgte en bestemt standard, men dette kostet noe ekstra tid.

Vi burde gjort en bedre jobb med å undersøke systemet til oppdragsgiver før vi startet på vårt eget system. Vi var klar over at det var noen forskjeller på systemet deres og det vi utviklet på, men det viste seg å lage mer jobb enn det vi ønsket da vi flyttet løsningen vår til deres system. Vi hadde laget mye kode for både mobil-applikasjonen og webløsningen som vi måtte fjerne under overflyttingen, blant annet alle PHP-filer. Hadde vi undersøkt litt mer rundt deres system før vi startet utviklingen, kunne vi utviklet med dette i tankene. Nå skal det sies at vi flyttet oss over til deres system av egen fri vilje, men vi var nok ikke helt klar over jobben det medførte.

Føring av arbeidslister var noe som burde blitt satt i et ordentlig system mye tidligere. I starten ble det det ført opp tidspunkt for når vi møttes, med dato, lengde på møtet, og hva som var temaet vi jobbet med. Dette fungerte ikke helt optimalt, da vi ikke følte dette ble detaljert nok. Vi gikk deretter over til å benytte oss av Toggl, men da hadde vi allerede mistet en del timer som ikke ble logget på en god nok måte. Dette førte til at arbeidslisten vår ikke ble komplett. I tillegg er flere av timene som er blitt logget, timer hvor vi har jobbet sammen. Disse er merket med "(alle)" i loggen. Dette er da blitt logget som en person, i stedet for to. Dette også gav et utslag på arbeidsrapporten.

6.3 Evaluering av gruppen

Vi synes gruppen har jobbet veldig godt sammen. Vi har kjent hverandre siden starten av det første semesteret, og har jobbet med flere oppgaver sammen. Dette gjorde at vi hadde en viss oversikt over nivået til hver enkelt på gruppa, og stolte på hverandre.

Vi har hatt daglige møter i ukene, hvor vi har møttes på skolen og jobbet sammen. Dette har skjedd omtrent uten unntak. Dette har gjort at vi har hatt et jevnt arbeidstempo, bra kommunikasjon, og vi har hatt muligheten til å ta avgjørelser etter behov. Hvis en person i gruppen ikke har hatt mulighet til å møte, hadde vi en fast rutine på hvordan, og når, dette skulle meldes ifra.

Vi har hatt mange diskusjoner, men vi har alltid kommet til en enighet. Ved for store uenigheter har vi bedt veileder eller oppdragsgiver om hjelp og meninger, og vi har kunne tatt avgjørelser basert på det.

6.4 Fremtidig arbeid

Utviklingsprosjekter blir i grunn aldri ferdige. De blir sluppet ut, for så å utvikles videre. Vi har allerede fått tilbud av oppdragsgiver om å fortsette utviklingen noe etter at denne oppgaven er innlevert. Dette er i hovedsak ferdigstilling av iOS-applikasjonen og webløsningen.

For videre utvikling av mobil-applikasjonen, vil det være logisk å få inn et system med prestasjoner som brukeren låser opp etterhvert som de benytter seg av applikasjonen. Dette er noe som brukere setter pris på, og som gir dem en større motivasjon for å benytte seg av løsningen.

For webløsningen vil det være naturlig å se på mulighetene for å vise mer statistikk om bruken av applikasjonen. Dette gjelder da spesielt historikk for å se utvikling over tid, og for å kunne sammenligne butikker og regioner. Det kan også være en god idé å kunne tilknytte brukere til byer eller enkelt-butikker. Slik systemet er nå, er brukere kun knyttet sammen med regioner. Disse regionene deler Norge i 3 biter, og det er derfor vanskelig (les: umulig) å vise statistikk for bare et lite område eller en spesifikk butikk.

6.5 Konklusjon

Selv om løsningen ikke er komplett, er vi fornøyde med prosjektet. Når vi først startet prosjektet hadde vi kun særlig relevant erfaring på Android, og av den grunn var det mye som tok lengre tid enn forespeilet. Vi har derimot lært veldig mye, spesielt innenfor Javascript, CSS, og utvikling for iOS. Vi har også fått mange erfaringer med å jobbe med et faktisk prosjekt, og det å forholde oss til en ekstern arbeidsgiver.

Vi har i løpet av dette prosjektet lært mye om strukturering av arbeidsdagen, og hvor viktig det er å holde god orden i arbeidsoppgavene. Vi har også fått erfare hvor lite som skal til for å forårsake forsinkelser i et prosjekt. Møtevirksomhet er noe vi ikke hadde mye erfaring med fra tidligere av, dette er noe vi nå føler oss sikrere på.

Vi ser fram til å videreutvikle denne løsningen fram mot sommeren. Vi har stor tro på at denne kan hjelpe butikkmedarbeidere med å lære seg PLU-koder, og at den kan fungere som et godt oppslagsverk som kan brukes daglig.

Bibliography

[1] IFPS. Plu-codes. http://www.ifpsglobal.com/Identification/PLU-Codes.

[2] Frugier, A. 2015. Using cursors with the new recyclerview. http://quanturium.github.io/2015/04/19/using-cursors-with-the-new-recyclerview.

[3] Google. Asynctask. https://developer.android.com/reference/android/os/AsyncTask.html.

[4] Google. Activities. https://developer.android.com/guide/components/activities.html.

[5] Google. Fragments. https://developer.android.com/guide/components/fragments.html.

[6] Oracle. Object ordering. https://docs.oracle.com/javase/tutorial/collections/interfaces/order.html.

[7] Christian Noon, K. F. Alamofire. https://github.com/Alamofire/Alamofire.

A Definisjoner

White-box testing

En form for testing som innebærer tilgang til koden.

Black-box testing

En form for testing som innebærer mangel på tilgang til koden.

Responsive internettsider

Internettsider som tilpasser seg enheten de blir sett på med.

HTML

Står for Hyper Text Markup Language, og er et programmeringsspråk for internettsider.

PDO

PHP Data Objects er en utvidelse for behandling av databaser i PHP.

AJAX

Asynchronous JavaScript and XML er en metode for å hente og sende data til og fra servere, og for å raskt endre dynamisk innhold på websider uten å laste hele siden på nytt.

jQuery

Et bibliotek for JavaScript. Blir brukt for enkel behandling av HTML dokumenter.

Hamburgermeny

En menyknapp. Ser ut som tre streker som ligger ovenfor hverandre, og viser en meny når de blir trykket eller klikket på.

Base64

En metode for konvertering av binære data til tekst. Kan f.eks. bruker for å konvertere et bilde til en lang tekststring.

Activity

B Forprosjekt

B.1 Mål og rammer

B.1.1 Bakgrunn

Vi har fått en oppgave fra Gitek om å lage et opplæringsverktøy for å hjelpe ansatte ved Coop Norge å lære PLU-koder. Price-Look-Up(PLU) brukes i butikker for å håndtere typisk løsvekt varer, fordi det er korte koder som gir en god tilknytning mellom vare og system.

Gitek AS leverer skreddersydde applikasjoner for bedrifter som ønsker bedre innsikt i driften gjennom dataanalyser som igjen er med på å effektivisere bedriften. Coop Norge er en av Giteks største kunder og de ønsker stadig nye applikasjoner som kan bidra til økt kundetilfredshet og effektivitet ute i butikkene.

B.1.2 Resultatmål

Ved prosjektets slutt skal følgende leveres:

- Et webgrensesnitt som en administrator kan bruke for å administrere kurs og oppgaver
- En mobil-applikasjon for Android hvor brukere kan øve på PLU-koder, delta på kurs og søke på varer
- Rankinglist for å se hvor mange poeng brukere har oppnådd på de forskjellige oppgavene

Utvidet funksjonalitet:

- Lage applikasjonen for iOS.

B.1.3 Effektmål

Følgende effektmål er gitt av Gitek:

- 50% mindre oppslag på PLU-kodetavle ved kasseområdet i løpet av ett år
- Fjerne papirversjoner av opplæringsmateriellet når det gjelder PLU-opplæring i løpet av ett år
- At 60% av alle ansatte benytter seg av applikasjonen utenom arbeidstid og oppsatte kurs

B.1.4 Rammer

- Mobil-applikasjonen skal fungere på minimum Android API level 16, som tilsvarer Android versjon 4.1
- Android-applikasjonen vil bli utviklet i Android-Java og XML
- Webgrensesnittet skal støttes fullt av Internet Explorer 11. Ønskelig at det også at det støtter nyeste versjon av Chrome og Firefox
- Webgrensesnittet vil bli utviklet i HTML, CSS og Javascript/jQuery
- Nødvendig tjenerfunksjonalitet vil bli utviklet i PHP og MSSQL

- All utvikling skal være ferdig og overlevert oppdragsgiver senest 18. Mai 2016.

B.2 Omfang

B.2.1 Fagområde

Price-look-up(PLU) koder brukes i butikker for å enkelt håndtere løsvekt varer som grønnsaker og frukt. Selv om de i seg selv er effektive og gir en god kobling mellom produkt og system, er de ikke veldig effektive om kasseoperatøren ikke kan dem.

PLU-koder ble tatt i bruk i 1990 i supermarkeder, og er en internasjonal standard satt av IFPS [1]. Selv om det er i utgangspunktet IFPS som bestemmer koder, har butikkjeder lov til å sette sine egne koder. Disse kodene er som regel 3-5 sifrede.

I dag eksisterer det ingen mobil-applikasjoner for opplæring av PLU-koder, og et oppslag av disse kodene er gjort via plasttavler som er satt opp ved kasseområdene. Det eksisterer et oppslagsverk for Android, men denne dekker kun IFPS sine koder.

B.2.2 Avgrensning

Vi skal lage en applikasjon som skal kunne brukes til opplæring av PLU-koder for medarbeidere i butikker, og som i tilleg skal kunne brukes som et oppslagsverk. Denne applikasjonen skal være til Android, men hvis tiden tillater det, ønsker oppdragsgiver at den også skal fungere på iOS.

Vi skal også lage et webgrensesnitt som skal brukes av administratorer. Der skal de kunne legge til og slette brukere, samt lage, endre og slette kurs. Disse kursene skal kunne bli assosiert til regioner, butikker, type vare og spesifikke brukere hvis ønskelig.

B.2.3 Oppgavebeskrivelse

Med Gitek PLU er tanken å tilby opplæring av PLU-koder til både nye og eksisterende medarbeidere på en tilgjengelig og mer engasjerende måte. For å oppnå dette skal Gitek PLU tilbys som en applikasjon på mobil, enten for iOS og/eller Android. For å gjøre det mer engasjerende er tanken å gjøre opplæring mer likt et spill. Det vil si et enkelt poengsystem og for eksempel vanskelighetsnivåer. Planen er at applikasjonen skal bli en fast del av opplæringen til nye ansatte.

For å sette opp og håndtere ulike opplæringer trengs det også et enkelt web-grensesnitt som administrator eller kursholder bruker. Denne må kunne opprette, endre og slette kurs samt håndtere brukere.

Funksjonalitet for mobil-applikasjonen:

- Ulike moduser for å lære PLU-koder. Trykk på korrekt PLU, eller tast komplett PLU.
- Enkelt belønningssystem i form av poenger
- Vanskelighetsnivåer
- Rankingliste/scoreboard
- Oppslagsfunksjon. Søke på PLU, varenavn eller varekategori
- Applikasjonen skal fungere uten nett-tilkobling

Funksjonalitet for webgrensesnitt:

- Legge til/lage nye kurs
- Assosiere kurs til brukere, butikk, slag eller region

- Endre og slette kurs
- Oversikt over progresjon til brukere
- Importering av PLU-koder
- Endre PLU-koder

Funksjonalitet som er listet opp er ikke absolutt og viser kun hovedtrekkene. Det må forventes at det tilkommer noen i utviklingsfasen når en ser behovet. Oppdragsgiver forventer innspill fra oss på dette.

Det vil også bli behov for å designe og utvikle database, samt utvikle nødvendig tjenerfunksjonalitet for å håndtere kommunikasjon med webgrensesnitt og applikasjonen.

Oppdragsgiver vil bistå med veiledning gjennom hele prosessen, og ønsker at terskelen for å kontakte de er lav.

B.3 Prosjektorganisering

B.3.1 Ansvarsforhold og roller

Da vi er en gruppe på kun to personer, vil ansvarsforholdene og rollene være delt mellom gruppens medlemmer. Eivind har derimot rollen som gruppeleder, og har derfor ansvaret for de endelige, og største avgjørelsene som angår gruppa og prosjektet.

Oppdragsgiveren vår er Gitek AS, som vil bistå med råd og veiledning. De vil også vurdere produktet underveis. Veilederen vår er Frode Haug.

B.3.2 Rutiner og regler i gruppa

- Forventninger
    - Alle gruppemedlemmer plikter seg til å jobbe minst 25 timer i uka hver. Disse timene inkludere også samlinger/møter hvor gruppemedlemmene jobber sammen
    - Det gies fritak fra påkrevd arbeidsinnsats på helligdager og i felles ferier. Det betyr derimot ikke at det er ulovlig å jobbe
    - Alle gruppemedlemmer skal gjøre det som blir avtalt, og til avtalt tid. Ved brudd på dette må det settes av ekstra tid for å komme ajour med arbeidet. Dette skal være utført innen utgangen av førstkommende sprint etter forsinkelsen
    - Ved ønske om ferie, må dette diskuteres internt i gruppa. Det må da tas hensyn til hvor lang tid som er igjen av prosjektet, og hvor mye jobb det er på de berørte sprintene. Hvis det blir godkjent ferie, kan det kreves opptil 50% arbeidsinnsats
- Møter
    - Alle gruppemedlemmer SKAL møte på avtalte møter, og til korrekt tid. Hvis dette ikke er mulig, skal alle gruppemedlemmer få beskjed så tidlig som mulig
    - Alle møter skal loggføres med et kort referat, tidpunkt og varighet
- Kostnader
    - Mindre kostnader for å dekke utgifter internt i gruppa, skal dekkes av gruppens medlemmer. Større kostnader som er nødvendig for å gjennomføre prosjektet diskuteres med oppdragsgiver

- Overtredelser og uenigheter
    - Ved uenighet om en avgjørelse, og det innen rimelig tid ikke er funnet en løsning, skal veileder og/eller kunde kontaktes for innspill
    - Etter gjentatte overtredelser av gruppereglene, og to skriftlige advarsler, kan gruppemedlemmer bli fjernet fra gruppa

B.4 Planlegging, oppfølging og rapportering

B.4.1 Systemutviklingsmodell

Grunnlag for valg av modell

Dette prosjekter består av mange forskjellige biter. Database-design, utvikling av et webgrensesnitt, samt design og utvikling av en mobil-applikasjon. Innenfor hvert at disse områdene vil det igjen være mange oppgaver. Dette medfører en blanding av arbeidsoppgaver og områder som krever forskjellig kompetanse. Det kan også være vanskelig å planlegge på tvers av disse oppgavene, da en del av feltene er avhengig av andre felter.

I tillegg kan det dukke opp endringer i kravspesifikasjonen etterhvert som utviklingen foregår. Vi ønsker tett oppfølging og jevnlige møter med oppdragsgiver, og dette gjør at vi må være fleksible når det gjelder tilbakemeldinger og endringer i prosjektet. Vi må være i stand til å endre planene våre på kort varsel, samtidig som vi må ha muligheten til å gjøre om på dokumentasjonen etterhvert som vi finner, eller oppdager, nye ting som må gjøres.

RUP og Fossefall

Disse to er rett og slett for store for vårt prosjekt. De krever lang planlegging, flere roller, har ikke så god fleksibilitet når det gjelder endringer og er rett og slett beregnet på større prosjekter med flere personer enn det vi er. I tilegg har oppdragsgiver ytret ønske om at vi ikke benytter oss av fossefallsmetoden.

Extreme Programming (XP)

XP er en modell som kan fungere for oss. Her blir ting utviklet og testet kjapt. I stedet for at alt blir dokumentert på en gang, for så å bli laget, har vi her muligheten til å tilpasse en del dokumentasjon og utvikling etterhvert. Det som gjør at denne ikke er helt optimal for oss, er at denne modellen er basert på at alle sitter sammen i par og utvikler. Dette betyr også at alle må kunne alt. Det medfører at vi må bruke enda lengre tid på å tilegne oss kunnskap, slik at alle kan noenlunde like mye. Siden vi bare er to personer i gruppa, gjør dette at vi blir mindre fleksible når det kommer til arbeidsoppgaver.

Scrum

Denne modellen har en en del til felles med XP. Vi får mer fleksibilitet når det kommer til endringer. Vi kan ha jevnlige møter hvor vi foretar endringer på dokumentasjon. Vi kan teste nye funksjoner fortløpende, og få raske tilbakemeldinger. Her kan vi fritt fordele arbeidsoppgavene mellom oss, og jobbe med forskjellige ting. Det eneste som virker negativt for vår del, er at det kreves en Scrum-master. Vi kommer til å se bort fra dette, og forholder oss til Gantt-diagrammet og ønsker fra oppdragsgiver.

Konklusjon

Utifra det vi har sett de forskjellige utviklingsmodellene har å tilby oss, har vi valgt å gå for Scrum. Vi kommer til å benytte oss av dette uten Scrum-master, men med tett oppfølging og samarbeid med både oppdragsgiver og veileder. Da det er mange forskjellige deler og systemer som skal på plass i dette prosjekter, har vi valgt å benytte oss av sprinter på en uke om gangen.

B.4.2 Plan for statusmøter og beslutningspunkter

Vi har avtalt faste møter med veileder hver tirsdag kl. 14:30, og med oppdragsgiver hver fredag kl. 10:00.

Vi i gruppa vil ha faste statusmøter hver dag kl. 10:00, unntak fredag, da vil møte foregå etter møtet med oppdragsgiver. Disse møtene vil enten foregå fysisk på skolen, eller via Skype.

B.5 Organisering av kvalitetssikring

B.5.1 Dokumentasjon, standardbruk og kildekode

Mobil-applikasjonen vil bli utviklet i Android-Java, og vi vil derfor benytte oss av Javadoc for dokumentering av funksjonalitet innad i applikasjonen. For struktur vil vi forholde oss til Google Java Style [**?**].

Dokumentering av web-applikasjon, tjenerfunksjonalitet og nettverkskommunikasjon vil bli gjort med en Wiki.

B.5.2 Konfigurasjonsstyring

Vi vil bruke Git med Bitbucket for versjonshåndtering av kode, med et prosjekt delt opp i server-del, web-del og applikasjons-del.

Vi vil bruke ShareLaTeX for håndtering av rapporter, og dokumentering underveis i prosjektet, da dette gir oss gode samskrivingsmuligheter.

B.5.3 Risikoanalyse

Under har vi utført vår risikoanalyse. Vi har Har skrevet opp relevante risikoer, og gitt disse en sannsynlighet og konsekvens. Skalaen for både sannsynlighet og konsekvens går fra 1 til 5, hvor en 1 lavest og 5 er høyest. Poengene blir så lagt sammen, og alt under 4 vil vi ikke gjøre noe med. 4 og 6 vil vi ha i bakhodet, og vi vil smått nevne disse. Fra og med en sum på 7 og oppover vil vi legge en plan for.

Analyse

NR	RISIKO	SANNSYNLIGHET	KONSEKVENS	SUM
1	Sykdom over 3 dager	2	2	4
2	Overskridelse av tidsfrist	2	5	7
3	Plutselige (store) endringer i kravspesifikasjonen	2	4	6
4	Løsningen får dårlig brukervennlighet	4	2	6
5	Kunden avbryter utviklingen	1	1	2
6	Mangel på teknisk kunnskap/ferdigheter	4	4	8

Kommentarer

NR	KOMMENTAR
1	Det er forholdsvis lite sannsynlige at noen av oss blir såpass syke at vi ikke får gjort noe på 3 dager. I tillegg bruker vi samskrivningsverktøy, Git og Skype, så vi får jobbet og kommunisert hjemmefra.
3	Her tenker vi på store endringer som vil forandre større deler av løsningen vår. Da vi har fått en ganske definert oversikt over hva som skal gjøre, ser vi ikke på dette som veldig sannsynlig. Skulle det derimot dukke opp, får vi fanget det opp ganske tidlig, siden vi har en ukers sprinter.
4	Vi har satt denne som ganske sannsynlig, da vi ikke har så stor erfaring med det å utvikle bruker-grensesnitt. Vi har derimot fått beskjed fra oppdragsgiver om at det ikke er noen krise, så lenge vi passer på at mobil-applikasjonen blir best mulig.

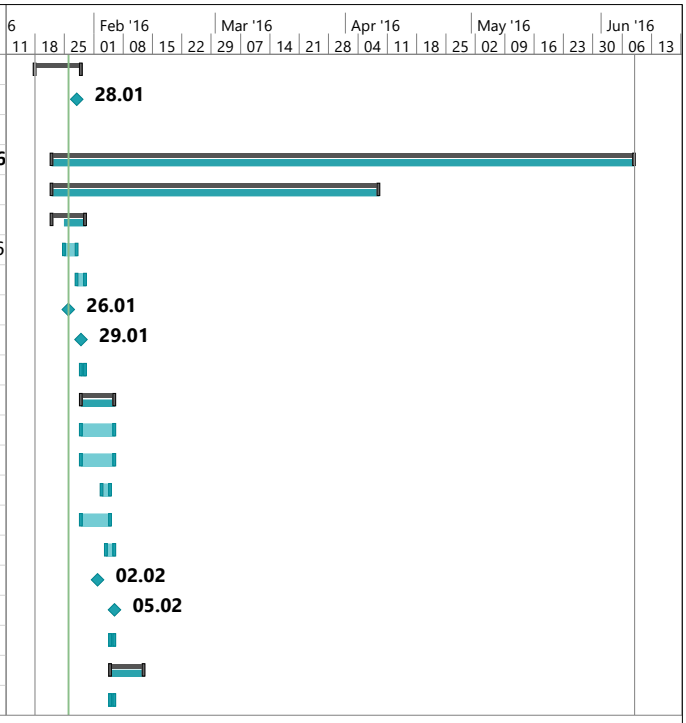
Tiltak

NR	TILTAK
2	Konsekvensen for å gå over tidsfristen for prosjektet er veldig stor, da det kan føre til at hele oppgaven ikke blir bestått. Her er det allerede en del tiltak som er i gang, derfor ser vi ikke på sannsynlighet som like stor. Vi benytter oss aktivt av planleggingsverktøy (bl. a. Gantt-skjema), har jevnlig møter med både veileder og oppdragsgiver, og jobber ofte sammen for å passe på at vi ikke blir stående fast med noe.
6	Vi har satt både konsekvens og sannsynlighet som ganske høyt på dette. Grunnen er at det er mange forskjellige teknologier som skal benyttes i dette prosjektet, og vi har ikke dyptgående kunnskap om alt. Dette kan også føre til at vi ikke klarer å gjennomføre oppgaven. Vi har allerede startet med tiltak på dette, ved at vi har samlet en del informasjon om relevante områder og teknologier. Vi benytter oss aktivt av oppslagsverk, og har tilgang til flere ressurspersoner i nærheten hvis vi skulle stå helt fast.

B.6 Plan for gjennomføring

B.6.1 Gantt-skjema

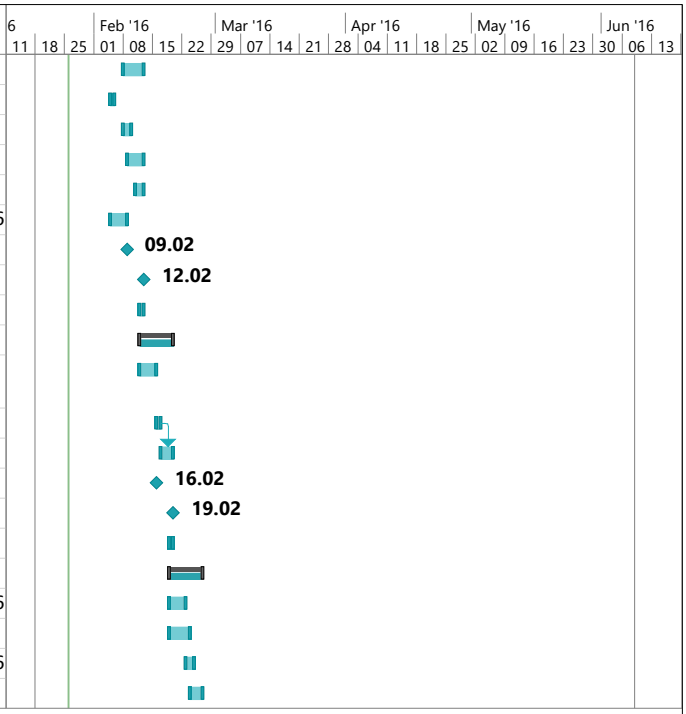
ID	Task Name	Duration	Start	Finish
1	**Forprosjekt**	**9 days**	**Mon 18.01.16**	**Thu 28.01.16**
2	Innlevering - Rapport	0 days	Thu 28.01.16	Thu 28.01.16
3				
4	**Prosjekt**	**99 days**	**Fri 22.01.16**	**Wed 08.06.16**
5	**Utvikling**	**56 days**	**Fri 22.01.16**	**Fri 08.04.16**
6	**Sprint 1**	**6 days**	**Fri 22.01.16**	**Fri 29.01.16**
7	Kravspesifikasjon	3 days	Mon 25.01.16	Wed 27.01.16
8	Prototyping	2 days?	Thu 28.01.16	Fri 29.01.16
9	Møte - Veileder	0 days	Tue 26.01.16	Tue 26.01.16
10	Møte - Oppdragsgiver	0 days	Fri 29.01.16	Fri 29.01.16
11	Oppdatering av dokumentasjon	1 day	Fri 29.01.16	Fri 29.01.16
12	**Sprint 2**	**6 days**	**Fri 29.01.16**	**Fri 05.02.16**
13	App - Material Design	6 days?	**Fri 29.01.16**	**Fri 05.02.16**
14	App - Modus for opplæring	6 days?	Fri 29.01.16	**Fri 05.02.16**
15	Databasedesign	2 days?	Wed 03.02.16	Thu 04.02.16
16	Web - Legge til / slette PLU	5 days	Fri 29.01.16	Thu 04.02.16
17	Web - Design av importfunksjon	2 days?	Thu 04.02.16	**Fri 05.02.16**
18	Møte - Veileder	0 days	Tue 02.02.16	Tue 02.02.16
19	Møte - Oppdragsgiver	0 days	Fri 05.02.16	Fri 05.02.16
20	Oppdatering av dokumentasjon	1 day	Fri 05.02.16	Fri 05.02.16
21	**Sprint 3**	**6 days**	**Fri 05.02.16**	**Fri 12.02.16**
22	Server - Oppsett database	1 day?	**Fri 05.02.16**	**Fri 05.02.16**



Project: Forprosjekt - Gantt
Date: Tue 26.01.16

Task
Split
Milestone
Summary
Project Summary
Inactive Task
Inactive Milestone
Inactive Summary
Manual Task
Duration-only
Manual Summary Rollup
Manual Summary
Start-only
Finish-only
External Tasks
External Milestone
Deadline
Progress
Manual Progress

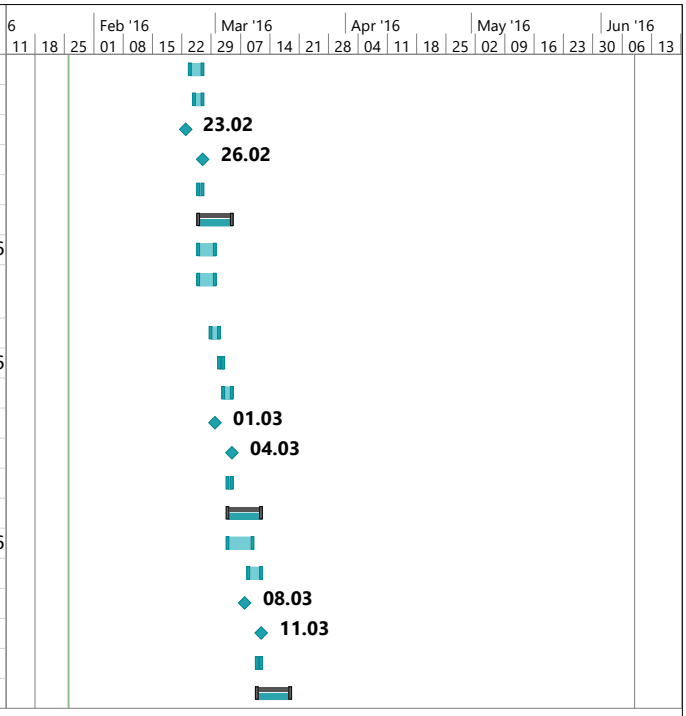
ID	Task Name	Duration	Start	Finish
23	Server - Legg til / hent vare	5 days?	Mon 08.02.16	**Fri 12.02.16**
24	App - Oppsett database	1 day?	**Fri 05.02.16**	**Fri 05.02.16**
25	App - Databasefunksjoner	2 days	Mon 08.02.16	**Tue 09.02.16**
26	Web - Legge til nytt kurs	4 days?	**Tue 09.02.16**	**Fri 12.02.16**
27	Server - Legge inn kurs	2 days?	Thu 11.02.16	**Fri 12.02.16**
28	Brukertest - App	2 days?	**Fri 05.02.16**	Mon 08.02.16
29	Møte - Veileder	0 days	Tue 09.02.16	Tue 09.02.16
30	Møte - Oppdragsgiver	0 days	Fri 12.02.16	Fri 12.02.16
31	Oppdatering av dokumentasjon	1 day	Fri 12.02.16	Fri 12.02.16
32	**Sprint 4**	**6 days**	**Fri 12.02.16**	**Fri 19.02.16**
33	Server - Implementering av import-funksjon	2 days	**Fri 12.02.16**	Mon 15.02.16
34	Brukertest I felt - App	1 day	Tue 16.02.16	Tue 16.02.16
35	Justeringer etter tilbakemeldinger	3 days	Wed 17.02.16	Fri 19.02.16
36	Møte - Veileder	0 days	Tue 16.02.16	Tue 16.02.16
37	Møte - Oppdragsgiver	0 days	Fri 19.02.16	Fri 19.02.16
38	Oppdatering av dokumentasjon	1 day	Fri 19.02.16	Fri 19.02.16
39	**Sprint 5**	**6 days**	**Fri 19.02.16**	**Fri 26.02.16**
40	Server - Hente kurs	2 days	**Fri 19.02.16**	Mon 22.02.16
41	Web - Legge til medarbeider	3 days	**Fri 19.02.16**	Tue 23.02.16
42	Server - Legge inn medarbeider	2 days	Tue 23.02.16	Wed 24.02.16
43	Server - Hente medarbeider	3 days?	Wed 24.02.16	Fri 26.02.16



Project: Forprosjekt - Gantt
Date: Tue 26.01.16

Task	Inactive Summary	External Tasks
Split	Manual Task	External Milestone
Milestone	Duration-only	Deadline
Summary	Manual Summary Rollup	Progress
Project Summary	Manual Summary	Manual Progress
Inactive Task	Start-only	
Inactive Milestone	Finish-only	

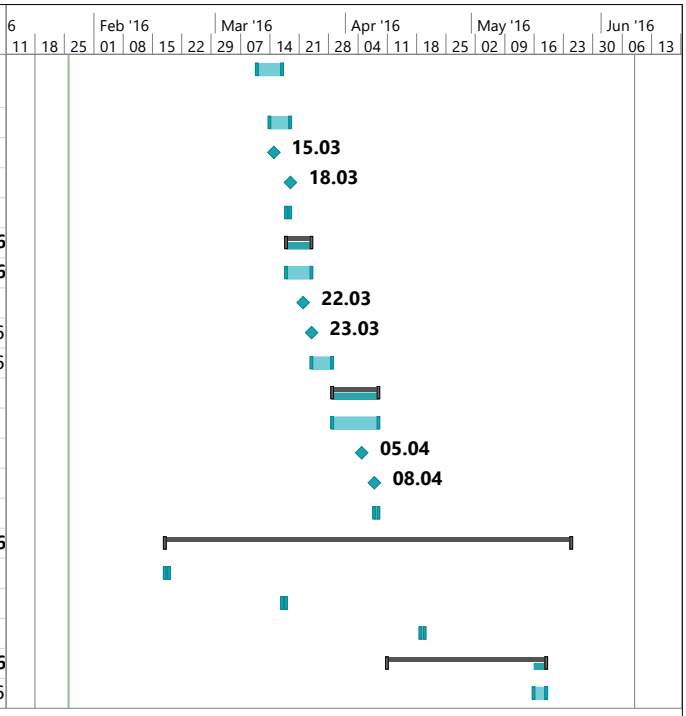
ID	Task Name	Duration	Start	Finish
44	App - Innloggingssystem	3 days	Wed 24.02.16	Fri 26.02.16
45	App - Belønningssystem / poengsyste	2 days	Thu 25.02.16	Fri 26.02.16
46	Møte - Veileder	0 days	Tue 23.02.16	Tue 23.02.16
47	Møte - Oppdragsgiver	0 days	Fri 26.02.16	Fri 26.02.16
48	Oppdatering av dokumentasjon	1 day	Fri 26.02.16	Fri 26.02.16
49	**Sprint 6**	**6 days**	**Fri 26.02.16**	**Fri 04.03.16**
50	Server - Legge til / hente region / but	2 days?	**Fri 26.02.16**	Mon 29.02.16
51	Web - Assosiere kurs til brukere, butikk, slag eller region	2 days	**Fri 26.02.16**	Mon 29.02.16
52	Web - Endre / slette kurs	2 days	Mon 29.02.16	Tue 01.03.16
53	Administratortest - Web	1 day?	Wed 02.03.16	Wed 02.03.16
54	Justering etter tilbakemeldinger	2 days?	Thu 03.03.16	Fri 04.03.16
55	Møte - Veileder	0 days	Tue 01.03.16	Tue 01.03.16
56	Møte - Oppdragsgiver	0 days	Fri 04.03.16	Fri 04.03.16
57	Oppdatering av dokumentasjon	1 day	Fri 04.03.16	Fri 04.03.16
58	**Sprint 7**	**6 days**	**Fri 04.03.16**	**Fri 11.03.16**
59	App - Vanskelighetsgrader	4 days	**Fri 04.03.16**	Wed 09.03.16
60	App - Rankingliste / scoreboard	3 days?	Wed 09.03.16	Fri 11.03.16
61	Møte - Veileder	0 days	Tue 08.03.16	Tue 08.03.16
62	Møte - Oppdragsgiver	0 days	Fri 11.03.16	Fri 11.03.16
63	Oppdatering av dokumentasjon	1 day	Fri 11.03.16	Fri 11.03.16
64	**Sprint 8**	**6 days**	**Fri 11.03.16**	**Fri 18.03.16**



Project: Forprosjekt - Gantt
Date: Tue 26.01.16

Task
Split
Milestone
Summary
Project Summary
Inactive Task
Inactive Milestone
Inactive Summary
Manual Task
Duration-only
Manual Summary Rollup
Manual Summary
Start-only
Finish-only
External Tasks
External Milestone
Deadline
Progress
Manual Progress

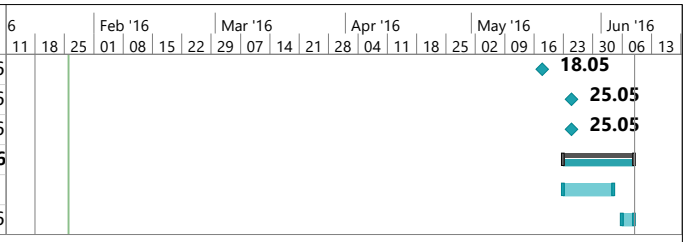
ID	Task Name	Duration	Start	Finish
65	Web - Oversikt over progresjonen til brukere	4 days	**Fri 11.03.16**	Wed 16.03.16
66	App - Søkefunksjon	5 days?	Mon 14.03.16	**Fri 18.03.16**
67	Møte - Veileder	0 days	Tue 15.03.16	Tue 15.03.16
68	Møte - Oppdragsgiver	0 days	Fri 18.03.16	Fri 18.03.16
69	Oppdatering av dokumentasjon	1 day	Fri 18.03.16	Fri 18.03.16
70	**Sprint 9**	**4 days**	**Fri 18.03.16**	**Wed 23.03.16**
71	App - Widget for søking	4 days?	**Fri 18.03.16**	**Wed 23.03.16**
72	Møte - Veileder	0 days	Tue 22.03.16	Tue 22.03.16
73	Møte - Oppdragsgiver	0 days	Wed 23.03.16	Wed 23.03.16
74	Påskeferie	3 days	Thu 24.03.16	Mon 28.03.16
75	**Sprint 10**	**9 days**	**Tue 29.03.16**	**Fri 08.04.16**
76	App - Push notifications	9 days?	**Tue 29.03.16**	**Fri 08.04.16**
77	Møte - Veileder	0 days	Tue 05.04.16	Tue 05.04.16
78	Møte - Oppdragsgiver	0 days	Fri 08.04.16	Fri 08.04.16
79	Oppdatering av dokumentasjon	1 day	Fri 08.04.16	Fri 08.04.16
80	**Dokumenter**	**69 days**	**Thu 18.02.16**	**Wed 25.05.16**
81	Statusrapport 1	1 day?	Thu 18.02.16	Thu 18.02.16
82	Statusrapport 2	1 day	Thu 17.03.16	Thu 17.03.16
83	Statusrapport 3	1 day?	Tue 19.04.16	Tue 19.04.16
84	**Rapport**	**28 days**	**Mon 11.04.16**	**Wed 18.05.16**
85	Innlevering - Rapport	3 days	Mon 16.05.16	Wed 18.05.16



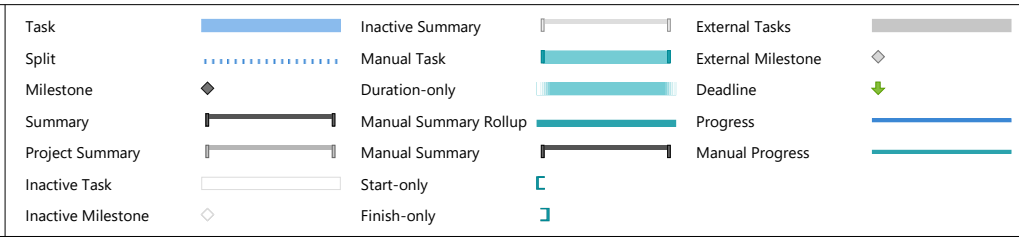
Project: Forprosjekt - Gantt
Date: Tue 26.01.16

Task
Split
Milestone
Summary
Project Summary
Inactive Task
Inactive Milestone
Inactive Summary
Manual Task
Duration-only
Manual Summary Rollup
Manual Summary
Start-only
Finish-only
External Tasks
External Milestone
Deadline
Progress
Manual Progress

ID	Task Name	Duration	Start	Finish
86	Siste frist - Innlevering 12:00	0 days	Wed 18.05.16	Wed 18.05.16
87	Plakat til utstilling	0 days	Wed 25.05.16	Wed 25.05.16
88	Individuelt refleksjonsnotat	0 days	Wed 25.05.16	Wed 25.05.16
89	**Framføring**	**13 days**	**Mon 23.05.16**	**Wed 08.06.16**
90	Forberedelse	10 days	Mon 23.05.16	Fri 03.06.16
91	Framføringer	3 days	Mon 06.06.16	Wed 08.06.16



Project: Forprosjekt - Gantt
Date: Tue 26.01.16



C Grupperegler

Grupperegler

Dette dokumentet inneholder regler for gruppen. Alle reglene gjelder for alle gruppemedlemmer, med mindre annet er skrevet i regelen. Alle gruppemedlemmer skal følge samtlige regler.

Forventninger

- Alle gruppemedlemmer plikter seg til å jobbe minst 25 timer i uka hver. Disse timene inkludere også samlinger/møter hvor gruppemedlemmene jobber sammen
- Det gies fritak fra påkrevd arbeidsinnsats på helligdager og i felles ferier. Det betyr derimot ikke at det er ulovlig å jobbe
- Alle gruppemedlemmer skal gjøre det som blir avtalt, og til avtalt tid. Ved brudd på dette må det settes av ekstra tid for å komme ajour med arbeidet. Dette skal være utført innen utgangen av førstkommende sprint etter forsinkelsen
- Ved ønske om ferie, må dette diskuteres internt i gruppa. Det må da tas hensyn til hvor lang tid som er igjen av prosjektet, og hvor mye jobb det er på de berørte sprintene. Hvis det blir godkjent ferie, kan det kreves opptil 50% arbeidsinnsats

Møter

- Alle gruppemedlemmer SKAL møte på avtalte møter, og til korrekt tid. Hvis dette ikke er mulig, skal alle gruppemedlemmer få beskjed så tidlig som mulig
- Alle møter skal loggføres med et kort referat, tidspunkt og varighet

Kostnader

- Mindre kostnader for å dekke utgifter internt i gruppa, skal dekkes av gruppens medlemmer. Større kostnader som er nødvendig for å gjennomføre prosjektet diskuteres med oppdragsgiver

Overtredelser og uenigheter

- Ved uenighet om en avgjørelse, og det innen rimelig tid ikke er funnet en løsning, skal veileder og/eller kunde kontaktes for innspill
- Etter gjentatte overtredelser av gruppereglene, og to skriftlige advarsler, kan gruppemedlemmer bli fjernet fra gruppa

Ved å signere dette dokumentet har jeg bekreftet at jeg har lest og forstått alle reglene:

Daniel Rosland
Daniel Rosland

Eivind Ristebråten
Eivind Ristebråten

D Prosjektavtale

NTNU
Norges Teknisk-Naturvitenskapelige Universitet
NTNU i Gjøvik, Avd. Informatikk og Medieteknikk

PROSJEKTAVTALE

mellom NTNU v/Avd. Informatikk og Medieteknikk (NTNU/AIMT) (utdanningsinstitusjon), og

GITEK AS (oppdragsgiver), og

EIVIND RISTE BRÅTEN, DANIEL ROSLAND (student(er))

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

1. Studenten(e) skal gjennomføre prosjektet i perioden fra 1/1-16 til 18/5-16.

   Studentene skal i denne perioden følge en oppsatt fremdriftsplan der AIMT 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 AIMT 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 Gjøvik/AIMT. Studentene dekker utgifter for ferdigstillelse av prosjektmateriell.
   - 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. AIMT 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 (intern og ekstern sensor). Likevel er det en forpliktelse for utøverne av prosjektet å fullføre dette til avtalte spesifikasjoner, funksjonsnivå og tider.

4. Alle bacheloroppgaver som ikke er klausulert og hvor forfatteren(e) har gitt sitt samtykke til publisering, kan gjøres tilgjengelig via NTNUs institusjonelle arkiv hvis de har skriftlig karakter A, B eller C.

   Tilgjengeliggjøring i det åpen arkivet forutsetter avtale om delvis overdragelse av opphavsrett, se «avtale om publisering» (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.

Den totale besvarelsen med tegninger, modeller og apparatur så vel som programlisting, kildekode mv. som inngår som del av eller vedlegg til besvarelsen, kan vederlagsfritt benyttes til undervisnings- og forskningsformål. Besvarelsen, eller vedlegg til den, må ikke nyttes av AIMT 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 NTNU/AIMT og/eller studenter har interesser.

NTNU
Norges Teknisk-Naturvitenskapelige Universitet
NTNU i Gjøvik, Avd. Informatikk og Medieteknikk

6. 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.

7. 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.

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

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

10. I det enkelte tilfelle kan det inngås egen avtale mellom oppdragsgiver, student(er) og AIMT som regulerer nærmere 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 med oppdragsgiver, skjer dette uten AIMT som partner.

11. Når NTNU/AIMT også opptrer som oppdragsgiver, trer NTNU/AIMT inn i kontrakten både som utdanningsinstitusjon og som oppdragsgiver.

12. 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.

13. Deltakende personer ved prosjektgjennomføringen:

NTNU/AIMTs veileder (navn): [signature] FRODE HAUG

Oppdragsgivers kontaktperson (navn): Khai Van Ngo

Student(er) (signatur): Eivind Riksfjorden dato 27/1-16

Daniel Rosland dato 28/01-16

dato _____

dato _____

Oppdragsgiver (signatur): [signature] dato 27-01-16

Signert avtale leveres digitalt i Fronter(IMT3912)
Godkjennes digitalt av AIMTs dekan

Om papirversjon med signatur er ønskelig, må papirversjon leveres til AIMT i tillegg.
Plass for evt sign:
AIMT Dekan/prodekan (signatur): _____ dato _____

E Statusrapporter

E.1 Statusrapport 1

Planlegging - framdriftsplan

Framdriftsplanen blir oppdatert etter hvert uke med oppdragiver. Dette er noe vi har gjort helt siden starten, og som vi synes fungerer veldig godt. Vi kommer derfor til å fortsette med dette. Vi synes også vi har klart å planlegge oppgavene våre ganske bra fra uke til uke.

Organisering

Vi har beholdt organiseringen av selve gruppen slik som beskrevet i forprosjektrapporten. Videre har vi internt avtalt at Daniel har hovedansvaret for mobil-applikasjonen, mens Eivind har hovedansvaret for web-utviklingen.

Løsningsmetode - koding

Kodingen går litt opp og ned. Vi har hatt perioder hvor det har gått veldig bra, og andre perioder hvor det ikke har fått så bra. Vi har også hatt flere revisjoner av koden vår, både når det gjelder applikasjonen for mobil, og applikasjonen for web. Det er mye nytt å sette seg inn i for oss begge, og dette har satt sitt preg på arbeidstempoet.

Rapportskriving

Ingen oppdatering på dette punktet.

E.1.1 Muligheter? Trusler/problemer

Oppdragsgiver har gitt ønske om at mobil-applikasjonen også skal bli utviklet til iOS. Det er noe vi har satt som en mulighet, men vi har sagt klart ifra at vi ikke kan garantere at den vil bli ferdigutviklet. Dette var ikke noe problem for oppdragsgiver. Vi har noen mindre problemer i utviklingen, men ikke noe vi foreløpig ser på som trusler.

E.1.2 Hva er avsluttet?

Ingenting er foreløpig avsluttet, da oppdragsgiver ønsker at vi skal få inn mest mulig funksjonalitet før vi finpusser og avslutter. Dette gjør at vi har større fokus på å lage funksjonalitet som fungerer, i forhold til finpusset funksjonalitet.

Foreløpig ferdig funksjonalitet

Funksjonalitet som er regnet som fungerende, men ikke er ferdigpusset er:

- To spillmoduser på mobil-applikasjon.
- Trenings-modus på mobil.
- Kurs-modus på mobil.
- Legge til varer på web.
- Importering av varer på web.
- Legge til kurs på web.

E.1.3 Hva er under arbeid?

Vi beregner bare arbeid for en og en uke, da vi har en-ukers sprinter. Dette gjør at det på en gitt tid ikke er så mye som er under arbeid. Vi jobber kontinuerlig med funksjoner som allerede er å regne som fungerende, da disse ofte henger sammen med nye funksjoner.

Under arbeid

Funksjonalitet som er å regne som under arbeid pr. 19.02.2016:

- Fjerne varer du har lagt til ved oppretting av kurs på web.
- Brukerhåndtering:
    - Legge til brukere.
    - Slette brukere.
    - Innlogging for brukere på mobil, med Google Sign-in.
- Legge til bilder av varer.
- Planlegging og gjennomføring av brukertest i felt.
- Belønnings-/poengsystem for mobil-applikasjonen.
- Loggføring av brukerstatistikk.

E.1.4 Tidsfrister

Vi har ikke hatt noen store problemer med tidsfristene våre. Det har hendt vi har måtte prioritert litt anderledes enn det som var planlagt, men det har gått bra. Vi har hatt noen småting som har gått litt over den tiden det skulle, men det har vi klart å ta inn igjen veldig raskt.

E.1.5 Motivasjon

Samarbeid, arbeidsformer og organisering

Samarbeidet har fungert utmerket, da vi har hjulpet hverandre hver gang den andre har hatt problemer. Dette har ført til at mange problemer har tatt vesentlig kortere tid å løse enn hvis vi skulle løst de alene. For å opprettholde godt samarbeid har vi valgt å møte hver ukedag for å jobbe sammen. Da jobber vi i all hovedsak med separate oppgaver, med mindre en skulle ha noen problemer, eller vi behøver tilbakemelding og / eller ideer fra den andre.

Forhold som oppmuntrer eller frustrerer

Vi har hatt noen problemer som vi har brukt langt tid på å løse, og som av og til resulterer i kasting av kode og funksjonalitet. Dette merker vi er litt frustrerende. Når vi derimot merker at problemene løser seg ved å starte på nytt, øker motivasjonen igjen. Mye av det vi har gjort så langt, har gitt godt synlige resultater veldig raskt. Dette er også noe som hjelper veldig på motivasjonen.

E.1.6 Veilederkontakt

Vi synes kontakten med veileder har fungert veldig godt. Vi har fått godt med tilbakemeldinger på møtene, og vi får veldig raskt svar når det er noe vi lurer på. Vi har ingen problemer med å få tak i veileder, så vi er veldig fornøyde med det.

E.2 Statusrapport 2

Planlegging - framdriftsplan

Det er blitt litt endringer i planene våre, dette har med at vi akkurat nå driver og flytter oss over på systemet til oppdragsgiver. Mer om dette lengre ned. Ellers har ting gått mer eller mindre som planlagt.

Organisering

Ingen endring siden forrige rapport. Daniel har hovedansvaret for mobil-applikasjonen, mens Eivind har hovedansvaret for web-utviklingen.

Løsningsmetode - koding

Kodingen har gått ganske greit. Det har vært både vanskelige og lettere perioder, og begge to har lært en god del i det siste. Daniel er i tillegg i gang med å lære seg Swift for iOS. Det ble også bestemt at mobilapplikasjonen skulle gå over til å bruke Realm.io som databaase-system, i stedet for SQLite. Dette medfører en del endringer i kode, men vi er sikre på at dette vil lønne seg.

Rapportskriving

Ingen oppdatering på dette punktet.

E.2.1 Muligheter? Trusler/problemer

Vi er nå i gang med å lage applikasjon for iOS, og dette kan da ta litt tid. Det viser seg å være litt utfordringer som må løses i forbindelse med dette, hovedsaklig i forbindelse med design av GUI. Vi er også i gang med å flytte oss over på systemet til oppdragsgiver nå. Dette gjør at vi blir satt litt tilbake, da vi må gjøre opp igjen ganske mye av det som allerede er gjort. Det blir nye funksjoner, og systemet til oppdragsgiver støtter ikke PHP, noe vi hadde basert all kommunikasjonen til både web og mobil på.

E.2.2 Hva er avsluttet?

Akkurat nå er det fortsatt ingenting som vi regner som helt avsluttet. Dette har å gjøre med at ting blir endret i forhold til sprinter, og at vi å driver med tilpasninger i forhold til oppdragsgivers systemer.

Foreløpig ferdig funksjonalitet

Vi har funksjonalitet på både web og mobil, og dette var fungerende før bytting av system. Det vil si at funksjonene er der, og vi må bare få på plass igjen ny og oppdatert database, og kommunikasjon med den:

- To spillmoduser på mobil-applikasjon (android).
- Trenings-modus på mobil (android).
- Kurs-modus på mobil (android).
- Legge til varer (web).
- Importering av varer (web).
- Visning av kurs, samt delt liste med pågående og avsluttede kurs (web).
- Fargekode på kurs for å visualisere hvilke kurs som holder på å bli avsluttet (web).
- Legge til kurs (nå med inkludering av varer) (web).
- Legge til og vise ansatte (web).

- Startet på dashbord med visning av data (web).

E.2.3 Hva er under arbeid?

Akkurat nå er det utvikling av applikasjon for iOS, sammen med overflytting til oppdragsgivers systemer som har første prioritet. Sistnevnte medfører b.la:

- Overflytting til oppdragsgivers database.
- Legge til/endre tabeller i nevnte database.
- Oppdragsgiver må lage funksjoner for kommunikasjon med database.
- Fjerne alt vi har av kommunkasjon via PHP, og opprette ny kommunikasjon/nye funksjoner via AJAX.

E.2.4 Tidsfrister

Vi har ikke hatt noen store problemer med tidsfristene våre. Det har hendt vi har måtte prioritert litt anderledes enn det som var planlagt, men det har gått bra. Vi har hatt noen småting som har gått litt over den tiden det skulle, men det har vi klart å ta inn igjen veldig raskt.

E.2.5 Motivasjon

Samarbeid, arbeidsformer og organisering

Samarbeidet har fungert utmerket, da vi har hjulpet hverandre hver gang den andre har hatt problemer. Dette har ført til at mange problemer har tatt vesentlig kortere tid å løse enn hvis vi skulle løst de alene. For å opprettholde godt samarbeid har vi valgt å møte hver ukedag for å jobbe sammen. Da jobber vi i all hovedsak med separate oppgaver, med mindre en skulle ha noen problemer, eller vi behøver tilbakemelding og / eller ideer fra den andre.

Forhold som oppmuntrer eller frustrerer

Vi har hatt noen problemer som vi har brukt langt tid på å løse, og som av og til resulterer i kasting av kode og funksjonalitet. Dette merker vi er litt frustrerende. Når vi derimot merker at problemene løser seg ved å starte på nytt, øker motivasjonen igjen. Mye av det vi har gjort så langt, har gitt godt synlige resultater veldig raskt. Dette er også noe som hjelper veldig på motivasjonen.

E.2.6 Veilederkontakt

Vi synes kontakten med veileder har fungert veldig godt. Vi har fått godt med tilbakemeldinger på møtene, og vi får veldig raskt svar når det er noe vi lurer på. Vi har ingen problemer med å få tak i veileder, så vi er veldig fornøyde med det. Nå skal det også sies at vi har hatt noen hektiske dager av og til, og dette har medført at vi rett og slett har glemt møtet med veileder ett par ganger.

E.3 Statusrapport 3

Planlegging - framdriftsplan

Det har oppstått noen forsinkelser pga. overflyttingen til Gitek sitt system, men de fleste problemene er løst nå. Vi har løpende dialoger med Gitek, og følger en oppdatert liste over arbeidsoppgaver fra uke til uke.

Organisering

Ingen endring siden forrige rapport. Daniel har hovedansvaret for mobil-applikasjonen, mens Eivind har hovedansvaret for web-utviklingen.

Løsningsmetode - koding

Daniel er blitt tryggere på Swift, så programmeringen går litt lettere for iOS. Det er fortsatt en del nytt og en del annerledes syntax å sette seg inn i. Vi har også hatt et par problemer med JavaScript for web, men vi har fått hjelp fra Gitek ved behov.

Rapportskriving

Ingen oppdatering på dette punktet.

E.3.1 Muligheter? Trusler/problemer

Utvikling til iOS kan ta lengre tid enn forventet, da vi ikke kjenner det systemet så godt. Vi kan også få litt utfordringer ved bruk av Gitek sin database, da vi ikke har noen kontroll eller måte å administrere denne på, og er avhengige av å bruke Gitek som et mellomledd.

E.3.2 Tidsfrister

Vi er blitt noe forsinket pga. overgangen til et annet system. Utvikling er derfor satt tilbake en del. Vi holder Gitek oppdatert på dette, og de er inneforstått med, og har godkjent, at det vil oppstå forsinkelser i utviklingen.

E.3.3 Motivasjon

Samarbeid, arbeidsformer og organisering

Det samme som ved forrige status: Samarbeidet har fungert utmerket, da vi har hjulpet hverandre hver gang den andre har hatt problemer. Dette har ført til at mange problemer har tatt vesentlig kortere tid å løse enn hvis vi skulle løst de alene. For å opprettholde godt samarbeid har vi valgt å møte hver ukedag for å jobbe sammen. Da jobber vi i all hovedsak med separate oppgaver, med mindre en skulle ha noen problemer, eller vi behøver tilbakemelding og / eller ideer fra den andre.

Forhold som oppmuntrer eller frustrerer

Det samme som ved forrige status: Vi har hatt noen problemer som vi har brukt langt tid på å løse, og som av og til resulterer i kasting av kode og funksjonalitet. Dette merker vi er litt frustrerende. Når vi derimot merker at problemene løser seg ved å starte på nytt, øker motivasjonen igjen. Mye av det vi har gjort så langt, har gitt godt synlige resultater veldig raskt. Dette er også noe som hjelper veldig på motivasjonen.

E.3.4 Veilederkontakt

Det samme som ved forrige status: Vi synes kontakten med veileder har fungert veldig godt. Vi har fått godt med tilbakemeldinger på møtene, og vi får veldig raskt svar når det er noe vi lurer på. Vi har ingen problemer med å få tak i veileder, så vi er veldig fornøyde med det.

F Møtelogg

Vedlegg for møteloggen.

Veileder, 14.01.16

Møte med veileder. Snakket litt om hvordan vi burde starte på forprosjektrapporten. Avtalte ukentlige møter, Tirsdager klokken 14:30.

Gitek, 15.01.16

Møte med oppdragsgiver. Fikk utlevert kravspesifikasjon, snakket litt rundt det praktiske og snakket om forventninger. Avtalte ukentlige møter. Fredager klokken 10:00.

Gitek, 22.01.16

Møte med oppdragsgiver. Gikk gjennom spørsmål vi hadde sendt tidligere. Fikk utlevert eksempel på eksport av PLU-koder. Fant ut at vi skulle bruke Gitek sin Github. Fikk tilbakemelding på det vi hadde i rapporten så langt.

Gitek, 05.02.16

Kommentarer fra Gitek: WEB:

- Knapper på høyresiden i tabellen av varer, legge in popup med dialog.
- Legg til filtrering av varer
- Legg til at søking merker hvilken tekst som matcher.
- Sjekk ut WAMP / AMPPS
- Bruk forside som et dashboard, med informasjon og grafer, sammenligninger etc.
- Bruk Bootstrap+bilder som en link til bilder.
- Lag importeringsfunksjon som tar imot .csv filer.
- Sjekk ut slimphp
- Splitt opp PHP i kategorier, med en fil per kategori.
- Bruk try/catch for hindre potiensielle feil.

Gitek, 12.02.16

Diskuterte oppsett av brukere, samt løsninger for å føre statistikk over brukere. Vi ble også enige om merking av nesten utgåtte kurs.

Gitek, 19.02.16

Diskuterte bilder av varer. Ble enige om at bilder skal lagres på filserver, med link i database. Vi avtalte at vi skulle produsere en APK for Android, og sende de den så de kunne teste applikasjonen.

Gitek, 25.02.16

Snakket med Gitek ang. bilder av varer. Fant ut at vi skulle flytte oss over på systemet dems. Dette medfører en liten stopp i utviklingen, da mye må gjøres om og tilpasses.

Gitek, 26.02.16

Ble enige om å lage en knapp i egen kolonne i tabeller for sletting av varer og kurs. Gitek savnet også bedre tilbakemelding til brukeren i web-grensesnittet, så vi skal implementere en bedre løsning for å gi beskjed hva som skjer (Slettet et kurs, lagt til vare med plu XXXX, etc)

Gitek, 04.03.16

Vanlig ukentlig møte med Gitek, men i tillegg hadde vi en gjennomgang i systemet dems.

Gitek, 11.03.16

Vanlig møte med Gitek, de ønsket av vi skulle flytte "Vis logg" til hovedmeny på web, og at vi skulle sende de database-design og nødvendig funksjonalitet.

Gitek, 18.03.16

Til etter påsken skal vi sette opp nye implementasjonen av kurs, og få oppdatert databasen på Android

Gitek, 01.04.16

Vi skal muligens få tilgang til Gitek sin implementasjon av database første halvdel av neste uke.

Gitek, 08.04.16

De har fjernet PLUGameType og PLUUserItems tabellene, ellers er implementering av database snart ferdig.

Gitek, 15.04.16

Vi fikk noe hjelp med implementeringen av bilder på web.

Veileder, 19.04.16

Fikk noen tips til oppsett av rapport.

Gitek, 22.04.16

De ønsker endringer i fargevalg, de bør tilpasses fargeblinde. De ønsker seg også en intro-side på iOS.

Veileder, 26.04.2016

Møte med veileder for å gå gjennom første utkast av rapportorganisering

Gitek, 29.04.16

Hvis vi ikke rekker Google Sign-in, skal vi lage egen innloggingsløsning. Ikke-aktive kurs skal kun vises hvis brukeren har deltatt på det.

Gitek, 06.05.16

De ønsker at vi skal prioritere rapporten. De skal fikse feil på bruker-registrering på deres side. Hvis applikasjonen pauses under kurs, skal timer fortsette å gå. De påpekte også en skrivefeil i applikasjonen.

Gitek, 13.05.16

De ønsker at vi skal få inn topp XX lister per kurs på web, og plassering og farge på slett-knappen skal endres. Implementere en knapp for å vise resultater per bruker. Vise høyste poengsum siste uke på web.

G Arbeidsliste

På grunn av systemet til ShareLatex, og det at vi importerer arbeidslisten vår fra PDF, fører dette til at overskriften på dette kapittelet står for seg selv, mens selve listen kommer på neste side.

Projects / Time entries	Duration
Bachelor - GitekPLU	**287:46:51**
App - Databasefunksjonalitet - Kurs	6:35:51
App - Fiks bug med nedlasting av kurs	0:20:03
App - Fullføring av foreløpig databasefunksjonalitet	5:56:01
App - generell bugfix	0:16:04
App - Implementering av Google achievements	3:51:25
App - Implementering av Google Sign-in	3:30:20
App - Implementering databasefunksjoner	4:25:07
App - Justering etter tilbakemeldinger	2:40:00
App - konvertering fra Course til PLUCourse	3:48:47
App - konvertering fra parts til pluitems	1:17:29
App - Material Design	6:31:59
App - Material Design	5:07:30
App - Menysystem	6:01:34
App - Oppdatering av nedlasting	1:22:56
App - Oppsett av kurs	2:23:54
App - Oppsett database	4:22:00
App - Søkefunksjon og widget	5:36:00
Databasedesign	3:30:00
Forprosjektrapport (alle)	21:45:00
Førsteutkast av rapport-layout	0:34:37
Hovedprosjektrapport (alle)	31:05:39
iOS - Course Details view	1:50:08
iOS - Course score view	0:39:10
iOS - Implementing database	4:04:10
iOS- Implementing plu and bilde modus	6:36:05
iOS - Lesing av dokumentasjon / video	15:26:00
kravspesifikasjon - oppdatering	0:31:47
Kravspesifikasjon - Use Case - Web diagram	0:30:00
Kravspesifikasjon - Use Case - Web høynivå	1:20:00
Lynkurs 1 (alle)	2:00:00
Oppdatering av Trello	2:54:08
Overgang til Gitek sitt system - Kurs (alle)	10:07:00
Overgang til Gitek sitt system - Varer (alle)	17:12:00
Prototyping (alle)	5:15:00
Undersøke opplæringsmaterialet	2:00:00
Utarbeidelse av grupperegler (alle)	1:00:00
Utarbeidelse av kravspesifikasjon (alle)	4:00:00

Web - Alert som lukker seg, og logg, ved oppretting av kurs	2:04:38
Web - Ansatte, slette og endre	1:30:00
Web - Bekreftelse ved lagring av varer	0:49:46
Web - Bilde av vare på hover i tabell	0:16:16
Web - Bugfiks fortsatt	0:27:13
Web - Bugfix, rader for å legge varer i kurs	1:03:00
Web - Celle for å legge til/fjerne varer i kurs	4:42:06
Web - Dashboard til web	1:10:27
Web - Dele kurs i aktive og avsluttede	1:42:46
Web - Detaljert oversikt for hvert enkelt kurs	3:01:31
Web - Endre og slette varer	3:30:50
Web - Endring av kurs	2:38:00
Web - Farge utgående kurs, endre varer, slette varer, teste app og generell bugfiksing	5:19:00
Web - Farging av rader for kurs (bootstrap for fargeblinde)	0:19:44
Web - Fjerne funksjon for legge til/endre brukere (skal gjøres fra telefon/app)	0:40:00
Web - Fungerende datepicker for IE11 og FF	0:48:00
Web - Hente medarbeider	1:23:00
Web - Importering av PLU-koder (csv)	1:57:30
Web - Justering etter tilbakemeldinger	4:55:00
Web - Kommentering og generell opprydding i kode	1:22:57
Web - Kontroll tomme felt for varer	1:57:00
Web - Konvertere bilder til blob	0:47:40
Web - Korrigering av funksjons-/variabelnavn	2:23:28
Web - Lagre varer sammen med kurs	0:21:44
Web - Lagring av varer i kurs (stoppet pga. feil i Ajax/DB)	0:53:00
Web - Legge inn kontroller av felter ved å legge inn kurs	1:31:00
Web - Legge inn kontroller av felter ved å legge inn varer	0:19:43
Web - Legge inn regioner for ansatte	1:28:00
Web - Legge til ansatte	0:31:40
Web - Legge til farger i kursoversikt, samt forklaring på fargekoder	1:02:00
Web - Legge til medarbeider	3:59:00
Web - Legge til varer i kurs	3:03:08
Web - Loading animasjon på alle sider	0:32:00
Web - Loading animasjon ved lasting av tabeller	0:49:00
Web - Mer fjerning av alerts ved oppretting av varer.	1:30:00
Web - Minityrbilder i tabell, forstørring ved mouseover, og pnotify ved varslinger i varer. Samt kontroll av felter.	1:44:00
Web - Opplasting av bilder av varer	2:30:14
Web - Opplasting, forhåndsvisning, og henting av varebilder	3:08:27

Web - Oppretting av kurs - Gamemode	0:28:36
Web - Oppretting av kurs - sende til DB	2:34:54
Web - Oppretting av kurs - vareliste	1:36:47
Web - Overflytting til Gitek - vise vareliste	2:35:00
Web - Sende vare til database	2:43:05
Web - Serverside Google Cloud Messaging (gcm)	0:44:06
Web - Skaffe varer i korrekt kurs	0:49:07
Web - Slette varer	0:48:00
Web - Sletting av kurs	0:26:00
Web - Sortere pågående og utgåtte kurs, fargekode rader	0:47:00
Web - Starte på statistikk over på kurs/varer på forside	0:36:03
Web - Tabeller for aktive og avsluttede kurs	1:35:00
web - Valg av region ved oppretting av kurs	0:27:42
Web - Vareliste - Vise bilde var knappetrykk	1:43:59
Web - Visning av pågående kurs, og oppretting av nye kurs	5:11:00

Created with toggl.com