FooBadges

Open Badge-system bestående av en mobilapplikation och en webbapplikation

OSAMA MENIM
MARTIN NGUYEN

BACHELOR´S THESIS

DEPARTMENT OF ELECTRICAL AND INFORMATION TECHNOLOGY
FACULTY OF ENGINEERING | LTH | LUND UNIVERSITY

Examensarbete

FooBadges

Open Badge-system bestående av en mobilapplikation och en webbapplikation

Osama Menim & Martin Nguyen

Department of Electrical and Information Technology
Faculty of Engineering, LTH, Lund University
SE-221 00 Lund, Sweden

Tryckt i Sverige
Lunds universitet
Lund 2017

Sammanfattning

FooBadges är ett initiativ från Foo Café där visionen är att öka deltagandet på event inom Foo Café och minska antalet personer som väljer att inte komma till ett event de anmält sig till av någon anledning.

För att öka deltagandet på event inom Foo Café och minska antalet personer som anmäler sig till event men väljer att inte komma skulle ett förslag på ett Open Badge-system tas fram. Ett Open Badge-system utfärdar och distribuerar badges åt användare. Systemet med namnet FooBadges skulle bestå av en mobilapplikation och en webbapplikation som skulle appliceras som ett tillägg till den redan existerande webbapplikationen som Foo Café använder sig av. I samarbete med kunden bestämdes det att mobilapplikationen skulle utvecklas för Android.

Examensarbetet kartlägger olika Open Badge-system på marknaden och utifrån dessa utvecklas ett eget system för Foo Café. En kartläggning för att undersöka varför personer anmäler sig till event men inte kommer görs även och bearbetas för att undersöka ifall det är möjligt att uppmuntra personer att delta i event med hjälp av ett Open Badge-system.

Examensarbetet resulterade i en prototyp i form av en Android-applikation och en webbapplikation som inkorporerades i den redan existerande webbapplikationen som Foo Café använder. I Android-applikationen kan användaren logga in för att få möjlighet att anmäla sig till olika sorters event och för att utföra en incheckning till ett event på Foo Café. En användare behöver därmed inte längre använda en webbläsare för att anmäla sig till ett event och kan istället använda mobilapplikationen.

I Android-applikationen har en användare även möjlighet till att visa samtliga badges som Foo Café har skapat och samtliga badges som en användare har fått. Användaren kan därmed kontrollera vad som krävs för att uppnå specifika mål för att belönas med badges och kontrollera vilka badges användaren hittills har fått.

I webbapplikationen har administratörerna i Foo Café möjlighet till att skapa egna badges med specifika kriterier och designa egna badges. Administratörerna har även tillgång till en specifik sida som ingår i administratörssidan, där en administratör får en lista på samtliga användare som anmält sig till ett event och en lista på vilka användare som varit på eventet och därmed utfört en incheckning till eventet med hjälp av mobilapplikationen som utvecklades. Med hjälp av denna information kan en administratör säkerställa att en deltagare fysiskt varit närvarande och deltagit i ett event.

Nyckelord: Badge, Mobilapplikation, Webbapplikation, Android, Foo Café, Event, JSON

Abstract

FooBadges is an initiative by Foo Café where the vision is to increase the participation in events at Foo Café and reduce the number of people who choose not to attend an event they have signed up for.

To increase the participation in events at Foo Café and reduce the number of people who sign up for an event but choose not to attend, a proposal for an Open Badge-system should be presented. An Open Badge-system issues and distributes badges for users. The system called FooBadges would consist of a mobile application and a web application that would be applied as an add-on to the already existing web application that Foo Café uses. In cooperation with the customer, it was determined that the mobile application should be developed for Android devices.

The thesis maps various Open Badge-systems that are available on the market and, based on it, develop our own system for Foo Café using existing systems as a starting point. A survey to investigate why people sign up for an event but choose to not show up is also done and processed to investigate if it is possible to encourage people to attend events using an Open Badge-system.

The thesis resulted in a prototype in the form of an Android-application and a web application that was incorporated into the already existing web application that Foo Café uses. In the Android-application, the user can log in to have the opportunity to sign up for various events and to check-in to an event at Foo Café. A user does no longer need to use a browser to sign up for an event and can instead use the mobile application.

A user is also able to view all the badges that Foo Café has created and all the badges that a user has received. The user can thus check what is required to achieve specific goals to get rewarded with badges and check which badges the user has achieved so far.

In the web application, administrators in Foo Café have the opportunity to create their own badges with specific criteria and design their own badges. Administrators also have access to a specific page contained in the Admin page, where an administrator receives a list of all users who signed up for an event and a list of users who attended the event and performed a check-in to the event using the mobile application that was developed. With the help of this information, an administrator can ensure that an attendee has been physically present and has attended the event.

Keywords: Badge, Mobile application, Web application, Android, Foo Café, Event, JSON

Förord

Detta examensarbete har genomförts i samarbete med Foo Café i Malmö. Det har gett oss möjligheten att ta första steget in i arbetslivet och utvidgat våra kunskaper inom programmering och ökat vår förståelse för samtidens teknik. Det har varit en utmanande avslutning på våra tre år på högskoleingenjörsutbildningen i datateknik vid Lunds Tekniska Högskola.

Vi vill tacka grundaren av Foo Café, Michael Tiberg, som gett oss den här möjligheten att utföra vårt examensarbete på en mötesplats som Foo Café. Michael har bidragit med hjälp och värdefulla synpunkter under examensarbetets gång.

Ett stort tack till vår handledare Steven R. Baker på Foo Café som hjälpt oss och väglett oss under examensarbetets gång. Steven har varit väldigt hjälpsam och svarat på samtliga frågor som ställts till honom. Vi är väldigt tacksamma över att ha fått Steven som vår handledare för detta examensarbete.

Vi vill även tacka vår handledare på Lunds tekniska högskola, Christin Lindholm, som har väglett oss under examensarbetets gång och besvarat samtliga frågor angående vårt examensarbete.

Slutligen vill vi tacka vår examinator på Lunds tekniska högskola, Christian Nyberg, som besvarat ett flertal frågor angående vårt examensarbete.

Innehållsförteckning

Kapitel 1

1 Inledning

I detta kapitel ges en introduktion till examensarbetet som utförts hos Foo Café. Beskrivning av vad som har utförts och vilka Foo Café är kommer att tas upp i detta kapitel. Examensarbetets syfte, målformulering, problemformulering och avgränsningar kommer även behandlas.

1.1 Bakgrund

Examensarbetet har utförts hos Foo Café. Foo Café är en organisation, eller snarare en mötesplats där människor träffar varandra för att dela med sig av sina kunskaper inom ämnen som berör IT-världen som exempelvis Internet of Things (IoT). I Foo Café brukar det på vardagar hållas event som består av seminarier och workshops som behandlar nya tekniker eller språk inom IT-världen.

Personer som besöker Foo Café är intresserade av att lära sig om exempelvis nya programmeringsspråk och möta nya människor inom sitt yrkesområde. Hos Foo Café sprids kunskaper och färdigheter som deltagarna kan ta med sig i sitt dagliga arbete. De flesta som deltar i event är privatpersoner som besöker Foo Café på sin fritid och övriga är exempelvis föreläsare eller organisatörer av diverse event.

Information om deltagarna och deras anmälan till olika event samlas i en databas. Den information som sparas är vilka event en person anmält sig till. Dock sparas ingen information om en person fysiskt varit närvarande och deltagit i ett event.

Med hjälp av informationen om deltagarna och deras anmälan till olika event samt en verifiering att en deltagare varit på plats på ett event kan Foo Café dock säkerställa att en deltagare fysiskt varit närvarande och deltagit i ett event. Detta kan därmed användas som ett bevis av en deltagare att denna person deltagit i ett event och kan med hjälp av det bygga på sitt CV.

För att göra det möjligt för deltagarna att bevisa att de har varit närvarande på ett event ville Foo Café skapa ett system med namnet FooBadges, som lagrar information om vilka event en person har deltagit i och sedan belöna denna person med så kallade “badges”. Efter ett X antal event får personen ett certifikat (badge) av Foo Café som bevis på fördjupande kunskaper inom ett visst område. Ett exempel: om en person har deltagit i tre event om programmeringsspråket Java får personen ett certifikat som bevis på att personen deltagit i Java event och har en fördjupad kunskap inom Java. Detta certifikat kan därmed användas i ett CV.

En badge som används av ett Open Badge-system är ett verifierbart och portabelt insignium som intygar att en person utfört en uppgift eller visat en förmåga som exempelvis ha avklarat en kurs eller deltagit i ett särskilt event. En badge kan jämföras med märken och pins som används idag för att exempelvis belöna barn som simmar med olika simmärken efter att barnen klarat

specifika mål. Ett exempel på ett simmärke är “Hajen brons” där märkestagaren ska simma 100 meter på djupt vatten. Badges bygger på samma princip som märken och pins, där skillnaden är att badges är digitala. En badge innehåller en bild och information om vad som krävs för att tilldelas denna badge, vem som delat ut och vem som mottagit denna badge. Datumet för när en badge utdelats finns även med i en badge.

Idag finns det olika typer av Open Badge-system ute på marknaden som exempelvis Basno, TrueCred och Mozilla Open Badges. Det sistnämnda är det Open Badge-system som används av de flesta aktörerna idag och är det system som var först ute på marknaden och som skapade hela konceptet med badges.

Det finns många olika aktörer som använder sig av Mozilla Open Badges som exempelvis IBM, Khan Academy och FitBit [1]. Företaget IBM använder sig av Open Badges genom att belöna sina anställda med badges för exempelvis avklarade online-kurser [2]. Ett flertal applikationer som använder sig av Open Badges kan inte användas och utnyttjas för dokumentering av meriter. Ett exempel på en mobilapplikation som använder sig av badges är Untappd [3], där badges delas ut som belöning för antalet ölsorter som har druckits. Badges för antalet ölsorter som har druckits kan inte exempelvis användas i ett CV som badges utfärdat från Foo Café kommer att göra.

För att en deltagare i ett event ska kunna få en badge utfärdat från Foo Café måste denna deltagare vara fysiskt närvarande på detta event. Aktörer som använder sig av Open Badges använder sig inte av en fysisk plats som Foo Café kommer att göra för att säkerställa att en person uppnått ett mål och verkligen förtjänar att få en badge som belöning, vilket är unikt. Ett exempel är mobilapplikationen Untappd, där verifieringen av att en person druckit en sorts öl inte behöver utföras på en fysisk plats och behöver inte heller verifieras av en särskild person. Det finns därmed inga faktiska bevis på att personen druckit en öl. Badges som kommer att utfärdas av Foo Café kommer även att verifieras av en tredje part som i detta fall är Mozilla, vilket medför hög tillförlitlighet.

Andra organisationer som anordnar event som exempelvis Meetup.com har enligt uppgift från Foo Café ingen liknande form av Open Badge-system som kan utfärda badges till sina deltagare och verifiera att dessa deltagare har varit på plats. Foo Café strävar därmed efter att använda Open Badges på ett unikt sätt som inte finns ute på marknaden.

1.1.1 Foo Café

Foo Café är en mötesplats där människor träffas för att inspireras av varandra genom att dela information med varandra om exempelvis erfarenhet inom olika programmeringsspråk eller olika arbetsmiljöer och på så sätt lära sig från varandra. Foo Café underlättar detta genom att arrangera event där seminarier eller workshops anordnas för att sprida kunskap om nya tekniker och programmeringsspråk bland deltagarna i dessa event. Foo Café event är gratis och tillgängliga för alla som är intresserade av ämnen som berör IT-världen [4] [5].

Michael Tiberg startade Foo Café 2012 i Malmö och 2016 startades Foo Café även i Köpenhamn och Stockholm. Michael Tiberg har arrangerat den största IT-konferensen i Sverige, Øredev, som arrangeras en gång per år. Genom att starta Foo Café kunde dessa IT-konferenser hållas i en mindre skala och mer frekvent, det vill säga vardagligen. Detta innebär att människor kan utbyta information med varandra mer frekvent i jämförelse med IT-konferensen Øredev [6].

Huvudsakliga syftet med Foo Cafés verksamhet är lärandet och spridandet av kunskap genom event och vara en nätverkslänk för deltagarna som vill hitta nya kontakter.

1.2 Syfte

Syftet med examensarbetet är att utveckla ett Open Badge-system för Foo Café som ska bidra till att fler personer ska delta i Foo Cafés event. Denna typ av system ska bidra med att belöna deltagarna med badges som ska agera som ett slags certifikat där de exempelvis kan ha i sina CV vid jobbansökan som bevis på utökade kunskaper inom ett visst område. I detta examensarbete kommer en kartläggning av olika Open Badge-system att utföras. Kartläggningen kommer därmed att leda till att ett av dessa system kommer att tillämpas inom ramen för detta examensarbete. Konceptet spelifiering (gamification) kommer att undersökas och hur det kan tillämpas till systemet.

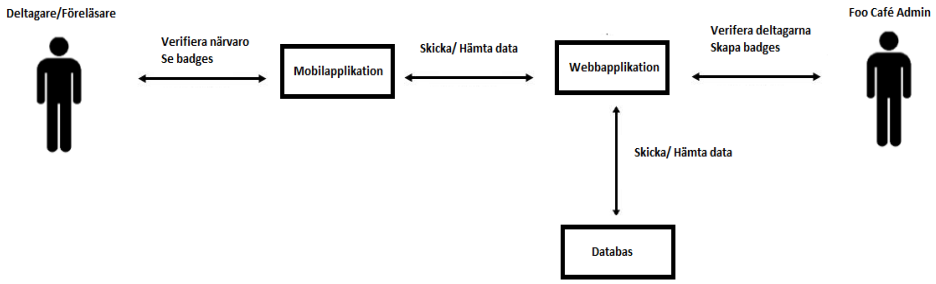
1.3 Målformulering

Målet med examensarbetet är att kartlägga olika Open Badge-system och utifrån undersökningen välja ett Open Badge-system för att utveckla ett system för Foo Café. En mobilapplikation ska utvecklas där personer som deltagit i ett event kan exempelvis se vilka badges och certifikat som har uppnåtts. Deltagarna ska använda sig av mobilapplikationen för att verifiera att de har varit på ett event och är berättigade att få en badge utfärdad. Extra funktionalitet ska även läggas till den redan existerande webbapplikationen som Foo Café i nuläget använder sig av. Webbapplikationen kommer att kunna användas av en administratör för att skapa badges samt verifiera deltagarnas närvaro vid event. Foo Cafés nuvarande databas ska uppdateras med nya olika tabeller och kommer därmed sättas upp för att ha kontroll på deltagarna och deras deltagande i event och de resulterade badges som de har fått.

En kartläggning ska även göras för att undersöka varför personer anmäler sig till event men inte kommer och denna kartläggning ska bearbetas för att undersöka ifall det är möjligt att uppmuntra personer att delta i event med hjälp av ett Open Badge-system. Detta ska utföras genom ett flertal intervjuer på olika event på Foo Café.

I figur 1 visas hur hela systemet fungerar och hur de olika applikationerna interagerar med varandra, där deltagarna använder sig av mobilapplikationen för att exempelvis se sina badges eller verifiera att de har deltagit i ett event. En administratör kommer att ha tillgång till webbapplikationen för att skapa och dela ut olika badges till deltagarna. Open badge-systemet

kommer att implementeras till applikationerna och båda applikationerna kommer att dela på samma databas.



Figur 1. Översikt över kommunikationen mellan applikationerna och databasen

1.4 Problemformulering

Följande frågeställningar kommer att besvaras i examensarbetet:

- Vilka Open Badge-system finns på marknaden och hur passar de till denna tillämpning?
- Hur sker integreringen av det valda Open Badge-systemet? Hur säker är denna typ av integrering? Kan deltagarna hacka sig in och okontrollerat skapa badges?
- Kartläggning av deltagarna vid event genom intervjuer. Detta ska utföras för att få deltagarnas syn på ett potentiellt Open Badge-system. Varför deltar de i olika event? Kan ett Open Badge-system uppmuntra deltagande i event?
- Vilka typer och nivåer av badges ska implementeras? Typer och nivåer avser vilken nivå som eventet riktar sig mot, det vill säga exempelvis om nivån är på en avancerad nivå ska en “Avancerad nivå” badge utdelas.
- På vilket sätt kommer webbapplikationen, mobilapplikationen och databasen att integrera med varandra?
- Hur ska deltagarna använda mobilapplikationen för att verifiera att de har varit närvarande vid ett event?
- Hur ska Foo Café använda sig av webbapplikationen för att skapa badges?

- Hur ska konceptet spelifiering (gamification) användas och hur ska det implementeras tillsammans med systemet?

1.5 Motivering av examensarbetet

Utvecklingen av ett Open Badge-system kommer att leda till att deltagare av event på Foo Café blir belönade med olika sorters badges som bevis på utökad kunskap inom olika ämnen. Detta kommer troligtvis leda till att fler personer kommer att dra sig till Foo Café och börja delta i fler event. Ju fler event en person deltar i, desto mer kunskap får denna person samt ett större kontaktnätverk som då kan möjliggöra eventuella jobberbjudanden från diverse kontakter. Utfärdandet av badges kommer även att leda till att deltagare får en större möjlighet till att få ett jobb, eftersom badges utfärdade av Foo Café kommer som sagt att kunna användas i arbetslivet och läggas till i ett CV.

Detta potentiella Open Badge-system är något som vi anser kan gynna samhället väldigt mycket då företag kan få ett större inblick på en persons kompetens och ambition vid läsning av ett CV innehållande badges i jämförelse med ett CV utan badges.

1.6 Avgränsningar

Foo Café har avgränsat mobilapplikationen till Android vilket betyder att mobilapplikationen inte kommer att skapas för iOS och Windows.

Kapitel 2

2 Teknisk bakgrund

I detta kapitel ges en teknisk bakgrund om de verktyg som använts i examensarbetet. Den utvecklingsmiljö som använts för utveckling av applikationerna, Open Badge-system som kartlades vid utvecklandet av applikationerna och databasen som använts av Foo Café vid utvecklandet av applikationerna beskrivs i detta kapitel. Information om konceptet spelifiering har behandlats och en teknisk bakgrund gällande beacons och positionering som mobilapplikationen använder sig av har även bearbetats.

2.1 Android

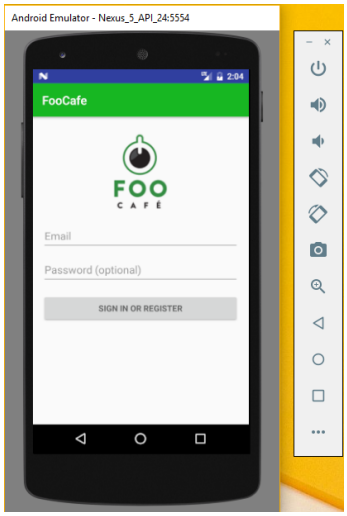
Android är ett mobilt operativsystem som utvecklades av företaget Android Inc år 2003, som numera ägs av Google. Android använder Linux operativsystemskärna och används främst för mobiltelefoner och pekplattor. Android används dock även inom bland annat TV-branschen, där Android utvecklat en smart TV-plattform vid namnet Android TV [26].

Utvecklaren använder sig av Android Software development kit (SDK) vid utvecklandet av Android-applikationer. Det vanligaste programmeringsspråket vid utvecklande av Android-applikationer är Java, som i vissa fall kan kombineras med programmeringsspråket C/C++.

2.2 Android Studio

Android Studio är en integrerad utvecklingsmiljö (IDE) för utveckling av Android-applikationer och är baserad på IntelliJ IDEA som är en integrerad utvecklingsmiljö för Java. Google lanserade Android Studio 1.0 under år 2014 och vid tidpunkten för detta examensarbete användes Android Studio 2.2.3, vilket även var den senaste versionen av Android Studio [7].

Några funktioner som Android Studio erbjuder är exempelvis ett Gradle-build system som gör kompilering av koden lättare att utföra genom att automatisera byggandet av applikationer. En annan funktion som Android Studio erbjuder är emulering av applikationer i olika mobila enheter, vilket effektiviserar testningen av applikationerna i olika mobila miljöer innan en eventuell lansering av applikationen kan ske. Ett exempel kan ses i figur 2.1, där en applikation testas för mobilen Nexus 5 [27]. Android Studio erbjuder även drag & drop funktioner för att designa layouten i applikationer och renderar detta i realtid, vilket medför att utvecklaren kan se designen av sin layout direkt i utvecklingsmiljön [8].

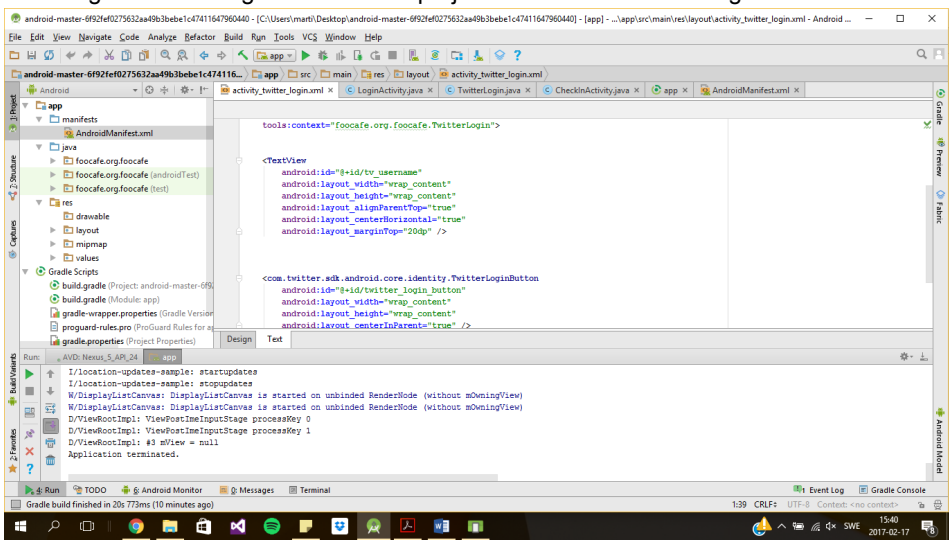


Figur 2.1. En applikation som körs på en Android-emulator av Nexus 5.

Android-projekt som skapas i Android Studio har samma struktur där alla build-filer är lokaliserade under Gradle Scripts och följande mappar medföljer:

1. Manifests - innehåller endast en AndroidManifest.xml fil.
2. Java - innehåller källkod för projektet.
3. Res - innehåller alla resurser som exempelvis bilder, ljud och XML-filer för layouten.

Visualisering av användargränssnittet och projektets struktur kan ses i figur 2.2.



Figur 2.2. Användargränssnitt för Android Studio.

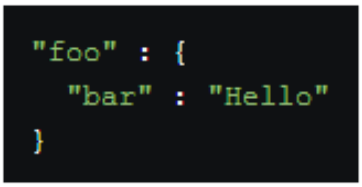
2.3 JSON

JavaScript Object Notation även känt som JSON är ett format för att strukturera data. Användning av JSON leder till en organiserad struktur och gör det enkelt för en användare att läsa respektive skriva data i JSON-format. JSON används exempelvis för kommunikation mellan en server och en webbapplikation [43]. JSON är uppbyggt på två strukturer:

- Objekt består av “key-value” par, vilket innebär att ett värde är kopplat till en nyckel. Klammerparenteser används för att markera att det är ett objekt.
- Vektor är en strukturerad lista av värden där hakparenteser används för att markera att det är en vektor.

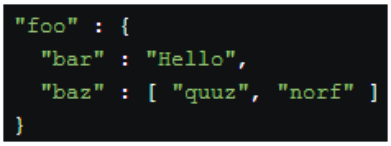
JSON är en universell datastruktur och har stöd för många olika programmeringsspråk [44]. Ett exempel på ett bibliotek som hanterar JSON i Java är RetroFit [45] som är en HTTP-klient som kan konvertera JSON-objekt till Java-objekt och tvärtom. Konvertering från JSON till Java utförs genom att exempelvis hämta en JSON-fil från en server, medan konvertering från Java till JSON utförs genom att skicka Java-objekt till en server.

I figur 2.3 visas ett exempel på ett JSON-objekt där ”foo” är namnet på objektet och ”key-value” paret består av nyckeln ”bar” och är kopplat till värdet ”Hello”. Objektet är omgivet av klammerparenteser för att markera att det är ett JSON-objekt



Figur 2.3. Ett JSON-objekt

I figur 2.4 visualiseras ett exempel på ett JSON-objekt där ”foo” är namnet på objektet och innehåller en nyckel ”baz” som är kopplat till en vektor som består av ”quuz” och ”norf”. Användningen av hakparenteser visar på att det är en vektor av värden som är kopplat till nyckeln.



Figur 2.4. Ett JSON-objekt som har en vektor associerad till en nyckel

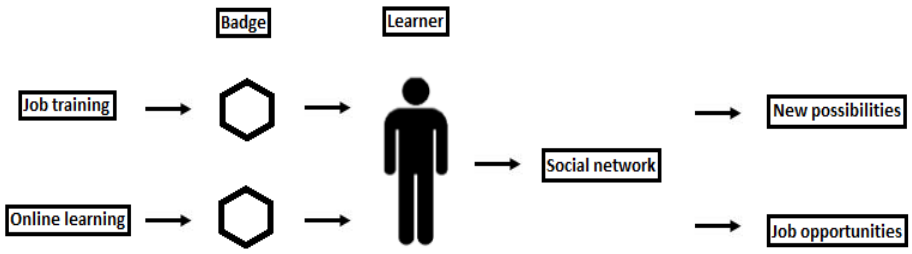
2.4 Badge-system

En digital badge är ett portabelt insignium som intygar att en person utfört en uppgift eller uppvisat en färdighet som exempelvis en avklarad kurs eller deltagit i ett särskilt event [28]. En badge kan jämföras med märken som används av scouter för att uppvisa vad de har gjort, deras intressen och tillhörighet. För att skapa olika badges existerar olika badge-system som hanterar skapandet och utfärdandet av badges. I detta kapitel kommer ett par av dessa badge-system översiktligt beskrivas.

2.4.1 Mozilla Open Badges

Mozilla skapade Open Badges år 2011 [29] med stöd av MacArthur Foundation och ett nätverk av olika utvecklare vars mål var att utveckla ett nytt sätt att värdesätta kunskap och inlärning av kunskap. Open Badges blev därmed ett innovativt alternativ för att belöna kunskap i jämförelse med exempelvis betyg.

Aktörer skapar och utfärdar Open Badges som en person kan samla. Aktörer som använder sig av Open Badges är exempelvis skolor, universitet, organisationer och företag [9]. Varje individuell badge innehåller data som exempelvis beskriver kraven som uppfylldes för att få badgen samt information om organisationen som utfärdade badgen. Denna data kan sedan uppvisas och delas på diverse sätt som exempelvis på hemsidor, i CV och i sociala medier som exempelvis Facebook eller LinkedIn. Visualisering för hur en person kan samla och använda sig av Open Badges kan ses i figur 2.5.



Figur 2.5. Flödesdiagram för hur Open Badges samlas och hur de kan användas. Inspirerat av Mozilla Open Badges flödesdiagram [42].

2.4.1.1 Processen för att skapa en Open Badge.

En beskrivning av processen som utförs för att skapa en Open Badge [10].

Steg 1. Skapa en badge-klass

En badge-klass innehåller information om badgen och skrivs i JSON format. Informationen som finns i en badge-klass är beskrivning på vad det är för badge, hur badgen ser ut, kraven som har uppfyllts för att få badgen, namnet på badgen och vem som har utfärdat badgen. Ett exempel på översikten för en badge-klass kan ses i figur 2.6, där det även är möjligt att lägga till fler fält än de som finns i exemplet.

```
{
    "name": "Rookie Badge Issuer",
    "description": "Issues great new Open Badges to self.",
    "image": "http://yoursite.com/rookie-badge-issuer.png",
    "criteria": "http://yoursite.com/rookie-badge-criteria.html",
    "issuer": "http://yoursite.com/badge-issuer.json"
}
```

Figur 2.6. En badge-klass i JSON format.

Steg 2. Skapa en design för badgen.

Den visuella representationen av badgen bör vara en fyrkantig bild i png-format med minimala dimensioner av 90x90 pixlar och ha en maximal filstorlek på 256kb. Det finns inga andra begränsningar för bilden i png format än de som har nämnts för designen av badgen.

Steg 3. Skapa en beskrivning om utfärdarna av badgen.

En beskrivning av aktören som utfärdar badgen ska skapas i JSON format där information om aktören och dess verksamhet bör finnas med. En aktör kan i detta sammanhang vara exempelvis en person eller en organisation. Ett exempel på hur en beskrivning kan se ut kan ses i figur 2.7, där det även är möjligt att ha med flera fält än de som finns i exemplet.

```
{
    "name": "New Badge Issuer",
    "url": "http://yoursite.com"
}
```

Figur 2.7. Beskrivning om aktören som har utfärdat badgen.

Steg 4. Uppladdning av filerna.

Ladda upp de tre filer som har skapats i föregående stegen till en server där de är tillgängliga.

Steg 5. Utfärda badgen till en person.

Skapa en beskrivning av personen som har fått badgen i JSON format och i samma JSON-fil länka till badgen som personen ska få. Ett exempel på beskrivningen kan ses i figur 2.8.

```
{
  "uid": "a1b2c3d4e5",
  "recipient": {
    "type": "email",
    "identity": "you@yourdomain.com",
    "hashed": false
  },
  "issuedOn":  1388534400,
  "badge": "http://yoursite.com/rookie-badge-class.json",
  "verify": {
    "type": "hosted",
    "url": "http://yoursite.com/rookie-badge-award.json"
  }
}
```

Figur 2.8. Beskrivning om personen som har fått badgen.

I figur 2.8 finns det information om exempelvis e-postadressen som tillhör personen som ska få badgen och hur länkningen till badge-klassen fungerar. Ladda upp denna fil till servern (se steg 4).

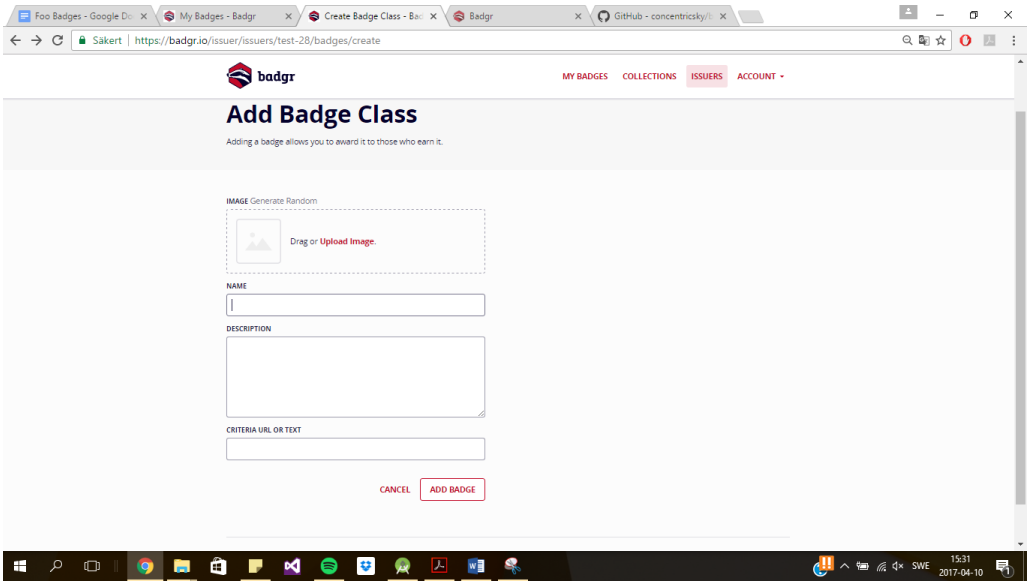
Steg 6. "Baka" badgen.

För att göra en badge mer portabel för en användare så är det möjligt att "baka" in informationen i den visuella representationen genom att använda sig av "the bakery" som finns på följande länk: http://bakery.openbadges.org/. När badgen har gått igenom "the bakery" är det möjligt att exempelvis ladda upp badgen manuellt på diverse hemsidor som tillåter uppladdning av badges. Ett exempel är Mozilla Backpack där en person kan manuellt ladda upp en badge eller att utfärdarna automatiskt skickar badgen till mottagarens backpack. Mozilla backpack är en tjänst från Mozilla som används för att visualisera och samla badges [11].

2.4.1.2 Badgr

Badgr är en webbapplikation som använder sig av Mozilla Open Badges och erbjuder verktyg för att visualisera badges samt verktyg som underlättar skapandet samt utfärdandet av badges [46]. Istället för att manuellt gå igenom alla steg som beskrivs i kapitel 2.4.1.1. utför Badgr detta med hjälp av ifyllande av information och ett knapptryck, som kan ses i figur 2.9. Badgr erbjuder även en backpack-tjänst som tillåter en användare att visualisera alla badges som en användare har fått. Badgr har även en betalningstjänst som tillåter användaren att följa en

badge-aktivitet och statistik, där exempelvis aktivitet kan vara badgens användning i sociala medier eller statistik på hur många gånger en badge har utfärdats till diverse användare. Att utfärda badges till användare är enkelt med Badgr där endast ifyllnad av användarens e-postadress räcker för att skicka badgen till användaren.



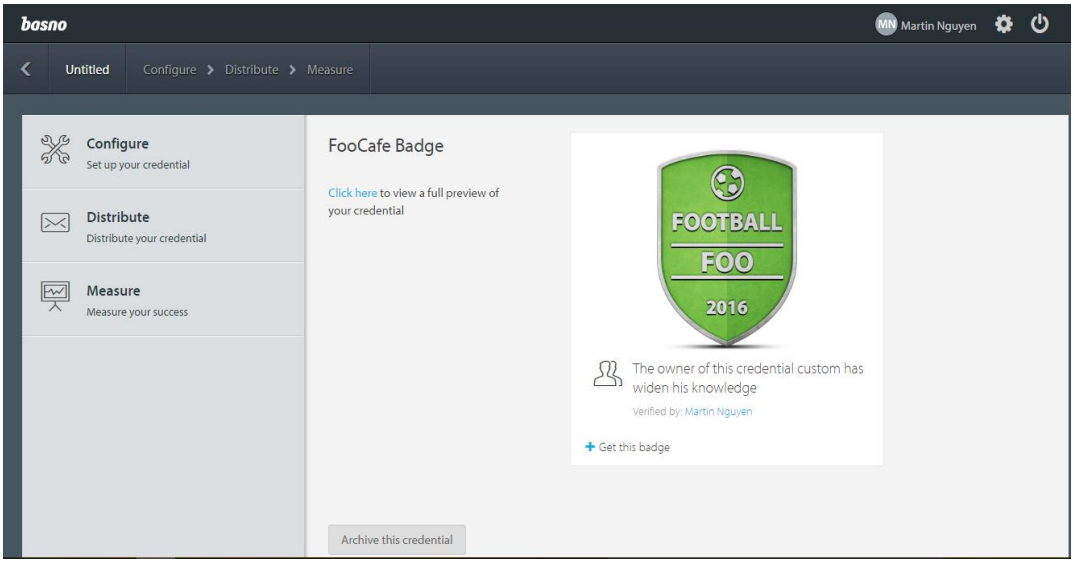
Figur 2.9. Användargränssnitt för att skapa en badge i Badgr.

Förutom att Badgr är en webbapplikation som kan användas av olika användare så finns hela webbapplikationen tillgänglig som öppen källkod på GitHub under AGPL licensen [47], detta innebär att en utvecklare kan använda sig av koden och ha Badgrs källkod som en bas för att sedan utforma det till ett personligt system med egna tillägg av funktioner. Badgr erbjuder även all sin funktionalitet via API [56], vilket innebär att med API-anrop kan funktionalitet anropas för att exempelvis att skapa badges eller utfärda badges.

2.4.2 Basno

Basno grundades år 2010 och är en digital plattform för visualisering, skapande och utfärdande av digitala badges. Aktörer som använder sig av Basno för badges är exempelvis HBO och Virgin America [13].

Basno erbjuder verktyg som automatiserar samt underlättar skapandet och utfärdandet av badges, där dessa verktyg exempelvis kan användas för att designa badgen och bestämma hur utfärdandet för badgen kommer att ske. Basno erbjuder även ett verktyg som analyserar badgens aktivitet i diverse sociala medier som exempelvis LinkedIn och Facebook [30], detta är dock en tjänst som är exklusivt tillgänglig för användare som har betalt och uppgraderat sitt konto till pro- eller premiumversion.

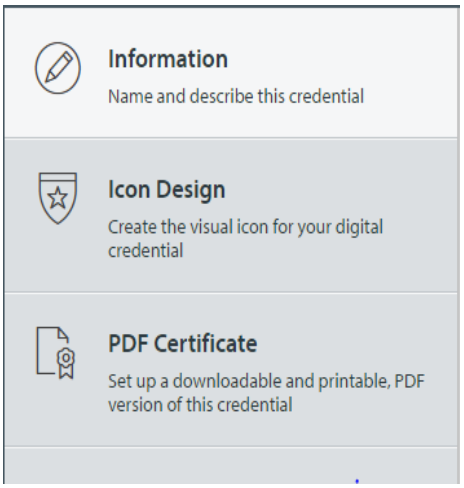


Figur 2.10. Användargränssnitt för Basno's plattform.

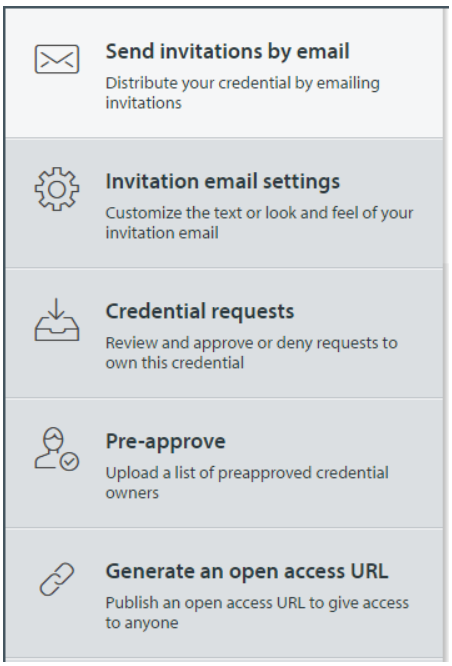
Generellt så följs följande riktlinjer för utfärdandet av en Basno badge [14]:

1. En person uppfyller ett mål eller genomför något som anses vara värdigt ett erkännande.
2. Utfärdaren av badgen skapar badgen med ett namn, beskrivning och en design, där detta utförs med Configure verktyget i figur 2.10.
3. Utfärdaren bestämmer hur utfärdandet av badgen kommer att ske, detta utförs med hjälp av de olika alternativ som finns i Distribute verktyget i figur 2.10.
4. Personerna får badgen och kan visa upp den på diverse sociala medier.

Figurerna 2.11 och 2.12 visar alternativen för verktygen som erbjuds av Basno. Verktygen är enkla att använda och kräver ingen större expertis inom området.

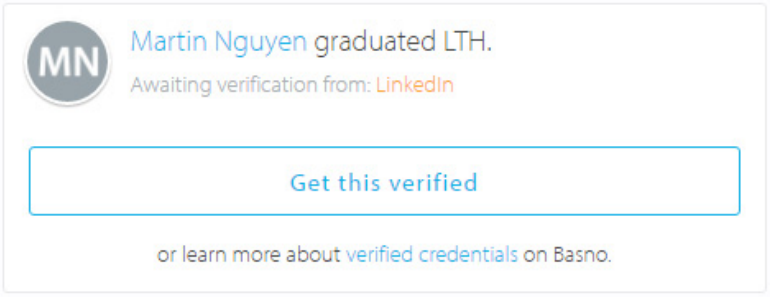


Figur 2.11. Verktyg för att skapa en badge.



Figur 2.12. Verktyg för att utfärda en badge

Som privatperson är det även möjligt att lägga till badges till sin personliga kollektion hos Basno, detta sker genom att följa ett par steg där användaren fyller i uppgifter om vad som har uppnåtts och vem utfärdarna av badgen är. Detta ska sedan verifieras av själva aktören som har utfärdat badgen för att autentiskt bekräfta badgen för privatpersonen. Figur 2.13 visar en badge som en privatperson själv har lagt till i sin kollektion, men inte har blivit verifierad av aktören där personen har fått badgen från.

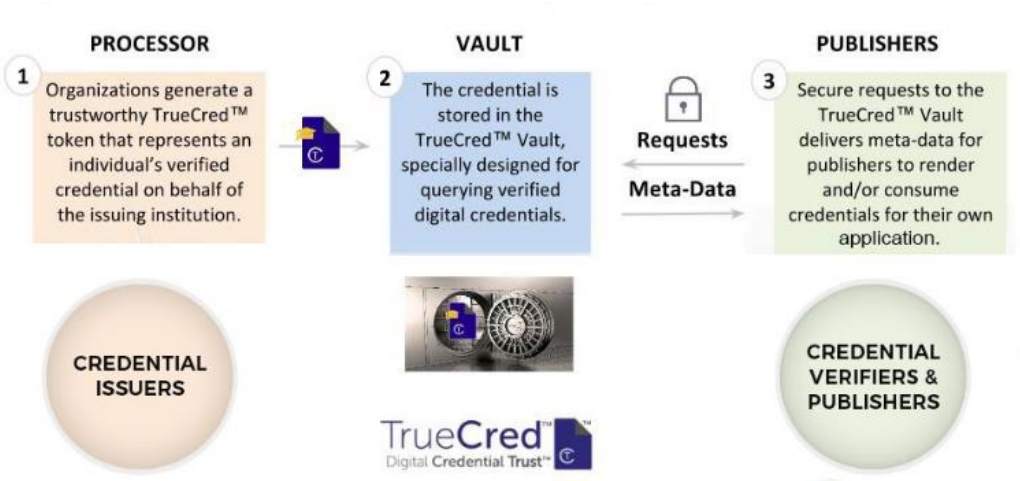


Figur 2.13. Ett exempel på en badge som väntar på verifiering av LinkedIn.

2.4.3 TrueCred

TrueCred skapades under 2012 av Accreditrust Technologies [15] och erbjuder en SaaS (Software as a service) teknologi för att skapa digitala badges som är verifierbara och skyddade från obehörig åtkomst. Detta för att skapa ett digitalt ekosystem som är baserad på tillit och säkerhet. TrueCred garanterar autenciteten för en badge genom att hantera information om vem som har utfärdat badgen och vem som äger badgen. Generella processen för att skapa en TrueCred badge sker i följande tre steg och visualiseras i figur 2.14 [31]:

1. En organisation eller företag skapar en badge som beskriver det uppnådda målet för en person.
2. Badgen placeras i TrueCred Vault, som är ett digitalt säkert lagringsutrymme för digitala badges och är utformad för att hantera förfrågningar av digitala badges från privatpersoner.
3. En person som har fått badgen kan skicka förfrågningar till TrueCred Vault för att få tillgång till badgen som sedan kan användas för att exempelvis visas upp på sociala medier.



Figur 2.14. Översiktlig vy för TrueCred's badge process.

TrueCred erbjuder verktyg som underlättar processen och de är följande:

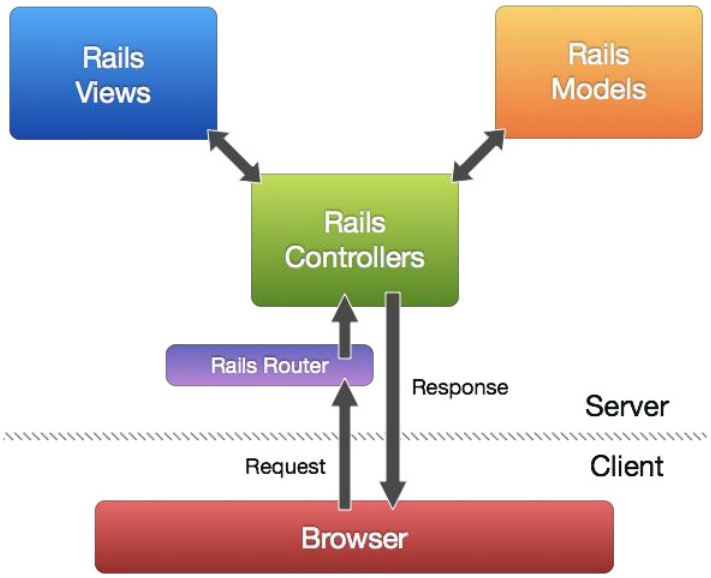
- TrueCred™ Processor: mjukvara för utfärdandet av digitala badges.
- TrueCred™ Vault: ett säkert lagringsutrymme för digitala badges.
- TrueCred™ Folio: en personlig konto service för digitala badges.
- TrueCred™ Publisher: mjukvara för verifiering och uppvisning av digitala badges.

2.5 Ruby on Rails

I detta kapitel beskrivs de komponenter som utgör Ruby on Rails.

2.5.1 Rails

Ruby on Rails, även känt som Rails, är ett ramverk som underlättar utveckling och underhåll av webbapplikationer. Rails bygger på två underliggande grundprinciper, DRY och Convention Over Configuration, som gör att kod som är skriven i Rails hålls korta och enkelt att tolka för utvecklare. DRY är en förkortning av ”Don't repeat yourself” och är en princip för mjukvaruutveckling som ska minska duplicering av kod. Convention Over Configuration innebär att Rails har en standardiserad syn på alla aspekter i att bygga en webbapplikation och innebär att om en utvecklare följer dessa conventions, så används det generellt mindre kod jämfört med en applikation i Java som använder sig av XML-konfigurering [48]. Rails underlättar även testning av applikationer då Rails automatiskt genererar test-stubbar för varje funktion som skapas. Förutom underlättandet av tester så har Rails även verktyg som förenklar versionshantering av applikationer där det är lätt att släppa en ny version av applikationen samt återgå till en föregående version. Alla Rails-applikationer är skrivna i programmeringsspråket Ruby och implementerar Model-View-Controller (MVC) arkitekturen. MVC strukturen i en Rails-applikation sker genom att Rails hanterar förfrågningar från en browser, söker därefter en kompatibel controller och utför sedan en metod som finns i controllern. Figur 2.15. visar ett exempel på hur MVC strukturen fungerar i en Rails-applikation.



Figur 2.15. MVC-struktur för applikationer i Rails.

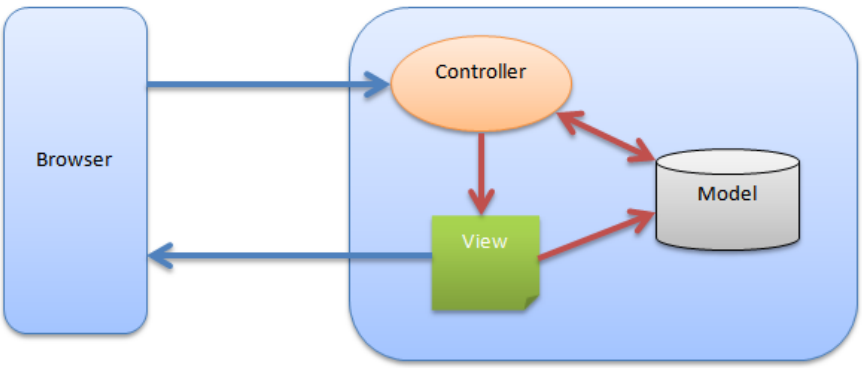
2.5.2 MVC

MVC är en designarkitektur för att utveckla interaktiva applikationer. MVC-arkitekturen innebär att en applikation delas in i tre olika komponenter, Model, View och Controller. Model är ansvarig för att hantera manipulering av data och applikationens funktioner. Model kommunicerar med exempelvis en databas för hantering av data och får instruktioner från en Controller som berättar vad för data som ska hanteras. Det som sker efter hantering av data är att Model meddelar View att användargränssnittet ska uppdateras. Ett exempel är att Model hämtar en lista av produkter som finns i en databas och skickar detta till View.

View är ansvarig för generering av ett användargränssnitt och är ansiktet utåt för applikationen. View hanterar ingen data, utan visar enbart för användaren den data som har manipulerats i Model. Ett exempel är om det finns en lista av produkter i Model så kan en View extrahera denna lista och presentera detta för användaren.

Controller är ansvarig för att hantera en användares interaktioner med applikationen och skickar dessa interaktioner till Model, där exempel på interaktioner kan vara knapptryck och input av data. Model hanterar därefter interaktionen och uppdaterar View. Ett exempel är om en användare vill se en lista av produkter och begär detta genom att trycka på en knapp i användargränssnittet.

Figur 2.16. visar ett exempel på hur MVC-arkitekturen används och hur kommunikationen mellan de olika komponenterna fungerar tillsammans.



Figur 2.16. Översikt för MVC-arkitekturen.

2.5.3 Ruby

Ruby är ett objektorienterat programmeringsspråk och kan liknas vid andra programmeringsspråk som exempelvis Java, JavaScript, PHP och Python [48]. I Ruby definieras en klass och används som en ”fabrik” för att generera objekt som sedan använder sig av metoder. En skillnad i Ruby jämfört med Java är användning av semikolon, där semikolon inte används i Ruby för att avsluta ett kodavsnitt. En annan skillnad är avsaknaden av klammerparenteser då det inte används för att markera avslutet av exempelvis en funktionskropp. Istället används nyckelordet ”end” för att markera slutet och detta kan ses i figur 2.17.

```
def say_goodnight(name)
  result = 'Good night, ' + name
  return result
end
```

Figur 2.17. Kod i Ruby med avsaknad av semikolon och klammerparenteser.

2.6 RubyMine

RubyMine är en integrerad utvecklingsmiljö (IDE) för utveckling av ramverket Ruby on Rails (RoR) som används för att utveckla olika typer av webbapplikationer. RubyMine är utvecklat av programvaruföretaget JetBrains och är en Ruby integrerad utvecklingsmiljö. JetBrains lanserade RubyMine 4.0 under år 2012 och vid tidpunkten av detta examensarbete användes RubyMine 2017.1, vilket även var den senaste versionen av RubyMine [49]. Figur 2.18. visar användargränssnittet för utvecklingsmiljön RubyMine.

Förutom Ruby stödjer RubyMine ett flertal olika programmeringsspråk för att det ska kunna vara möjligt att utveckla webbapplikationer i utvecklingsmiljön. Programmeringsspråk som RubyMine stödjer är Javascript, Coffeescript, Typescript, HTML, CSS och Less or Sass. RubyMine stödjer även ett flertal ramverk som exempelvis Node.js, Dart och AngularJS [49].

2.7 Databas

Per definition är en databas vanligtvis en samling av data som är organiserad på ett sätt som underlättar sökning samt tillgängligheten av datan. Ett Database Management System (DBMS) är mjukvara som tillåter användare och applikationer att kommunicera med databasen för att få tillgång till data samt att manipulera data. Det finns olika typer av databaser och några av dem är: [20]

- Relationsdatabaser - en databas med all data lagrade i tabeller.
- Objektorienterade databaser - innebär att databasfunktionalitet har lagts till som persistens för ett objektorienterat språk som exempelvis C++ eller Java.
- Objekt-relationella databaser - innebär att en databashanterare har expanderats genom tillägg av en objektorienterad datamodell med klasser, arv och metoder.

2.7.1 PostgreSQL

PostgreSQL är ett objekt-relationell databashanterare (DBMS) som släpptes 1996 och har kontinuerligt utvecklats i över 20 år. PostgreSQL kan köras på de flesta operativsystem och detta inkluderar de mer kända operativsystemen Linux , UNIX och Windows. PostgreSQL stödjer lagring av stora binära objekt som exempelvis bilder, ljudfiler och video. Förutom att PostgreSQL stödjer stora delar av SQL-standarden så erbjuder systemet även funktioner som exempelvis transaktioner, asynkrona procedurer och multiversion concurrency control (MVCC) [16].

Grunden till PostgreSQL lades redan under 80-talet och var en uppföljning av projektet Ingres ett projekt som gick ut på skapa en databas i enighet med den teoretiska modellen av ett relationsdatabassystem. Projektet som utfördes efter Ingres döptes till Postgres, Postgres utvecklades mellan 1986-1994 och var ett projekt som utforskade konceptet objekt-relation som då skulle vara nästa innovativa steg inom databasteknik. År 1995 utvecklades systemet för att stödja SQL och bytte även då namn till Postgres95. Under året 1996 bröt Postgres95 sig ur de akademiska gränserna och introducerades till världen av öppen källkod där olika utvecklare kunde bidra med vidareutveckling av systemet och var även då som systemet tog upp namnet PostgreSQL [17].

PostgreSQL är av öppen källkod vilket betyder att den är tillgänglig för alla och gör det möjligt att ändra på systemet för att uppfylla ett företags- eller personliga behov, vilket anses vara en av många fördelar jämfört med andra databashanterare (DBMS) [18].

2.8 Beacon

En beacon är en liten radiosändare som använder sig av standarden och kommunikationstekniken Bluetooth för att skicka ut signaler till olika enheter. Beacon fungerar på precis samma sätt som en fyr som skickar signaler till fartyg och båtar för varning för grund eller som positioneringshjälp. En beacon skickar dock radiosignaler till bland annat mobiltelefoner och surfplattor för att exempelvis väcka en applikation som finns installerad på enheten som tar emot dessa signaler. Samtliga enheter som har Bluetooth påkopplat och som finns i närheten av en beacon som skickar ut liknande signaler kommer att fånga upp dessa signaler, precis på samma sätt som fartyg och båtar kommer att se signalerna från en fyr [36].

Beacons är små enheter som består endast av en processor, en radio och batterier. En beacon kan dock anslutas med en USB-kabel för strömförsörjning istället för användning av batterier. En del beacons kan innehålla exempelvis en accelerometer och olika typer av sensorer [36].

Det som skickas i signalen som sänds ut av en beacon är ett unikt nummer (ID) som samtliga beacons identifieras med. En enhet som fångar upp en signal sänt från en beacon med exempelvis ID nummer ”22” vet därmed precis vilken beacon som skickat signalen. ID numret som skickas leder dock inte till någon särskild aktivitet i enheten som fångar upp signalen. För att en aktivitet ska ske i enheten måste det programmeras in i applikationen. I en applikation kan man exempelvis programmera att denna applikation ska väckas upp då enheten som applikationen är installerad på fångar upp en signal med ID nummer ”22” [36].

Om en butik exempelvis installerat två olika beacons i sin butik kan ägaren av dessa beacons bestämma vad en enhet med butikens applikation installerad på enheten ska utföra vid mottagandet av signalerna. Detta programmeras som tidigare nämnt i applikationen eller programmet som ska användas av en användare. Om en enhet fångar upp signaler av beacon med ID nummer ”1” kan ägaren exempelvis vilja att en rabattkupong visas i applikationen. Om enheten däremot fångar upp signaler av beacon med ID nummer ”2” kan applikationen exempelvis erbjuda navigationshjälp i butiken [36].

Ett annat namn på beacons är “BLE beacons”, där BLE står för Bluetooth Low Energy och är den teknik som används av beacons idag. BLE är en energisnål version av Bluetooth som utvecklades år 2010. Detta beror på att en beacon sänder vanligtvis ut flera signaler per minut till enheter och behöver därmed använda sig av en energisnål teknik som BLE för att inte behöva byta ut batteri varje månad. Med hjälp av BLE kan en beacon klara sig i flera år på små knappcellsbatterier [36].

2.8.1 Eddystone

Eddystone är ett beaconformat som utvecklats av Google i juli 2015 och kan användas av en beacon för att kommunicera med bland annat Android- och iOS-enheter. iBeacon är ett liknande format utvecklat av Apple som olika sorters beacons kan använda sig av. Eddystone fungerar på så sätt att den använder sig av olika payloads för att få en beacon att utföra olika uppgifter. Eddystone-UID är en payload med ett unikt ID och används för att identifiera en beacon. Eddystone-UID består av 16 bytes och är indelat i två olika delar. Den första delen kallas för namespace och består av 10 bytes och kan användas för att filtrera bort andra beacons från en särskild grupp av beacons som ägs av en person. Denna person kan därmed identifiera sina samtliga beacons genom att tilldela dem identiska namespace. Ett exempel på ett namespace som används som default av ett företag som utvecklar beacons är ”EDD1EBEAC04E5DEFA017”. Den andra delen kallas för instance och består av 6 bytes och kan användas för att skilja mellan sina egna beacons [12].

Eddystone-URL är en payload som består av en URL, där längden av denna URL inte måste vara en specifik längd och spelar mindre roll i jämförelse med Eddystone-UID. Eddystone-URL fungerar på så sätt att en beacon som använder sig av EddyStone-URL kan exempelvis användas vid en biograf för att sända ut länkar (URL) för olika trailers för filmer på bio till Android- och iOS-enheter. För att det ska vara möjligt att öppna en Eddystone-URL måste en fysisk webbläsare vara installerad på enheten som används. Ett exempel på en Eddystone-URL är “https://www.youtube.com/” [12].

Eddystone-TLM är en payload som kan användas för att exempelvis skicka ut signaler till Android- och iOS-enheter att batteriet snart är slut och behöver därmed laddas eller bytas ut mot nya batterier. Eddystone-TLM kan även användas till att skicka ut signaler om temperaturen för en beacon, hur många signaler som har skickats senast en beacon slagits på och hur lång tid en beacon har varit påslagen [12].

Eddystone-EID är en payload som består av ett Ephemeral ID och fungerar på så sätt att den skyddar en beacon mot två olika sorters attacker. Den första sortens attack kallas för Hijacking eller Piggybacking och utförs på så sätt att en tredjepartsprogram använder sig av infrastrukturen för att skicka sina egna signaler till Android- och iOS-enheter och får det att se ut som att dessa signaler skickas från någon specifik beacon, trots att denna beacon skickar helt annorlunda signaler till enheterna. Användaren av en enhet får därmed två helt olika signaler och tror troligtvis att dessa två signaler kommer från samma beacon trots att ena signalen inte alls kommer från en beacon [12].

Den andra attacken kallas för Spoofing och utförs på så sätt att en tredje part kopierar signalerna från en beacon och skickar dessa signaler på en annan geografisk plats. Om signalerna från en beacon som finns i en butik kopieras kan tredje parten skicka ut dessa signaler i exempelvis en buss till enheterna trots att det inte finns en beacon i bussen [12].

2.8.2 U-blox

U-blox är ett schweiziskt företag som grundades år 1997 och tillverkar chipp och moduler för trådlös kommunikation och positionering. Bluetooth och Wi-Fi är bland annat de tekniker som U-blox använder sig av för utvecklingen av chipp och moduler. Några exempel på chipp och moduler som utvecklas är olika sorters beacons och GPS-moduler.

Ett flertal beacons som utvecklats av företaget U-blox användes vid utvecklingen av mobilapplikationen. De beacons som användes vid utvecklingen är från NINA-B1 serien och är små BLE beacons som stödjer Bluetooth v4.2. De kan dock uppdateras för att kunna stödja Bluetooth v5.0. Räckvidden utomhus för signaler från en beacon från NINA-B1 serien är ca 300-350 m och räckvidden inomhus är ca 30-40 m, vilket dock kan påverkas om andra enheter använder sig av samma radiofrekvenser eller om metallföremål ligger i vägen och blockerar radiofrekvenserna. NINA-B1 beacons stödjer bland annat formatet Eddystone som användes vid utvecklingen av mobilapplikationen [24].

2.9 Positionering

Positionering syftar på tekniken att lokalisera ett mål för att sedan använda sig av denna information för olika tjänster. Termer som är essentiella för att förstå positionering är position och positioneringssystem. Position avser de koordinater som målet befinner sig på och positioneringssystem är ett system som används för att lokalisera positionen. Informationen som fås av positioneringssystem kan sedan användas för exempelvis nödsituationer eller navigering. Positioneringssystem består av exempelvis GPS som använder sig av satelliter, nätverksbaserad teknologi som är Wi-Fi och mobilbaserad teknologi som är nätverk från mobilstationer [23].

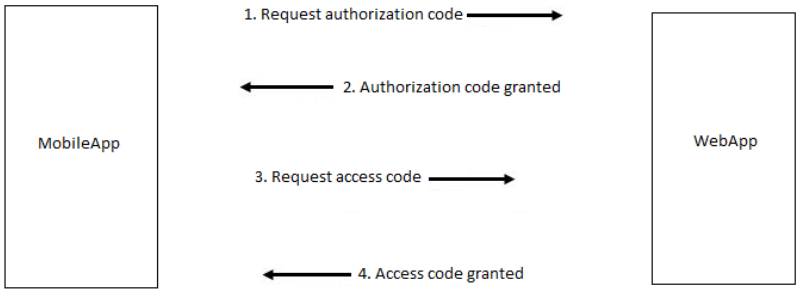
Google maps är ett exempel på en tjänst som använder sig av positionering för att bestämma en användares position och kan exempelvis hjälpa användaren med navigering mot en destination.

2.10 OAuth 2

OAuth 2 är ett ramverk för autentisering som tillåter applikationer få tillgång till användarkonton hos webbtjänster som exempelvis Facebook eller Github. OAuth 2 fungerar på ett sätt att en utomstående applikation som exempelvis en mobilapplikation, delegerar autentiseringen till en tjänst som hanterar användarkonton. Autentiseringen genomförs sedan genom att en inloggning sker och vid lyckad inloggning skickas en auktoriseringskod tillbaka. Auktoriseringskoden används sedan för att få en accesskod från den tjänst som hanterar användarkonton [50]. Tjänster som erbjuder OAuth 2 autentisering är exempelvis Doorkeeper [51] och Spring Security OAuth [52].

Ett exempel på autentisering med OAuth 2 är om en mobilapplikation vill ha tillgång till ett konto så är det möjligt för mobilapplikationen att delegera autentiseringen till en webbapplikation som har ett inloggningssystem, vilket innebär att mobilapplikationen inte hanterar känsliga uppgifter

som exempelvis lösenord och samtidigt får tillgång till kontot. Översiktlig vy för autentiseringen visualiseras i figur 2.18.



Figur 2.18. Översiktligt flöde för OAuth 2 autentisering.

1. Mobilapplikationen delegerar autentisering till webbapplikationen.
2. Inloggning utförs på webbapplikationen, där en auktoriseringskod skickas tillbaka vid en lyckad inloggning.
3. Auktoriseringskoden används sedan för att få en accesskod.
4. Accesskoden används sedan för kommunikation mellan mobilapplikationen och webbapplikationen. Accesskoden innehåller vanligtvis information om användaren som har utfört inloggningen.

2.11 Spelifiering (Gamification)

Spelifiering, eller gamification är ett koncept som använder sig av spelmekanismer i applikationer och verksamheter som inte nödvändigtvis behöver ha någon koppling till spelande. Grundidén bakom spelifiering är därmed att öka engagemang och intresse för applikationer och produkter som inte använder sig av spelmekanismer. Detta utförs därmed för att motivera användarna att fortsätta använda dessa applikationer och produkter [21].

Spelifiering kan användas på flera olika sätt och appliceras inom ett flertal verksamheter som exempelvis handel, IT, transport och utbildning. Spelifiering har börjat användas mer flitigt i dagens produkter och kan hittas i Hondas hybriddrivna bil, Insight [35]. Honda har installerat ett belöningssystem åt sina förare där det visas blommor vid hastighetsmätaren om föraren kör miljövänligt. Antalet blommor som visas beror på hur miljövänligt en förare kör och kan därmed motivera föraren till att köra miljövänligt. Blommorna som visas vid hastighetsmätaren kan även i och med det påverka och utbilda föraren i hur man man bäst använder bilen.

Företaget Nissan använder sig av liknande koncept och har spelifierat sin bil Leaf genom att de kopplat upp bilen mot internet och skapat en hemsida som förarna kan använda sig av för att

läsa information om exempelvis förarens bränsleförbrukning [33]. På hemsidan har en förare möjlighet att jämföra sin bränsleförbrukning med andra förare av liknande bil och därmed tävla med andra förare. Figur 2.19. visar exempel på statistik för Nissan Leaf.

Figur 2.19. Exempel på Nissan Leaf statistik

För att kunna lyckas med spelifieringen av applikationer och produkter måste man utsätta användaren för någon typ av utmaning som motiverar användaren att fortsätta använda produkten. Nycklarna till det är att utmaningen måste leda till att användaren får någon sorts belöning för det användaren utför och att det är meningsfullt att utföra utmaningen. Tre olika frågor kan användas för att säkerställa att spelifiering av en produkt lyckas. Den första frågan är en fråga om utmaning och vad användaren ska uppnå med sin aktivitet. Den andra frågan är en fråga om belöning och vad användaren får när den klarat en utmaning. Den tredje och sista frågan är en fråga om meningsfulla framsteg och varför man ska fortsätta samla på sig belöningar. Finns svar till dessa tre frågor kommer spelifiering av en applikation eller en produkt troligtvis lyckas och motivera användarna att använda det och inte tröttna på det [21].

Ett exempel på ett spel som besvarar dessa tre frågor tydligt och klart är spelet Tetris [34]. Utmaningen i spelet går ut på att försöka skapa kompletta rader av blocken som en användare automatiskt får under spelets gång. Belöningen för att en användare skapat en komplett rad är att raden tas bort och att poäng delas ut för det. Meningsfulla framsteg sker genom att spelet blir snabbare ju mer tid det går, vilket leder till att mer poäng delas ut. Poängen som samlas ihop kan därmed användas till att tävla mot andra användare.

För att kunna spelifiera en applikation eller en produkt krävs det att det används en eller flera spelmekanismer. Det finns ett flertal olika spelmekanismer som kan användas vid spelifiering som exempelvis poängsystem. Ett poängsystem ger användaren möjlighet till att se sina progressiva framsteg och kan anses vara en belöning för exempelvis avklarade utmaningar. En annan populär spelmekanism är levels (nivåer) och fungerar på så sätt att de visar likt ett poängsystem progressiva framsteg. Badges är även det en spelmekanism som kan användas vid spelifiering som fungerar på så sätt att en person som klarar av diverse utmaningar får badges som belöning. En badge är ett verifierbart och portabelt insignium som intygar att en

person utfört en uppgift eller visat en förmåga som exempelvis ha avklarat en kurs eller deltagit i ett särskilt event [21].

2.12 Liknande applikationer

Liknande applikationer som använder sig av spelifiering för att uppmuntra användarna till att använda deras applikation är exempelvis Untappd och Swarm.

2.12.1 Untappd

Untappd är en mobilapplikation som inriktar sig på användarnas konsumtion av öl. Untappd använder sig av konceptet spelifiering genom att belöna användarna med badges för exempelvis att ha konsumerat ett specifikt öl eller genomfört en utmaning där exempelvis fyra olika sorters öl ska konsumeras. Applikationen används för nöjes skull där det inte finns någon kontroll för att användaren faktiskt har konsumerat ölen och därför är det möjligt att få alla badges utan att ha konsumerat öl. Untappd har även funktioner som exempelvis ett nyhetsflöde där en användare kan dela med sig av sin aktivitet och visa upp sina badges [3].

2.12.2 Swarm

Swarm är en mobilapplikation som tillåter användarna att utföra incheckningar på olika platser som de besöker, ett exempel är om en användare besöker Lunds universitet så kan användaren använda sig av applikationen för att lagra information om platsen där incheckningen har utförts. Swarm använder sig även av konceptet spelifiering genom att belöna användarna med coins som i detta fall är en form av poäng som används i syfte att bestämma placeringarna i en leaderboard. Utöver poängsystemet erbjuder applikationen även upplåsningsbara märken för olika uppfyllda krav gällande incheckningar, där dessa märken sedan kan användas för sociala funktioner som exempelvis för att kommentera en väns incheckning. Det finns dock ingen kontroll av att en användare har varit på platsen där incheckningen har utförts, vilket gör det enkelt att lura systemet och låsa upp samtliga märken samt manipulera en leaderboard till en personlig fördel [19].

2.13 Övrigt

Beskrivning av verktyg som användes för att underlätta versionshantering av koden och dokumentering examensarbetet.

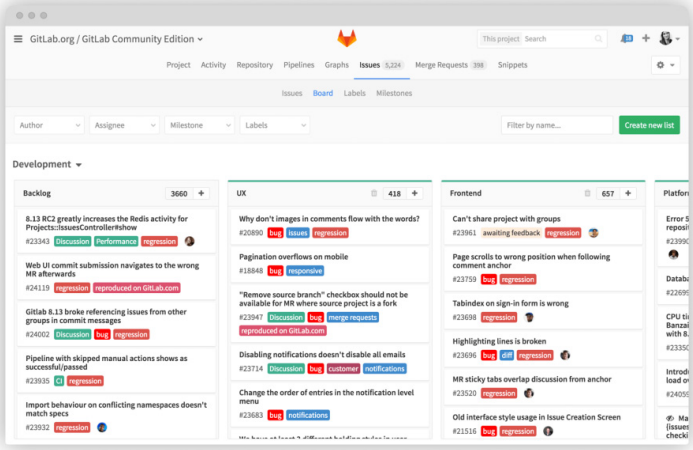
2.13.1 Git

Git är ett versionshanteringsprogram [22] som skapades år 2005 av programmeraren Linus Torvalds, som även skapat det Unix-liknande operativsystemet Linux. Git skapades av Torvalds för att kunna hantera källkoden till Linuxkärnan som används i operativsystemet Linux. Git fungerar på så sätt att den spårar ändringar i datafiler. Vid varje ändring av en fil sparas ändringarna och samtliga versioner av filen. Om exempelvis Alice och Bob delar på ett dokument och Alice redigerar i dokumentet kommer både Alice och Bob få en ny version av dokumentet med den nya redigeringen och precis vad som redigerats i dokumentet. Det gamla dokumentet som inte innehåller Alice redigering sparas också.

För att ge Alice och Bob möjligheten att dela på ett dokument och därmed få Git att fungera har Torvalds skapat olika kommandon som en användare kan utföra, bland annat push och pull. Push fungerar på så sätt att när exempelvis Alice redigerar sitt lokala dokument och använder kommandot push skickas allt som redigerats av Alice till det delade dokumentet som både Alice och Bob delar på. Bob kan dock inte se Alice redigering av dokumentet i sitt lokala dokument. Det vill säga att Bobs lokala dokument är inte uppdaterad och är inte identisk med det delade dokumentet. För att Bobs lokala dokument ska bli identisk med det delade dokumentet kan Bob använda sig av kommandot pull. Detta kommando fungerar på så sätt att Bobs lokala dokument uppdateras och blir identisk med det delade dokumentet [22].

2.13.2 GitLab

GitLab är en webbaserad Git-kodhanterare som bidrar med versionshantering likt Github. GitLab bidrar med olika funktioner som exempelvis versionshantering, fördelning av olika roller och en KanBan-liknande dashboard för planering samt genomförande. Versionshantering används för att underlätta kontroll och spårande av förändringar av koden under projektets gång. Fördelning av rollerna används för att bestämma projektmedlemmarnas auktoritet i projektet där exempelvis en medlem som har rollen ”guest” inte har samma befogenheter som en medlem som har rollen ”developer”. Dashboarden som erbjuds av GitLab kan jämföras med ett Trello-board [32] som underlättar planeringen och utförandet av olika moment i projektet, ett exempel på dashboarden från GitLab kan ses i figur 2.20.



Figur 2.20. Dashboard från GitLab.

2.13.3 Google Drive

Google Drive är ett filhanteringssystem som är skapat och underhålls av Google. Google Drive tillåter användarna att ladda upp sina filer och hantera sina filer i en molnbaserad miljö, där detta innebär att filernas tillgänglighet är stor. Google Docs som är en textredigerare är inbyggt i Google Drive och bidrar till att olika personer kan arbeta med samma dokument i realtid. Google Slides samt Google Sheets är även inbyggt och kan användas för att arbeta tillsammans på presentationer eller olika tabeller.

Kapitel 3

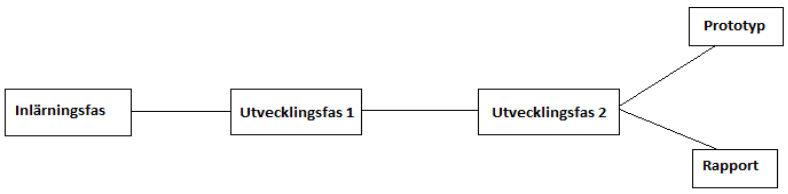
3 Metod och Analys

I detta kapitel beskrivs metoder som har använts för att utföra examensarbetet och motivering för de olika val som gjordes gällande metod samt teknik. En beskrivning av arbetsprocessen för mobilapplikationen och webbapplikationen finns i detta kapitel. Projektmodell, intervjumetodik och källkritik behandlas också.

3.1 Arbetsprocess

Arbetsprocessen bestod av tre faser, inlärningsfas, utvecklingsfas 1 och utvecklingsfas 2. Utvecklingsfas 1 är fasen där utveckling av mobilapplikationen påbörjades. Utvecklingsfas 2 är slutfasen för prototypen och involverade implementering av funktionalitet för webbapplikationen samt mobilapplikationen. Figur 3.1. visar en översiktlig vy över arbetsprocessen för examensarbetet som resulterade i en prototyp som omfattar mobilapplikationen och webbapplikationen.

Arbetet har utförts på Foo Cafés kontor där examensarbetarna tillbringade minst fem timmar dagligen för ett effektivt samarbete mellan alla deltagande parter. Detta underlättade även all kommunikation med handledarna och värdefull feedback kunde kontinuerligt fås. Utöver kontorstiden spenderades även timmar på att arbeta med examensarbetet hemifrån och ett antal timmar användes även till att delta i olika event hos Foo Café för att utföra intervjuerna. För att underlätta arbetet med examensrapporten och diverse dokument användes Google Drive samtidigt som kontinuerlig backup av rapporten sparades lokalt och i en molnmiljö som exempelvis Dropbox [37]. GitLab användes för versionshantering av källkoden och bidrog till att källkoden var tillgänglig för alla deltagande parter samt kunde minimera risken att arbeta med fel version. Ett dokument i Google Drive användes som en loggbok för att dokumentera den vardagliga aktiviteten.



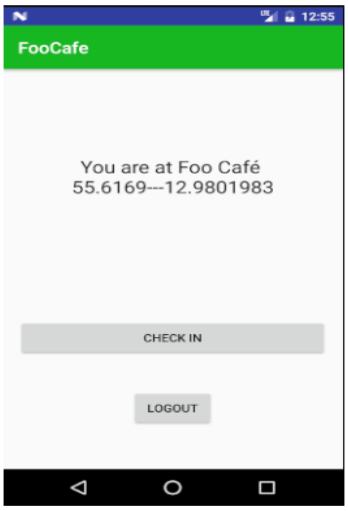
Figur 3.1.Översiktlig vy för arbetsprocessen

3.1.1 Inlärningsfas

Inlärningsfasen var en viktig fas i examensarbetet då den lade grunden till utvecklandet av mobilapplikationen och bidrog till insamling av information om de olika tekniker och verktyg som användes för examensarbetet. Då utveckling i Android Studio främst sker i programmeringsspråket Java, lades inte mycket tid på själva inlärningen av Java vilket beror på att examensarbetarna var trygga i sin förmåga att använda programmeringsspråket Java. Stora delar av inlärningen bestod främst av att bekanta sig med den nya utvecklingsmiljön som var Android Studio. Inlärningen skedde genom kontinuerlig bearbetning av information från exempelvis Youtube, Android Studio dokumentationer och öppen källkod från olika applikationer som tidigare har utvecklats i Android Studio. Inlärningen pågick kontinuerligt under hela examensarbetets gång, men denna grundliga inlärningsfas pågick under en period på två veckor varefter utvecklingsfas 1 påbörjades.

3.1.2 Utvecklingsfas 1

Detta var en fas som främst bestod av att skapa en grundläggande bas för mobilapplikationen och utveckla funktioner som eventuellt skulle användas till mobilapplikationen. Syftet med att skapa en grundläggande bas var att utforska utvecklingsmiljön Android Studio och försöka få förståelse för hur utvecklingsmiljön fungerar. Ett exempel på detta är hur relationen mellan en design skriven i XML och en kodfil skriven i Java fungerar. Basen för mobilapplikationen involverade en enkel design för mobilapplikationen som gav examensarbetarna välbehövlig kunskap för fortsatt utveckling av mobilapplikationen. Figur 3.2. visar en enkel design av mobilapplikationen som skapades under utvecklingsfas 1.



Figur 3.2. En enkel design med två knappar och text som visar koordinater.

3.1.3 Utvecklingsfas 2

Denna fas är slutfasen av examensarbetet och bestod av utveckling av mobilapplikationen och implementering av funktionalitet till Foo Cafés webbapplikation. Utvecklingen av mobilapplikationen gjordes med tillägg av funktionalitet och design som beskrivs i kapitel 4. Tillägg av funktionalitet underlättades då många funktioner redan utvecklades under utvecklingsfas 1 och detta innebar att dessa funktioner kopplades ihop med tillhörande design och resulterade i en fungerande mobilapplikation under utvecklingsfas 2.

Under denna fas introducerades examensarbetarna till Foo Cafés webbapplikation som är skriven i programmeringsspråket Ruby on Rails och en inlärning av utvecklingsmiljön RubyMine samt förståelse för den redan existerande koden genomfördes. Integrering av badge-systemet gjordes under denna fas. Därefter implementerades funktionalitet på Foo Cafés webbapplikation som kopplade samman badges till incheckningarna som görs på mobilapplikationen. Funktionalitet för att skapa badges på webbapplikationen och ett inloggningssystem för webbapplikationen skapades även. Utvecklingsfas 2 resulterade i en fungerande mobilapplikation som tillsammans med webbapplikationen utgör FooBadges.

3.2 Projektmodell

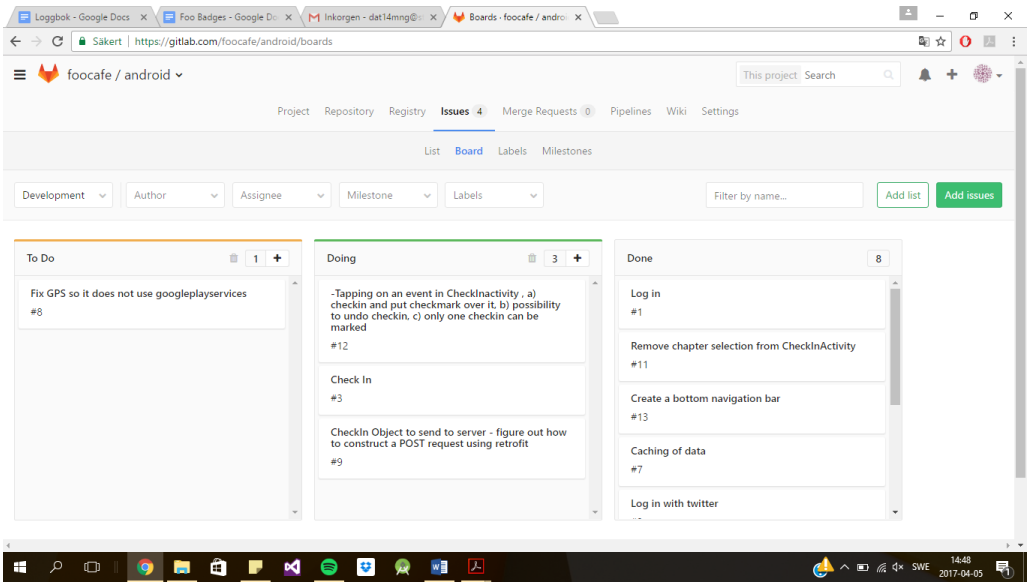
Att arbeta efter en agil projektmodell bestämdes väldigt tidigt i examensarbetet tillsammans med handledarna hos Foo Café. Ett exempel på agila projektmodeller är Kanban och Scrum [38]. Dessa agila projektmodeller bygger på manifestet för agil utveckling som är riktlinjer och principer som bör följas och uppfyllas för att anses vara agil utveckling. Manifestet kan summeras med följande fyra grundpelare [39]:

- **Individer och interaktioner** framför processer och verktyg.
- **Fungerande programvara** framför omfattande dokumentation.
- **Kundsamarbete** framför kontraktsförhandling.
- **Anpassning till förändring** framför att följa en plan.

Scrum baserar arbetsflödet på iterationer, även kallad sprintar, med regelbundna leveranser av produkten i slutet av varje iteration [38]. Dessa iterationer pågår vanligtvis i perioder av 2-4 veckor. Ett projekt som ska utföras enligt Scrum använder sig av en backlogg där arbetet förs in. Arbetet visualiseras med hjälp av en Scrum-board som har olika kolumner där det beskrivs vilket stadium som arbetet befinner sig i, där dessa olika stadier är exempelvis ToDo, OnGoing och Done. Det finns dock möjlighet att ha fler kolumner som kan detaljerat beskriva arbetet. Före en iteration bestäms vilka uppgifter från backloggen som ska genomföras och dessa uppgifter placeras på Scrum-boarden. Under en pågående iteration är det inte möjligt att lägga till uppgifter till Scrum-boarden, vilket innebär att inga andra uppgifter än de som finns på Scrum-boarden kan genomföras, vilket begränsar arbetet som utförs under varje iteration. En iteration avslutas sedan med ett möte där utvärdering av iterationen sker i syfte att förbättra och förbereda kommande iterationer.

KanBan använder sig inte av iterationer vid utförandet av arbetet. KanBan begränsar istället arbetet som kan utföras under ett givet stadium. Detta görs med hjälp av en KanBan-board som även den har kolumner som beskriver vilket stadium arbetet befinner sig i. Dessa stadier kan exempelvis vara ToDo, OnGoing och Done, där det dock är möjligt att ha flera kolumner som kan detaljerat beskriva arbetet. Kolumnerna har ett begränsat utrymme som används som en begränsning för antalet uppgifter som kan pågå i ett stadium. Detta innebär att nya uppgifter inte kan dras in i produktion förrän pågående uppgifter har slutförts och frigjort utrymme i en kolumn. Detta leder till att KanBan är ett Pull-system jämfört med Scrum som kan liknas vid ett Push-system. Ett utmärkande drag i KanBan är mätning av ledtid, vilket innebär mätning av tiden det tar för att slutföra en arbetsuppgift. Därefter optimeras arbetsprocessen så att ledtiden blir så kort som möjligt [38].

Projektmodellen som användes för examensarbetet har sin grund i KanBan men är inte strikt KanBan och har inslag av Scrum. Utvecklingsfasen av examensarbetet har utförts enligt KanBan där uppgifter har dragits in i utvecklingsfasen när pågående uppgifter har slutförts. Scrum hade förmodligen begränsat processen då det inte är möjligt att utföra andra uppgifter än de som har antagits under en pågående iteration, vilket innebär att om en uppgift slutförs väldigt snabbt så är det inte möjligt att påbörja en ny uppgift förrän iterationen är över. KanBan leder till en mer flexibel arbetsprocess gällande utvecklingen av mobilapplikationen och webbapplikationen. Ledtiden som har nämnts som ett utmärkande drag för KanBan har inte använts då syftet med att mäta ledtiden är att optimera processen för att minska ledtiden för kommande uppgifter. Examensarbetarna ansåg att detta inte var nödvändigt för examensarbetet då ledtiden kunde variera kraftigt beroende på uppgiften som skulle genomföras. Visualiseringen av utvecklingsfasen gjordes med hjälp av en GitLab-board (se kapitel 2.9.2) som påminner om en KanBan-board och kan ses i figur 3.3. Scrums regelbundna leveranser ansågs vara ett verktyg som kunde tillämpas under examensarbetet, detta för att få regelbundet feedback så att Foo Café får prototypen som de har efterfrågat. Dessa leveranser användes som kontrollpunkter för att informellt visa upp för Foo Café vad som har uppnåtts efter varje slutförd uppgift.



Figur 3.3. GitLab-board som användes under examensarbetet.

3.3 Intervjuer

Det utfördes tio intervjuer under examensarbetet med olika personer som hade en bakgrund inom IT, där personerna hade yrkesroller inom exempelvis design, utveckling och säkerhet. Att det skulle utföras intervjuer med olika deltagare på Foo Cafés event bestämdes tidigt i examensarbetet eftersom det är viktigt med återkoppling från framtida användare av mobilapplikationen. Ett enkelt sätt att få återkoppling från en person och få information och åsikter från personen om exempelvis en prototyp är genom att visa prototypen och ställa frågor om prototypen. Enkelheten och effektiviteten av att utföra en intervju var de stora anledningarna till att det valdes att utföras intervjuer.

En väl genomförd intervju måste uppfylla tre olika krav på användbarhet. Det första kravet är att intervjumetoden måste ge tillförlitliga svar. Det andra kravet är att svaren som ges av personerna som intervjuas måste vara giltiga. Ett tillförlitligt och giltigt svar anses vara ett svar som kan användas, vilket betyder att svaret exempelvis kan användas för att att dra en slutsats i en vetenskaplig rapport eller vara grund för beslut om anställning till ett jobb i ett företag. Det tredje kravet är att andra personer ska kritiskt kunna granska slutsatserna av intervjuerna [53].

För att få ett tillförlitligt och giltigt svar från intervjupersonen krävs det av intervjuaren att få ut data av intervjun som speglar källan. Intervjuaren ska försöka fånga intervjupersonens uppfattningar och åsikter om prototypen och inte försöka bekräfta sin egen tankemodell och därmed försöka vinkla intervjupersonens svar till sin egen nytta. Svar som samlats in från intervjupersonerna har därmed analyserats noga och inte vinklats för bekräftelse av egen tankemodell [53].

En intervju kan utföras på flera olika sätt och det finns ett flertal olika sorters intervjumetoder. Olika former av intervjuer kan delas in och utgå från “skillnader i grad och struktur”. Intervjuer delas därmed vanligtvis in i fyra olika typer, där den första typen kallas för öppen intervju. En öppen intervju är en sorts intervju där intervjuaren försöker minska sitt inflytande på intervjupersonens svar till frågorna. Intervjuaren försöker därmed inte leda och rikta intervjupersonen på något vis. Intervjuaren frågar öppna och ostrukturerade frågor där intervjupersonen får svara fritt och på vilket sätt den personen vill göra. Varje intervjuperson kommer troligtvis ge olika svar till frågorna och därmed kommer intervjuerna bli helt olika [54].

Den andra typen av intervju kallas för riktat öppen intervju och går till på så sätt att intervjuaren använder sig av exempelvis en teori eller modell som följs under intervjun. En riktat öppen intervju är annars rätt lik en öppen intervju där det inte finns stora skillnader på hur intervjun utförs och vilka typer av frågor som ställs till intervjupersonen [54].

Den tredje typen av intervju kallas för halvstrukturerad intervju och fungerar på så sätt att en intervjuare använder sig av både öppna och slutna frågor. Till skillnad från en öppen intervju och en riktat öppen intervju används det förberedda och bestämda frågor och där följdfrågor kan möjligtvis ställas utifrån vad intervjupersonen svarar på frågorna. Intervjuaren utgår därmed efter en intervjuplan och följer denna plan [54].

Den fjärde typen av intervju kallas för strukturerad intervju och fungerar på så sätt att intervjuaren använder sig endast av slutna frågor. Följdfrågor av intervjuaren kan inte ställas till intervjupersonen i en strukturerad intervju och frågorna som ska ställas till intervjupersonen är redan skrivna i förväg. En strukturerad intervju har fasta svarsalternativ, vilket leder till att det är lika för alla intervjupersoner [54].

3.3.1 Intervjumetodik

Typen av intervju som valdes att utföras under detta examensarbete var halvstrukturerade intervjuer. Denna typ av intervju valdes eftersom intervjuaren kan både ställa öppna och slutna frågor och där följdfrågor givetvis kan ställas till intervjupersonerna utifrån vad de svarar på frågorna. Personer som har intervjuats har valts på de olika event som Foo Café hade anordnat och bestod av personer som var intresserade av att ställa upp på en intervju.

Upplägget för intervjuerna har utförts enligt bilaga 1, där intervjun börjar med en introduktion till FooBadges och en demonstration av mobilapplikationen. Intervjun utfördes sedan där alla frågor enligt bilaga 1 ställdes och de olika svaren dokumenterades med hjälp av anteckningar som utfördes under intervjun. Övriga diskussioner kring FooBadges och mobilapplikationen som inte täcktes av intervjufrågorna dokumenterades även och användes för att komplettera deltagarnas olika synpunkter. Svaren sammanställdes i kapitel 3.3.2 och slutsatser från sammanställningen presenteras i kapitel 3.3.3.

3.3.2 Sammanställning av intervjuer

Här presenteras en sammanställning av svaren från intervjupersonerna.

1. Vilka anledningar finns det för att en deltagare inte kommer till ett event de anmält sig till?

Anledningar till att en anmäld deltagare inte kommer till ett event från intervjuerna:

- Vädret kan vara en anledning till varför en deltagare inte kommer till där bra väder respektive dåligt väder kan påverka.

- Prioritering av annat där ett gratis event inte prioriteras framför exempelvis vänner, familj, väder eller aktiviteter.

- Plötslig insjuknande och glömska kan vara anledningar till att en deltagare inte kommer.

- Anmälan till ett event är gratis och försäkrar en plats för deltagaren till eventet utan några som helst konsekvenser för deltagaren ifall personen inte kommer. Detta innebär att deltagaren inte förlorar en deposition i jämförelse med exempelvis ett event som kostar pengar.

2. Kan ni tänka er använda vår mobilapplikation?

Responsen från intervjupersonerna var väldigt positiv då majoriteten kan tänka sig använda mobilapplikationen. Anledningar var att badges verkar vara en bra idé att komplettera sin LinkedIn profil med och samtidigt vara ett bra sätt att spelifiera Foo Cafés event.

En gemensam nämnare var att det krävs en bra representation av badges som beskriver badgens betydelse och vad för kunskap badgen representerar. Vissa personer önskade sig även andra typer av belöningar eftersom digitala badges kanske inte är tilltalande för alla personer.

En faktor som ansågs vara väldigt relevant för användningen av mobilapplikationen är hur mycket som krävs av deltagaren, det vill säga hur enkelt det ska vara för en användare att utföra en incheckning till ett event.

3. Kan man möjligtvis öka deltagandet med dessa badges?

Responsen för denna fråga var positiv och intervjupersonerna ansåg att badges möjligtvis kunde öka deltagandet på event. Intervjupersonerna menade att badges kan vara en bidragande faktor för ökad deltagande utöver de redan existerande faktorerna, gratis pizza och dricka.

4. Finns det andra möjligheter för spelifiering?

Förslag på andra möjligheter för spelifiering från intervjuerna:

- Implementation av en leaderboard där en användare kan tävla mot en “rival” eller vänner och möjligtvis vinna olika sorters priser. En nackdel med en implementation av en leaderboard ansågs vara fusk. Fusk hade med stor sannolikhet börjat förekomma av ett flertal personer för att försöka vinna priserna som hade delats ut.

- Införandet av materiella ting som exempelvis stickers av sin badge som kan placeras på en dator eller en tröja med sin badge intryckt som logo var några ideér som nämndes bland deltagarna.

- Prioritera deltagarna som har många incheckningar och ge dessa deltagare förmånen att gå på event som är fullbokade, det vill säga att ha en prioriteringslista så att en deltagare som frekvent deltar i event har en plats även om eventet skulle vara fullbokat.

5. Har ni några förslag på annan funktionalitet för mobilapplikationen?

Förslag på funktionalitet som kan läggas till i mobilapplikationen från intervjuerna:

- Notifikationer och reminders ansågs vara en funktionalitet som bör läggas till för att påminna deltagaren att ett event kommer att ske på Foo Café och samtidigt påminna dem att göra en incheckning på eventet.

- Funktionalitet för att utforma en profil där deltagaren kan exempelvis sätta ett mål och därefter få en lista av intressanta event som hjälper deltagaren att uppnå detta mål. Ett exempel på detta är om deltagaren vill bli en mästare på design så kan en lista presenteras med event som är inriktade på design. I samband med profil borde det finnas funktionalitet för att kunna kontakta andra deltagare på samma event.

6. Övriga tankar kring FooBadges

- Badges ska vara visuellt tilltalande och utstråla känslan av autenticitet för att uppmuntra deltagaren av låsa upp badges.

- Mobilapplikationen borde inte lagra onödig information, inte ha en överdriven batterikonsumtion och inte ha en överdriven konsumtion av data.

- Fokusera på nödvändig funktionalitet först och sedan kan tillägg av annan funktionalitet läggas till i efterhand.

- Logout knappen sitter på ett konstigt ställe just nu och vore bra om ni kunde flytta den upp till Action bar eller liknande. Viktigt att ni får feedback av folk för att utveckla

applikationen så att den lyckas. Ni behöver inte använda er av all feedback men det är bra att ta emot all feedback och gå igenom den noga.

3.3.3 Slutsatser från intervjuer

Här presenteras slutsatserna från intervjuerna och på vilket sätt svaren ska bearbetas.

En slutsats kring varför en deltagare anmäler sig till ett event men inte kommer beror på faktorer som är väldigt individuella. Hur deltagarna prioriterar ett Foo Café event är väldigt individuellt där i princip vad som helst kan prioriteras före ett event. Här kan spelifiering spela en stor roll ifall belöningar som utdelas i samband med spelifiering kan få deltagarna att prioritera event före exempelvis andra aktiviteter som träning eller dylikt.

Funktionaliteten för incheckning och inloggning ska utföras snabbt och utan problem för att uppmuntra användning av mobilapplikationen. Badges måste utformas på ett sätt som ger en känsla av autenticitet och vara visuellt tilltalande för att locka deltagaren till att låsa upp dem.

Användning av konceptet spelifiering kunde ske på andra sätt som kunde involvera förmåner och fysiska objekt som belöningar för att ha många incheckningar. Enbart digitala badges för uppvisande på sociala medier kanske inte är tilltalande för alla deltagare och därför borde olika former av priser introduceras. Detta är något som möjligtvis kan göras som en framtida utvecklingsmöjlighet.

Implementering av funktionalitet för notifikationer och profilhantering bör läggas till i mobilapplikationen. Då glömska är en anledning till att en deltagare inte kommer till ett event, så är notifikationer en väldigt relevant funktionalitet som kan påminna deltagaren om eventet och även påminna deltagaren om att göra en incheckning när deltagaren väl är på eventet. Profilhantering och notifikationer ansågs av examensarbetarna vara relevant funktionalitet som möjligtvis kan implementeras som en framtida utvecklingsmöjlighet.

Övriga tankar och diskussioner kring FooBadges togs i hänsyn och hjälpte examensarbetarna att arbeta vidare med examensarbetet.

3.4 Val av badge-system

Genom en kartläggning av olika badge-system som utfördes i kapitel 2.3 så kunde en analys göras och ett badge-system kunde väljas. Kartläggningen utfördes med hjälp av dokumentationer och information från respektive badge-system samt tester av verktygen som erbjöds av respektive badge-system. Kartläggningen av de olika badge-systemen Basno, Mozilla Open Badges och TrueCred ledde till att badge-systemet Mozilla Open Badges valdes och en Badgr-server skulle därmed användas till att skapa samt utfärda badges till Foo Cafés deltagare. Detta val utfördes tillsammans med handledarna hos Foo Café och grunden till detta är främst Foo Cafés förespråkande av öppen källkod och tillgänglighet.

En av de största anledningarna till att Mozilla Open Badges valdes är den öppna specifikationen och den öppna källkoden som är tillgänglig för allmänheten [40]. Den öppna källkoden tillsammans med licensen MPL [41] gör det möjligt för en användare att utveckla koden för att passa ett företag eller för personliga behov. Den öppna specifikationen medför även att ett potentiellt personligt system kan utvecklas och att beroendet av en tredje part försvinner, vilket innebär att ens personliga system kan leva vidare trots att Mozilla Open Badges inte längre är tillgänglig.

En annan anledning till att Mozilla Open Badges valdes var att det är framtaget av ett väletablerat företag och har enligt examensarbetarna ett väldigt gott anseende globalt. Detta medför att det går att lita på Mozilla Open Badges och att systemet är av en hög kvalitet, vilket examensarbetarna kommer att ha nytta av i examensarbetet. Jämfört med TrueCred och Basno så är kännedomen av dessa inte helt klara, vilket även kunde bekräftas av handledarna hos Foo Café som inte kände till dessa företag och deras verksamhet. Handledarna hos Foo Café ansåg därför att ett val av ett badge-system bör göras med ett företags anseende i åtanke även om TrueCred och Basno bidrar med högkvalitativa badge-system.

Från ett säkerhetsperspektiv är säkerheten direkt beroende på lagringsstället som en badge befinner sig på. I Mozilla Open Badges fall beror säkerheten på tillgången till servern där filerna är lagrade, vilket betyder att om en obehörig person har direkt tillgång till servern är det möjligt att skapa egna badges. Ett exempel på det är om en obehörig person har tillgång till Foo Cafés server är det möjligt att skapa egna badges i Foo Cafés namn. Säkerheten hos Basno är direkt kopplad till användarkontot som existerar hos Basno. Om en obehörig person får tillgång till ett användarkonto hos Basno är det möjligt att skapa badges i användarkontots namn. Säkerheten hos TrueCred är beroende på ett användarkonto hos TrueCred samt det så kallade TrueCred Vault, badges kan enbart skapas av en användare hos TrueCred och TrueCred Vault förser användaren med säker lagring av badges och som sägs göra det omöjligt för utomstående att ha tillgång eller manipulera badges som är placerade i TrueCred Vault. Säkerheten var nödvändigtvis inte en anledning till varför Mozilla Open Badges valdes men det är värt att notera att säkerheten är en omfattande del av digitala badges som oftast beror på en tredje part.

3.5 Test

Här presenteras tillvägagångssättet för test av mobilapplikationen och webbapplikationen.

3.5.1 Test av mobilapplikation

Mobilapplikationen har testats enligt black-box testning som innebär att testen har utförts genom direkt interaktion med mobilapplikationen. Detta innebär att test av mobilapplikationens funktionalitet har gjorts manuellt där interaktion med mobilapplikationen ska resultera i ett förväntat utfall. Tillägg av ny funktionalitet utfördes inte förrän förväntat resultat hade uppnåtts. Ett exempel på test som utfördes är inloggning med mobilapplikationen där förväntat utfall var att användaren loggades in och Events-sidan presenterades. Alla implementerade funktioner har testats kontinuerligt under examensarbetets gång och regressionstest på samtliga funktioner har utförts i samband med tillägg av funktionalitet. Regressionstest innebär testning

av gammal funktionalitet för att kontrollera att ingen funktionalitet slutat fungera efter tillägg av ny funktionalitet. Test av mobilapplikationen har gjorts informellt där inga testfall skapades och där ingen dokumentation av testning har utförts.

3.5.2 Test av webbapplikation

Test av webbapplikationens tillägg av funktionalitet har utförts tillsammans med handledaren på Foo Café, där test med ett förväntat utfall utformades och sedan utfördes automatiskt. Detta innebär att tillägg av funktionalitet testades grundligt via automatiserade test i webbapplikationen innan funktionaliteten kunde testas tillsammans med mobilapplikationen. Ett exempel på test som utfördes i webbapplikationen var att alla badges skulle vara tillgängliga i ett JSON-objekt. Det automatiserade testet som utfördes var att två badges skulle placeras i ett JSON-objekt och det förväntade resultatet skulle då vara att JSON-objektet innehöll två badges. Denna funktionalitet testades sedan med mobilapplikationen, där samtliga badges hämtades från webbapplikationen och presenterades för användaren i mobilapplikationen.

3.5.3 Resultat av test

Den resulterade mobilapplikationen ansågs av examensarbetarna vara bevis på att test av mobilapplikationens- och webbapplikationens funktionalitet har utförts med goda resultat.

3.6 Källkritik

Här motiveras källorna som användes i examensarbete. För motivering av källorna har följande checklista använts för att bestämma källans tillförlitlighet [25]:

- Vem står bakom källan?
- Varför är källan skapad?

3.6.1 Källkritik internet

Open Badges anses vara pålitlig eftersom det är bland annat Mozilla Foundation som skapat webbsidan och som även är ansvariga över webbsidan. Open Badges är en officiell webbsida och anses därmed vara pålitlig.

IBM Badges anses vara pålitlig eftersom det är en del av företaget IBM:s officiella webbsida. IBM är ett stort och välkänt företag och deras officiella webbsida anses därmed vara pålitlig, där företagets anställda får tag på information om det behövs.

Untappd anses vara pålitlig eftersom webbsidan är Untappd:s officiella webbsida och informationen på webbsidan bör därmed vara korrekt och pålitlig för att locka potentiella användare. Det bekräftades även att informationen är pålitlig genom att mobilapplikationen testades.

Information om Foo Café anses vara pålitlig då information om dess verksamhet kan bekräftas genom att delta i deras event. Informationen används i ett syfte att locka personer och företag att delta i deras event och informationen bör därför vara korrekt.

Information från Basno's hemsida anses vara pålitlig eftersom informationen används i syfte att förse en användare med information om deras produkt samt företag, vilket bör vara korrekt och tillförlitligt. Då produkten möjligtvis konkurrerar med andra aktörer så bör det finnas en medvetenhet om att informationen kan vara vinklad för att få produkten att framstå som ett av de bättre alternativen.

Information från TrueCred's hemsida anses vara pålitlig eftersom informationen används i syfte att förse en användare med information om deras produkt samt företag, vilket bör vara korrekt och tillförlitligt. Då produkten möjligtvis konkurrerar med andra aktörer så bör det finnas en medvetenhet om att informationen kan vara vinklad för att få produkten att framstå som ett av de bättre alternativen.

Information som är tagen från artikeln *"Han arrangerar it-konferens varje dag"* kan anses vara pålitlig eftersom den är publicerad hos idg.se, som anses vara nordens största IT-nätverk. Artiklarna som publiceras hos idg.se görs i syfte att informera och bör inte innehålla information som är felaktig.

Information om Swarm är hämtad från deras officiella hemsida och denna information kunde bekräftas genom att testa applikationen, informationen anses därför vara korrekt.

Information om PostgreSQL är hämtad från deras officiella hemsida och används i ett syfte att förse användarna med information om deras databas system. Informationen anses vara korrekt eftersom felaktig information kan leda till ett sämre rykte och att användarna söker andra alternativ.

Information om positionering är hämtad från en rapport som är framtagen av LG Electronic Mobile Research, rapporten publicerades år 2008 men informationen anses vara korrekt då informationen är framtagen av en avdelning inom LG som fokuserade på att forska om teknologin bakom positionering.

Android Studio anses vara pålitlig eftersom det är företaget Android Inc:s officiella webbsida och Android är ett välkänt företag och har ingen anledning till att vinkla information på deras webbsida om Android och dess funktioner.

Eddystone anses vara pålitlig eftersom det är bland annat företagen Estimote Inc, Bluetooth, Apple och Google som är ansvariga för Eddystone och att webbsidan är Estimotes officiella webbsida. Det vore konstigt att skriva felaktig information om ens produkt på den officiella webbsidan.

Information om NINA-B1 från U-Blox är hämtad från en officiell produktbeskrivning som är publicerad på U-Blox hemsida. U-Blox har ett ekonomiskt intresse att bidra med en produktbeskrivning som måste vara korrekt för att locka potentiella användare.

Information om spelifiering är hämtad från en publikation från Media Evolution som är en stor förening i Malmö och är ett innovativt mötesplats för många organisationer samt företag. Information anses vara tillförlitligt då det kommer från en verksamhet vars syfte är att utbilda och skapa möjligheter för kompetensutveckling.

Information om Git är hämtad från företaget Atlassian som äger BitBucket, som är mjukvara som använder sig av Git. Informationen som hämtades anses vara korrekt då det kommer från en aktör som direkt använder sig av Git.

Information om produkter som exempelvis Nissan Leaf och Honda Insight är hämtad från produktbeskrivningar. Då det finns ett ekonomiskt intresse för företagens produkter så bör informationen vara korrekt för att locka potentiella kunder.

Referenser till olika tjänster som exempelvis DropBox, Trello och Facebook är refererade till respektive hemsida och examensarbetarna anser därför att informationen bör vara korrekt för att locka potentiella användare.

Information om intervjumetodik från Kungliga Tekniska Högskolan (KTH) i Stockholm som är publicerad på KTH:s hemsida anses vara pålitlig eftersom informationen kommer från ett svenskt universitet som finns för att utbilda elever och dela med sig korrekt information till sina elever. KTH har ingen anledning till att vinkla information om intervjumetodik eller ge sina elever fel information.

Information JSON är hämtad från deras officiella sida som även hänvisar vidare till en officiell publikation hos ECMA angående JSON. Examensarbetarna anser därför att information är tillförlitlig då informationen med stor sannolikhet kommer från denna publikation.

Information om Badgr är hämtad från Badgrs officiella hemsida som bidrar med information och specifikation för de potentiella användarna. Informationen bör vara korrekt för att locka potentiella användare.

3.6.2 Källkritik litteratur

Databasteknik (2005) anses vara pålitlig eftersom det är en bok som används som kurslitteratur i syfte att utbilda och bidra med korrekt information. Boken är publicerad av Studentlitteratur AB som kan anses vara ett av de större utbildningsförlagen.

Kanban and Scrum - making the most of both (2010) anses vara pålitlig då det är en bok som används som kurslitteratur i högre utbildningar och bidrar med korrekt information i syfte att utbilda.

Agile Web Development with Rails 5 (2016) anses vara tillförlitlig då boken är publicerad av The Pragmatic Bookshelf som är ett förlag som har publicerat många olika böcker om utveckling i syfte att utbilda och göra livet enklare för mjukvaruutvecklare. Information borde av den anledningen vara korrekt för att underlätta lärandet för en mjukvaruutvecklare.

Intervjumetodik (2013) anses vara pålitlig eftersom det är en bok som är publicerad av Studentlitteratur AB, som är ett förlag som publicerar böcker i utbildningssyfte och informationen i boken anses därmed vara korrekt. Det vore konstigt av ett förlag att släppa ut utbildningsböcker med fel fakta och information.

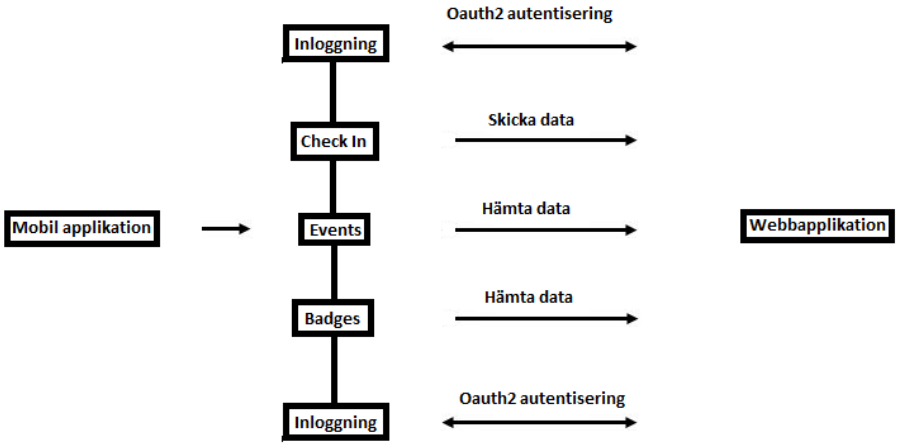
Kapitel 4

4 Resultat

I detta kapitel presenteras resultatet av prototypen FooBadges som omfattas av mobilapplikationen och webbapplikationen.

4.1 Mobilapplikation

Användargränssnittet för mobilapplikationen består av fyra sidor, Check In, Events, Badges och Inloggning. Check In är den sida som hanterar verifieringsprocessen för en incheckning till ett event. Events visar alla event som är tillgängliga för respektive stad. Badges innehåller alla badges som finns tillgängliga för användaren att låsa upp. Beroende på om en användare är inloggad eller inte så presenteras Check In och Badges annorlunda då användaren istället får en inloggningssida ifall användaren inte är inloggad. Alla sidor innehåller en navigeringsbar som gör det möjligt för en användare att nå alla olika sidor oavsett vilken sida som användaren befinner sig på för tillfället. Figur 4 visar flödet för mobilapplikationen där alla sidor är sammankopplade och visar hur kommunikationen med webbapplikationen sker.

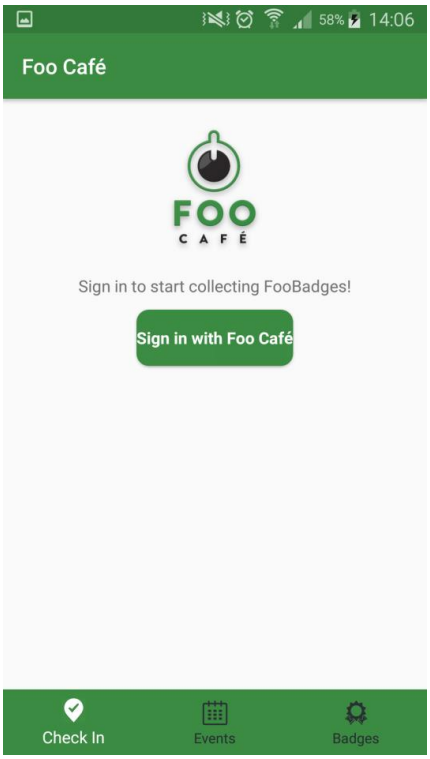


Figur 4. Flödesdiagram för mobilapplikationen och dess kommunikation med webbapplikationen.

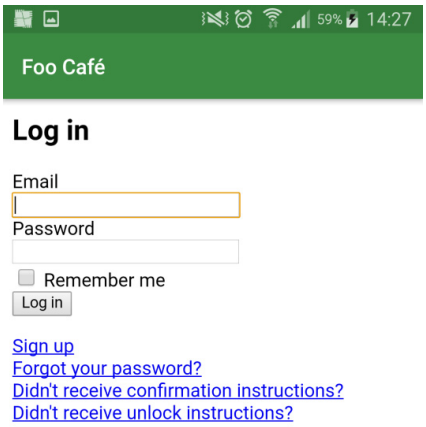
4.1.1 Inloggning

För att bestämma vilken deltagare som är berättigad till en badge krävs en unik identifierare. Detta uppnås genom att ett inloggningssystem implementerades i webbapplikationen. Mobilapplikationen använder sig av webbapplikationens inloggningssida för att identifiera användarna. Vid lyckad inloggning efterfrågar mobilapplikationen användaren om tillåtelse för att använda sig av användarkontot. Mobilapplikationen använder sig enbart av inloggning ifall en användare vill utföra en incheckning eller kontrollera sina badges, vilket innebär att applikationen även kan användas för att exempelvis kontrollera vilka event som är tillgängliga. Inloggningen sparas sedan undan så att användaren inte behöver logga in varje gång en incheckning ska utföras.

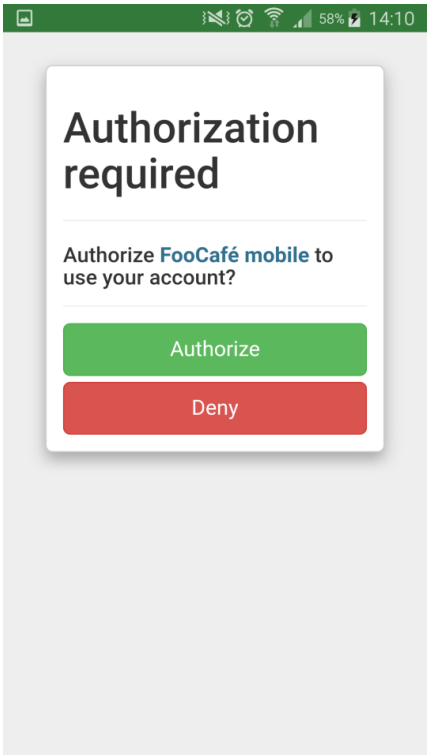
Processen för inloggning visualiseras med hjälp av figur 4.1, 4.2, 4.3 och 4.4. Figur 4.1 är inloggningssidan som visas när en användare vill utföra en incheckning eller kontrollera sina badges utan att vara inloggad. Figur 4.2 omdirigerar användaren till webbapplikationens inloggningssida efter att en användare har tryckt på knappen “Sign In with Foo Café” i figur 4.1. Figur 4.3 visar en sida där mobilapplikationen efterfrågar tillåtelse att använda användaruppgifterna. Om användaren tillåter användning av användaruppgifterna presenteras sidan Events som kan ses i figur 4.4.



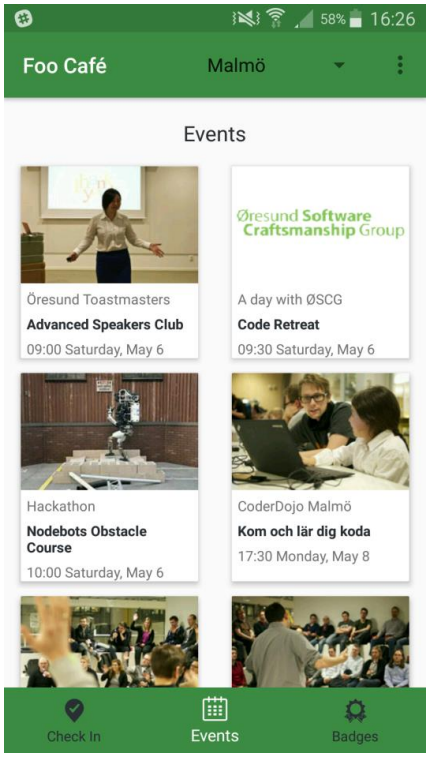
Figur 4.1. Inloggningssida för mobilapplikationen



Figur 4.2. Omdirigering till webbapplikationens inloggningssida



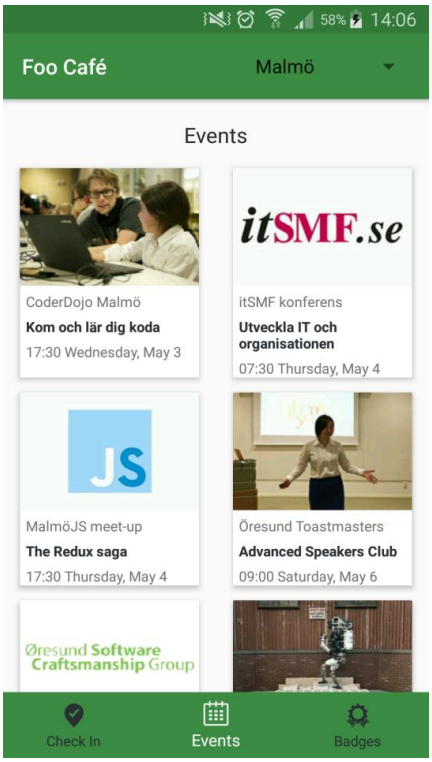
Figur 4.3. Vid lyckad inloggning efterfrågar mobilapplikationen tillåtelse att använda användaruppgifterna.



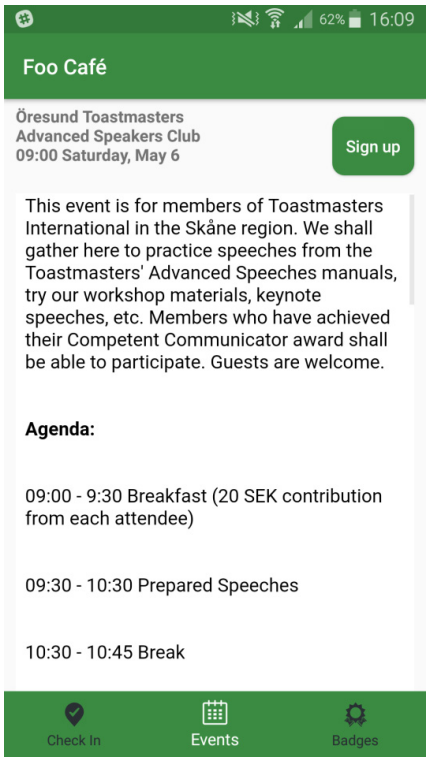
Figur 4.4. Events-sidan presenteras om användaren tillåter användandet av användaruppgifterna.

4.1.2 Events

Events är en sida där användaren kan se alla event som är tillgängliga i Malmö, Stockholm eller Köpenhamn. Denna sida är tillgänglig för samtliga och kräver ingen inloggning. Det som sker är att mobilapplikationen hämtar en lista av event som är tillgängliga på Foo Cafés hemsida som ett JSON-objekt och konverterar JSON-objektet med hjälp av Retrofit till Javaobjekt. Informationen som fås av JSON-objektet visualiseras sedan för användaren. Om användaren trycker på ett event öppnas en sida med beskrivning på eventet samt ger användaren möjlighet till att anmäla sig till eventet. Events visualiseras i figur 4.5 och i figur 4.6 visas ett exempel på hur en beskrivning för ett event presenteras.



Figur 4.5. Events-sida



Figur 4.6. Beskrivning av ett event.

4.1.3 Check In

Check In är en sida där användaren kan utföra en incheckning till ett event genom att trycka på eventet som ska checkas in. Sidan presenterar alla event som är tillgängliga för dagen vilket underlättar för användarna. Det som sker sedan är en multi-faktor verifiering för användarens incheckning där olika faktorer måste uppfyllas för att en incheckning ska anses vara giltig. Anledning till en multi-faktor verifiering är för att förhindra användaren från att förfalska en incheckning och samtidigt vara faktorer som kan endast uppfyllas om användaren är närvarande på ett event hos Foo Café. Följande faktorer behöver uppfyllas för att en incheckning ska anses vara giltig:

- GPS - en faktor som använder sig av positionering för att hitta användarens position. I detta sammanhang kontrollerar mobilapplikationen ifall användaren är på ett Foo Café med hjälp av koordinater. Koordinaterna för användaren jämförs med koordinaterna för ett Foo Café med en felmarginal på 2 km, där felmarginalen existerar då användning av GPS inte alltid är möjlig eller har låg precision.

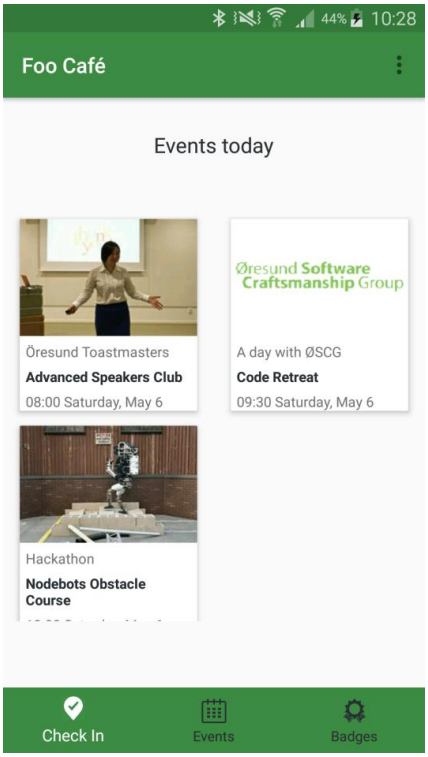
- Beacon - en faktor som endast kan uppfyllas om en deltagare är närvarande på ett Foo Café då signalen från en beacon installerad av Foo Café endast är tillgänglig vid tillfället av ett event. Mobilapplikationen skannar omgivningen efter beacons och när den känner igen en signal från en beacon hos Foo Café registreras beacon-identifieraren.

Förutom GPS och beacon inkluderar en incheckning även identifierare för användaren, eventet och tidsstämpel för incheckningen. Med hjälp av de olika faktorerna skickas incheckningen till webbapplikationen som hanterar lagringen och kontrollerar om incheckningen är giltig eller inte.

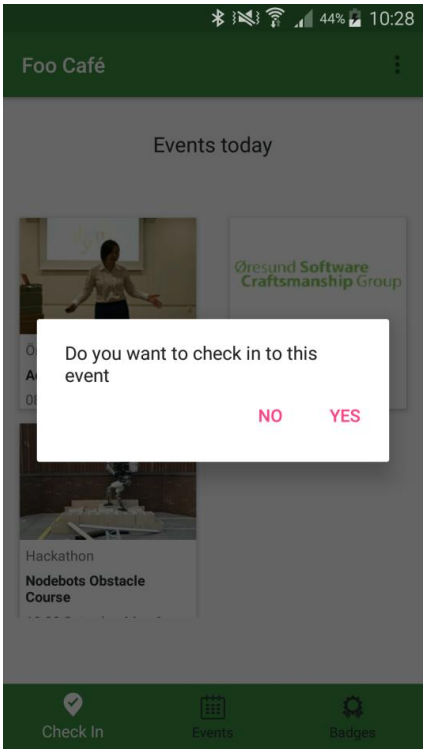
Felmeddelanden som kan visas för användaren är i följande ordning:

- “Please turn on your WiFi/GPS” - vilket indikerar på att användaren inte kan lokaliseras.
- “You are not at a Foo Café” - vilket indikerar användaren inte är på ett Foo Café.
- “Please turn on your Bluetooth” - vilket indikerar på att användaren inte kan skanna omgivningen för en beacon.
- "Contact Foo café regarding beacon failure" - vilket indikerar på hårdvarufel hos Foo Cafés beacon.

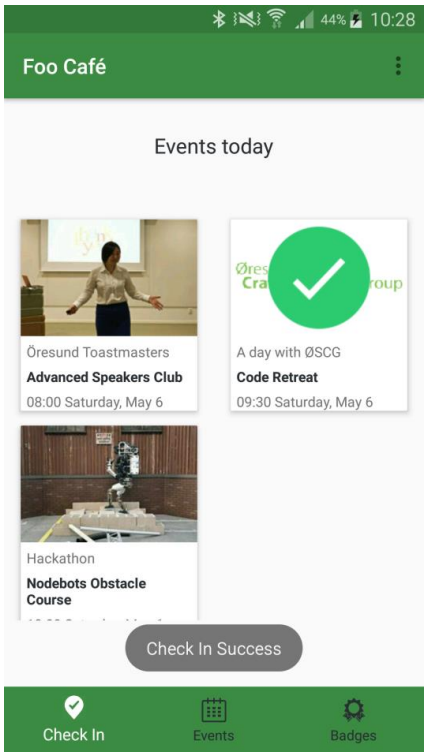
Processen för att utföra en incheckning visualiseras med hjälp av figur 4.7, 4.8, 4.9 och 4.10. Figur 4.7 är Check In-sidan som visas när en användare vill utföra en incheckning och presenterar alla event som är tillgängliga för dagen. Användaren trycker sedan på det event som användaren deltar i och en dialog visas enligt figur 4.8. Vid en lyckad incheckning presenteras sidan enligt figur 4.9. där eventet markeras med ett bock-tecken för att markera att incheckningen har lyckats. Figur 4.10 visar ett exempel på när en användare försöker utföra en incheckning utan att ha skannat omgivningen för en beacon.



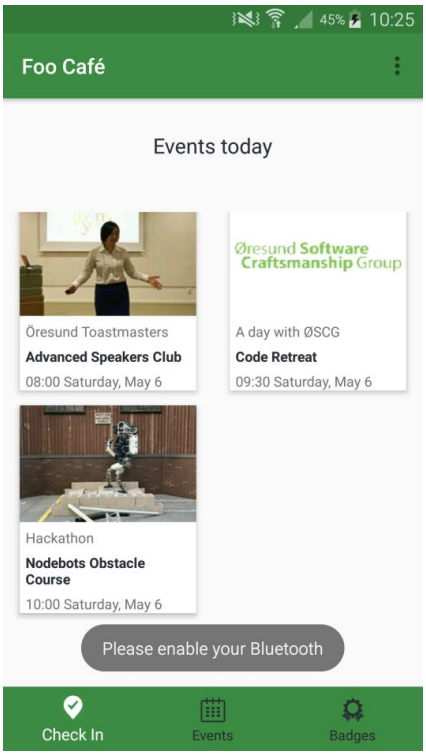
Figur 4.7. Check In-sida



Figur 4.8. Incheckning till ett event



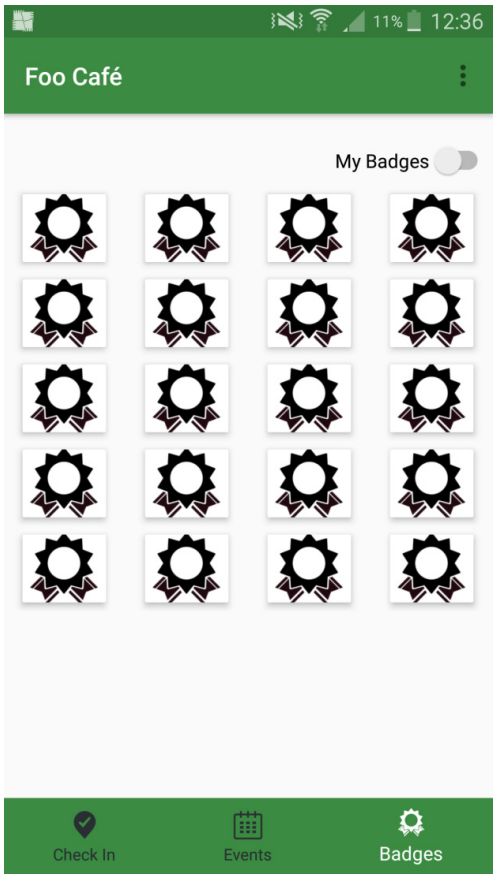
Figur 4.9. Vid lyckad incheckning



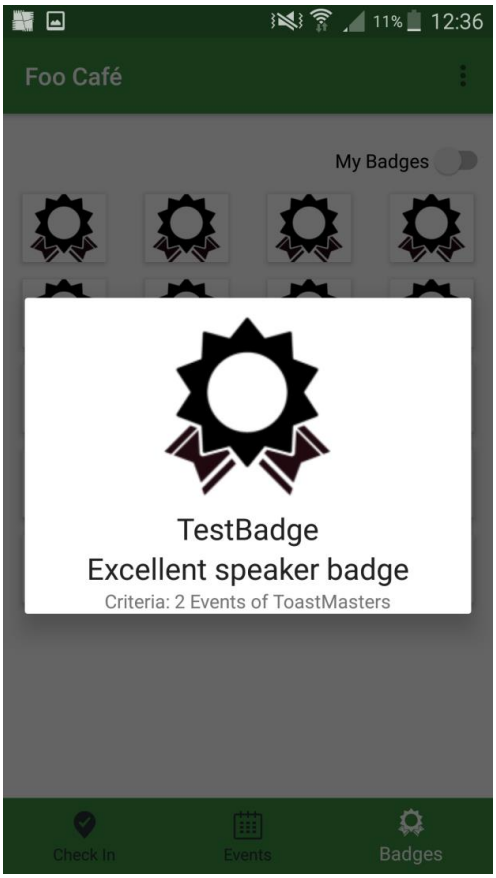
Figur 4.10. Vid misslyckad incheckning presenteras ett av felmeddelanden.

4.1.4 Badges

Badges är en sida där användaren kan se alla badges som finns tillgängliga hos Foo Café. Om användaren trycker på en badge presenteras informationen om badgen. Om användaren trycker på knappen “My Badges” presenteras listan av alla badges som användaren har fått. Samma princip för hämtande av listan av event utförs, där alla badges hämtas i form av ett JSON-objekt. Konvertering av JSON-objektet till Javaobjekt sker sedan och alla badges visualiseras för användaren. Figur 4.11 visar exempel på alla badges som finns tillgängliga och figur 4.12 visar detaljerad information om badgen som användaren har tryckt på.



Figur 4.11. Visualisering av test badges.



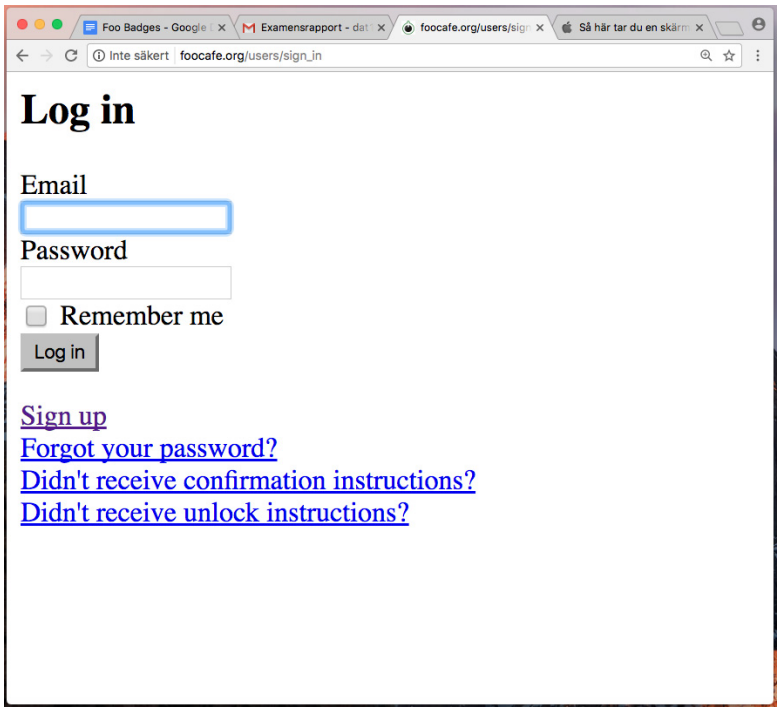
Figur 4.12. Detaljerad information av en test badge

4.2 Webbapplikation

Här presenteras den funktionalitet som har lagts till på Foo Cafés webbapplikation, detta involverar exempelvis inloggning och hur integrering av det valda badge-systemet kommer att ske.

4.2.1 Inloggning

Ett inloggningssystem implementerades för webbapplikationen där en användare skriver in sitt användarnamn och lösenord. Kontrollen för inloggningen sker sedan via databasen i webbapplikationen. På webbapplikationen implementerades även funktionalitet för registrering och felhantering av användarkonton som exempelvis glömt lösenord. Visualisering av inloggningssidan kan ses i figur 4.13. För att tillåta mobilapplikationen att utföra autentisering via webbapplikationen implementerades en OAuth 2 autentisering med hjälp av Doorkeeper, som erbjuder OAuth 2 autentisering för applikationer skrivna i Rails. OAuth 2 autentiseringen används enbart när en användare loggar in via mobilapplikationen.



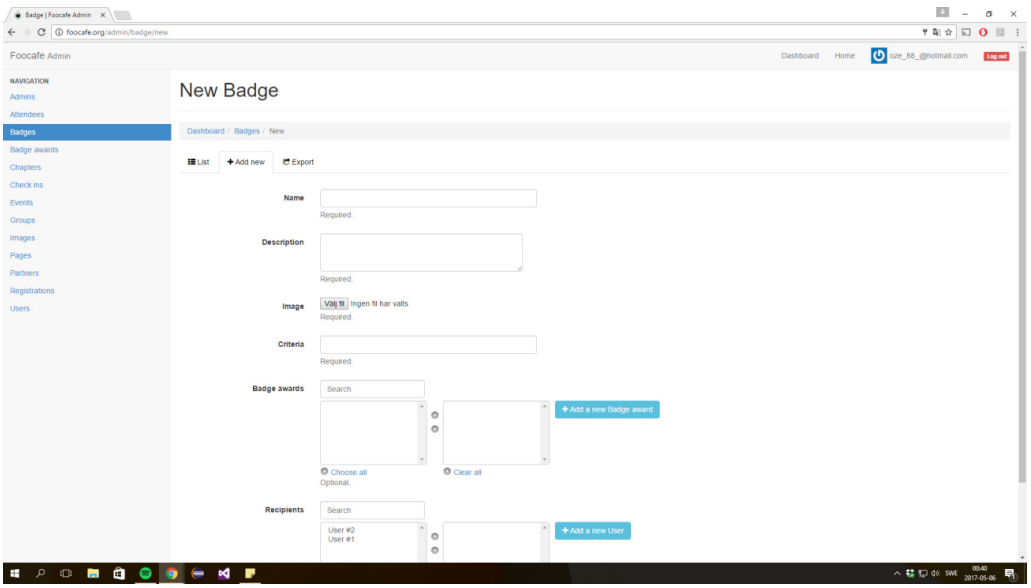
Figur 4.13. Inloggning på Foo Cafés webbapplikation.

4.2.2 Integration av Mozilla Open Badges

Integration av Mozilla Open Badges på webbapplikationen sker genom att en Badgr-server installerades på Foo Cafés server. Installation av Badgr har utförts enligt en guide som finns på Badgrs Github sida [55]. Installation av Badgr hos Foo Café medför total kontroll över badge-systemet och innebär att beroendet av en tredje part försvinner. Det som avses med att beroendet av en tredje part försvinner är att skapande, lagring och utfärdande av badges inte blir beroende på tjänster som erbjuds från Badgrs officiella hemsida [46] och kan leva vidare även om Badgrs officiella hemsida slutar erbjuda sin tjänst. Webbapplikationen kommunicerar sedan med den egna Badgr-servern genom API-anrop, där POST- och GET- förfrågningar skickas för att kalla på funktioner som automatiskt sköter exempelvis skapandet av badges enligt Mozillas Open Badges process som beskrivs i kapitel 2.4.1.1.

4.2.3 Badges

Funktionalitet för att skapa en badge implementerades på webbapplikationen och tillåter en administratör hos Foo Café att skapa badges. Skapandet av en badge utförs genom att fylla i fält som presenteras i figur 4.14. Ett JSON-objekt konstrueras som innehåller informationen som fylldes i och en POST-förfrågan skickas till den egna Badgr-servern för att skapa badgen.



Figur 4.14. Användargränssnitt för att skapa en badge på webbapplikationen

På webbapplikation implementerades funktionalitet för kopplingen mellan badges och incheckningar. En incheckning från mobilapplikationen skickas till webbapplikationen där Observer-mönstret används för kontrollera om en användare har uppnått kriteriet för att få en badge. Observer-mönstret är ett designmönster som används för att exempelvis notifiera en klass att en ändring har utförts och i detta fall används Observer-mönstret på klassen check-ins

på webbapplikationen. Detta innebär att vid varje ny incheckning till webbapplikation så kallas en metod hos webbapplikationen som går igenom alla badges som finns i systemet för att kontrollera ifall användaren har uppfyllt kriteriet för någon badge som finns i systemet. Om kriteriet uppfylls konstrueras en API-förfrågan till Badgr-servern som utfärdar badgen och badgen skickas ut till användarens e-postadress.

För att tillåta mobilapplikationen hämta samtliga badges som finns lagrade hos Foo Café implementerades funktionalitet för att möjliggöra API-anrop från mobilapplikationen till webbapplikationen. GET- och POST-förfrågningar skickas till webbapplikationen som returnerar JSON-objekt med information. Detta utfördes även för incheckningar där en incheckning utförs genom att skicka en POST-förfrågan till webbapplikationen och GET-förfrågan utförs för att hämta exempelvis en lista av event.

Kapitel 5

5 Slutsats

I detta kapitel besvaras problemformuleringarna som togs upp i kapitel 1.4. Framtida utvecklingsmöjligheter diskuterades även i detta kapitel.

5.1 Problemformulering

- *Vilka Open Badge-system finns på marknaden och hur passar de till denna tillämpning?*

Ett antal badge-system som finns ute på marknaden var Basno, TrueCred och Mozilla Open Badges. Dessa badge-system har kartlagts och informationen kring dessa badge-system har bearbetats. Kartläggningen för de olika badge-system som genomfördes kan läsas i kapitel 2.3. Slutsatsen som kunde dras för de olika badge-systemen är att alla de olika systemen kunde potentiellt användas för vår tillämpning. Badge-systemet som valdes var Mozilla Open Badges och under kapitel 3.4 motiveras varför Mozilla Open Badges valdes framför de andra alternativen.

- *Hur sker integreringen av det valda Open Badge-systemet? Hur säker är denna typ av integrering? Kan deltagarna hacka sig in och okontrollerat skapa Badges?*

FooBadges använder sig av källkoden för Badgr för integration av Mozilla Open Badges. En Badgr-server installerades på Foo Cafés server och webbapplikationen kommunicerar med denna Badgr-server via API-anrop. Genom API-anrop kan funktionerna som finns hos Badgr användas för FooBadges. En slutsats kring säkerheten är att det är bara säkert så länge en obehörig inte har tillgång till en administratörs användarkonto. Detta beror på att det krävs en administratörsbehörighet för att skapa badges och göra API-anrop till Badgr-servern.

- *Kartläggning av deltagarna vid event genom enkäter/intervjuer. Detta ska göras för att få deltagarnas syn på ett potentiellt Open Badge-system. Varför deltar de i olika event? Kan ett Open Badge-system uppmuntra deltagande i event?*

Kartläggning av deltagarna har utförts med hjälp av intervjuer där återkoppling av prototypens funktionalitet och diverse synpunkter kring ett Open Badge-system har bearbetats. Intervjuerna har utförts enligt semistrukturerade intervjuer med ett par förberedda frågor som sedan kunde följas upp med följdfrågor. Responsen från deltagarna som ställde upp på intervjuerna har varit värdefull och omfattande. Detaljerad information om intervjuernas uppbyggnad och resultat kan läsas under kapitel 3.3.

- *Vilka typer och nivåer av badges ska implementeras? Typer och nivåer avser vilken nivå som eventet riktar sig mot, det vill säga exempelvis om nivån är på en avancerad nivå ska en "Avancerad nivå" badge utdelas.*

En specifikation för de olika Badges som ska implementeras är framtagen tillsammans med Foo Café. Enligt specifikationen finns det så kallade "Tracks" som kan anses vara spår som en användare följer för att få en badge. I dessa spår definieras typen av badgen och innehåller oftast minst tre nivåer. De olika nivåerna involverar en bas-nivå, "Serious"-nivå och "Master"-nivå. För en mer detaljerad information om typer och nivåer som ska implementeras hänvisas till bilaga 2.

- *På vilket sätt kommer webbapplikationen, mobilapplikationen och databasen att integrera med varandra?*

Kommunikationen som sker mellan databasen, webbapplikationen och mobilapplikationen sker genom API-anrop, där POST- och GET- förfrågningar skickas till webbapplikationen. Informationen som skickas och returneras utförs i form av JSON-objekt. Webbapplikationen skapar dessa JSON-objekt som används av mobilapplikationen och kommunicerar direkt med databasen där all data kring badges och event finns tillgänglig. Mobilapplikationen hämtar exempelvis alla event och badges som finns i databasen för att sedan visualisera detta i mobilapplikationens användargränssnitt. Mobilapplikationen skickar även information om användarens närvaro vid ett event genom verifieringsprocessen. För detta används biblioteket Retrofit som hanterar konvertering av JSON-objekt till Java och vice versa.

- *Hur ska deltagarna använda mobilapplikationen för att verifiera att de har varit närvarande vid ett event?*

Deltagarna ska använda sig av en incheckningsfunktion för att verifiera att de har varit närvarande på ett event. Funktionen är byggd på multi-faktor verifiering som innebär att deltagaren går genom flera faktorer för att verifiera att deltagaren har varit närvarande vid ett event. Anledningen till en multi-faktor verifiering är att det ska minimera chansen för deltagarna att kunna förfalska sin närvaro vid ett event. En kontroll görs även hos webbapplikationen för att kontrollera att en incheckning är giltig. Om det saknades faktorer för verifiering så hade det exempelvis varit möjligt för en deltagare att sitta hemma och verifiera att personen har varit närvarande och fått sin badge även om personen inte fysiskt har varit närvarande på ett Foo Café event. Kapitel 4.1.3 beskriver hur processen går till för att verifiera att en deltagare har varit på ett event.

- *Hur ska Foo Café använda sig av webbapplikationen för att skapa badges?*

Funktionalitet för att skapa badges implementerades i Foo Cafés webbapplikation och Foo Café har därmed möjligheten att använda sig av webbapplikationen för att skapa badges. En administratör hos Foo Café har tillgång till att skapa en badge genom att fylla i fälten som krävs för att skapa en badge och sedan utförs ett API-anrop till Badgr-servern som skapar badgen

enligt Mozilla Open Badges processen som beskrivs i kapitel 2.4.1.1. För en mer detaljerad information och visuell presentation av användargränssnittet som används för att skapa badges hänvisas kapitel 4.2.3.

- *Hur ska konceptet spelifiering (gamification) användas och hur ska det implementeras tillsammans med systemet?*

Konceptet spelifiering används i form av att belöna deltagarna med olika badges som deltagarna får av att delta i olika event hos Foo Café. Samtidigt som deltagarna får en belöning i form av en badge så utvecklar de sin egen personliga kompetens. Detta kan sedan uppvisas på sociala medier som ett bevis på att man tar hand om sin personliga kompetensutveckling, vilket möjligtvis är attraktivt på ett CV eller dylikt.

En slutsats som kan dras är att konceptet spelifiering kunde användas mer i form av en implementation av exempelvis en leaderboard eller virtuella mynt som man kan få för varje incheckning till ett event som utförs. Leaderboard kunde möjligtvis visa vilken deltagare som har deltagit på flest event och belöna denna deltagare med en speciell belöning i slutet på en månad. Virtuella mynt kunde möjligtvis användas för att handla upplåsningsbara objekt. Detta är något som kan göras i en framtida utvecklingsfas då det inte har implementerats för prototypen.

Spelifieringen kunde även utföras så att materiella ting och förmåner kunde delas ut som exempelvis olika stickers som kan placeras på olika sorters objekt eller diverse rabatt på Foo Cafés sortiment. En annan förmån kunde även vara att en deltagare kunde hamna i en så kallad prioriteringslista där deltagaren har förtur till ett event och därmed ha möjligheten till att delta i ett event som är fullbokat.

5.2 Reflektion över etiska aspekter

FooBadges är ett Open Badge-system som kommer att leda till att deltagare i event på Foo Café kommer att bli belönade med olika sorters badges som bevis på utökad kunskap inom olika ämnen. Detta kommer troligtvis leda till att deltagare kommer att börja delta i fler event på Foo Café och få mer kunskap samt ett större kontaktnätverk. Ett kontaktnätverk som kan möjliggöra eventuella jobberbjudanden från diverse kontakter och bidra till samhällsnytta.

FooBadges kan anses vara uppmuntrande för följande inom samhället:

- Deltagare som får en utökad kunskap inom olika ämnen och ett större kontaktnätverk, vilket kan leda till en större möjlighet till att få ett jobb, eftersom badges utfärdade av Foo Café kommer att kunna användas i arbetslivet och läggas till i exempelvis ett CV eller på LinkedIn.

- Företag som kan få ett större inblick på en persons kompetens och ambition vid läsning av ett CV innehållande badges i jämförelse med ett CV utan badges. Företag som

organiserar olika event på Foo Café kommer även få möjligheten att få träffa ett flertal personer tack vare FooBadges och bidra till samhällsnytta genom att möjligtvis rekrytera ett antal personer till sitt företag.

5.3 Framtida utvecklingsmöjligheter

Resultatet av examensarbetet lämnar utrymme för vidareutveckling där diverse funktionalitet kan implementeras på mobilapplikationen. Den funktionalitet som bör prioriteras först i en vidareutvecklingsfas är djupare profilhantering och notifikationer. Djupare profilhantering innebär att en användare ska ha större möjlighet att personligt utforma sin profil. Därför bör det finnas möjlighet för att exempelvis dela aktivitet, dela kontaktuppgifter och sätta personliga målsättningar. Notifikationer är även funktionalitet som kan behövas då glömska är en faktor som bidrar till att en deltagare kanske glömmer att det är ett event eller glömmer att utföra en incheckning.

Som det tidigare är nämnt i kapitel 1.6 är tanken att vid en lyckad prototyp kan en möjlighet för vidareutveckling vara att skapa mobilapplikationen för operativsystemen iOS och Windows. Detta kan därmed utföras för att göra mobilapplikationen tillgänglig för många olika operativsystem och potentiellt öka användningen av mobilapplikationen.

Från intervjuerna i kapitel 3.3 önskades även andra former av priser än badges. Detta är något som kan utföras genom att exempelvis införa en slags digital valuta som en användare kan få för incheckningar på event. Tanken är då att denna digitala valuta kan användas för att möjligtvis låsa upp diverse förmåner hos Foo Café eller dylikt. En annan form av priser önskades i form av materiella ting, men detta är något som examensarbetarna ansåg vara en önskan som bör hanteras av personalen hos Foo Café

En framtida utvecklingsmöjlighet kan vara att utföra incheckningar på andra sätt som kanske kan involvera QR kod eller NFC. Detta kan utvecklas för att tillåta en användare ha flera möjligheter att utföra en incheckning än de alternativ som har behandlats i detta examensarbete. Då det möjligtvis finns deltagare som inte använder sig av Bluetooth eller Wi-Fi, blir detta en lösning som kan möjligtvis tillgodose samtliga användare.

Webbapplikationen kan även vidareutvecklas där funktionalitet för att tillåta en användare visualisera badges och statistik gällande deltagande i event. Med andra ord finns det stora möjligheter för vidareutveckling av både mobilapplikationen och webbapplikationen.

Kapitel 6

6 Terminologi

BLE	Strömsnål version av Bluetooth
DBMS	Database management system
Doorkeeper	Tjänst för OAuth 2 autentisering
DRY	Princip inom programmering för att eliminera duplicerad kod
GPS	Global Positioning System
HTTP-klient	Program som använder sig av HTTP för att ansluta till en webbserver
JSON	Textbaserat format som används för att utbyta data
MVC	Model-View-Controller
MVCC	Multiversion concurrency control
NFC	Near Field Communication
OAuth 2	Ramverk för autentisering
Persistens	Objekt som skapas i ett program finns kvar mellan programkörningar
QR code	Quick Response code
RoR	Ruby on Rails
Spelifiering	Koncept som använder sig av spelmekanismer
XML	Extensible Markup Language

Kapitel 7

7 Referenser

[1] *Participating issuers*
https://openbadges.org/about/participating-issuers/
Hämtad 16/2-2017

[2] *IBM Badges*
https://www-03.ibm.com/services/learning/ites.wss/zz-en?pageType=page&c=M425350C34234U21
Hämtad 16/2-2017

[3] *Untappd*
https://untappd.com/
Hämtad 16/2-2017

[4] *About Foo Café*
http://foocafe.org/global/about
Hämtad 16/2-2017

[5] *Foo Café manifesto*
http://foocafe.org/global/manifesto
Hämtad 16/2-2017

[6] *Han arrangerar it-konferens varje dag*
http://computersweden.idg.se/2.2683/1.498831/han-arrangerar-it-konferens-varje-dag
Hämtad 16/2-2017

[7] *Android Studio releases*
https://developer.android.com/studio/releases/index.html
Hämtad 17/2-2017

[8] *Android Studio intro*
https://developer.android.com/studio/intro/index.html
Hämtad 17/2-2017

[9] *Participating issuers*
https://openbadges.org/about/participating-issuers/
Hämtad 2/3-2017

[10] *Give yourself a badge*
https://github.com/mozilla/openbadges-backpack/wiki/New-Issuers:-Give-Yourself-a-Badge

Hämtad 20/3-2017

[11] *Mozilla backpack*
https://backpack.openbadges.org/backpack/welcome
Hämtad 20/3-2017

[12] *Eddystone*
http://developer.estimote.com/eddystone/
Hämtad 21/3-2017

[13] *Basno business*
https://basno.com/business
Hämtad 21/3-2017

[14] *About Basno*
https://basno.com/about
Hämtad 21/3-2017

[15] *About TrueCred*
https://www.truecred.com/about-accreditrust/
Hämtad 21/3-2017

[16] *About PostgreSQL*
https://www.postgresql.org/about/
Hämtad 22/3-2017

[17] *PostgreSQL history*
https://www.postgresql.org/about/history/
Hämtad 22/3-2017

[18] *PostgreSQL advantages*
https://www.postgresql.org/about/advantages/
Hämtad 22/3-2017

[19] *Swarm*
https://www.swarmapp.com/
Hämtad 23/3-2017

[20] T. McCarthy, T. Risch,*"Databasteknik",* Lund, Studentlitteratur, ISBN: 978-91-44-04449-1, 2005
[21] *Spelifiering - om hur vi kan använda spelmekanik i saker som inte är ett spel*
http://mediaevolution.se/sites/default/files/spelfiering_sv.pdf
Hämtad 23/3-2017

[22] *What is Git*
https://www.atlassian.com/git/tutorials/what-is-git
Hämtad 23/3-2017

[23] *Location based services for mobiles: Technologies and standards*
http://to.swang.googlepages.com/ICC2008LBSforMobilessimplifiedR2.pdf
Hämtad 24/3-2017

[24] *Nina-B1 series information*
https://www.u-blox.com/en/product/nina-b1-series#product-information
Hämtad 28/3-2017

[25] *Källkritik på Internet*
https://www.iis.se/lar-dig-mer/guider/kallkritik-pa-internet/
Hämtad 29/3-2017

[26] *Android Tv*
https://www.android.com/tv/
Hämtad 30/3-2017

[27] *Nexus 5*
http://www.lg.com/se/mobiltelefoner/lg-Nexus-5-D820
Hämtad 30/3-2017

[28] *Digital badges*
https://www.hastac.org/initiatives/digital-badges
Hämtad 30/3-2017

[29] *About Open Badges*
https://openbadges.org/about/
Hämtad 30/3-2017

[30] *Basno*
https://basno.com/
Hämtad 30/3-2017

[31] *What Is TrueCred*
https://www.truecred.com/what-is-truecred/truecred-suite/
Hämtad 30/3-2017

[32] *Trello*
https://trello.com/
Hämtad 30/3-2017

[33] *Nissan Leaf*
https://www.nissanusa.com/electric-cars/leaf/
Hämtad 30/3-2017

[34] *About Tetris*
http://tetris.com/about-tetris/
Hämtad 30/3-2017

[35] *Honda Insight*
http://world.honda.com/INSIGHT/eco/
Hämtad 30/3-2017

[36] *What is a Beacon?*
https://kontakt.io/beacon-basics/what-is-a-beacon/
Hämtad 21/3-2017

[37] *About DropBox*
https://www.dropbox.com/about
Hämtad 5/4-2017

[38] H. Kniberg, M.Skarin, "*Kanban and Scrum - making the most of both"*, C4Media, ISBN: 978-0-557-13832-6, 2010

[39] *The Agile Manifesto*
https://www.agilealliance.org/agile101/the-agile-manifesto/
Hämtad 5/4-2017

[40] *Open Badge Specification*
https://openbadgespec.org/
Hämtad 5/4-2017

[41] *Mozilla public license*
https://www.mozilla.org/en-US/MPL/2.0/
Hämtad 5/4-2017

[42] *Open Badges flow diagram*
https://openbadges.org/images/openbadges-flow-diagram-mozilla.png
Hämtad 6/4-2017

[43] *What is JSON*
https://developers.squarespace.com/what-is-json/
Hämtad 10/4-2017

[44] *JSON introduction*
http://www.json.org/
Hämtad 10/4-2017

[45] *Retrofit*
http://square.github.io/retrofit/
Hämtad 10/4-2017

[46] *Badgr*
https://info.badgr.io/
Hämtad 10/4-2017

[47] *Badgr source code license*
https://github.com/concentricsky/badgr-server/blob/master/LICENSE
Hämtad 10/4-2017

[48] S.Ruby, "Agile Web Development with Rails 5", Pragmatic Bookshelf, ISBN: 978-1-68050-171-1, 2016

[49] *RubyMine - the Best IDE for...*
https://www.jetbrains.com/ruby/features/
Hämtad 28/4-2017

[50] *The OAuth 2.0 Authorization Framework*
https://tools.ietf.org/html/rfc6749
Hämtad 1/5-2017

[51] *Doorkeper-OAuth 2 provider for Rails*
https://github.com/doorkeeper-gem/doorkeeper
Hämtad 1/5-2017

[52] *Spring Security OAuth*
https://projects.spring.io/spring-security-oauth/
Hämtad 1/5-2017

[53] Annika Lantz, "*Intervjumetodik*", Studentlitteratur AB, ISBN: 978-9-144-08123-6, 2013

[54]
http://www.nada.kth.se/kurser/kth/2D1410/04_05/kurspm/intervju.pdf
Hämtad 20/4-2017

[55] *Badgr-server README*
https://github.com/concentricsky/badgr-server
Hämtad 5/4-2017

[56] *Badgr API*
https://api.badgr.io/docs/
Hämtad 5/4-2017

7.1 Figur referenser

[Figur 1] Skapad av examensarbetarna
[Figur 2.1] Skapad av examensarbetarna
[Figur 2.2] Skapad av examensarbetarna
[Figur 2.3] Skapad av examensarbetarna
[Figur 2.4] Skapad av examensarbetarna
[Figur 2.5] Skapad av examensarbetarna
[Figur 2.6] https://github.com/mozilla/openbadges-backpack/wiki/New-Issuers:-Give-Yourself-a-Badge
[Figur 2.7] https://github.com/mozilla/openbadges-backpack/wiki/New-Issuers:-Give-Yourself-a-Badge
[Figur 2.8] https://github.com/mozilla/openbadges-backpack/wiki/New-Issuers:-Give-Yourself-a-Badge
[Figur 2.9] https://github.com/mozilla/openbadges-backpack/wiki/New-Issuers:-Give-Yourself-a-Badge
[Figur 2.10] Skapad av examensarbetarna
[Figur 2.11] Skapad av examensarbetarna
[Figur 2.12] Skapad av examensarbetarna
[Figur 2.13] Skapad av examensarbetarna
[Figur 2.14] https://www.truecred.com/what-is-truecred/truecred-suite/
https://www.truecred.com/policies-and-terms-of-service/
[Figur 2.15] http://blog.ifuturz.com/wp-content/uploads/2013/03/railsmvc.png
[Figur 2.16] https://www.codeproject.com/KB/aspnet/344292/mvc.PNG
[Figur 2.17] Skapad av examensarbetarna
[Figur 2.18] Skapad av examensarbetarna
[Figur 2.19] http://mediaevolution.se/sites/default/files/spelfiering_sv.pdf
[Figur 2.20] https://about.gitlab.com/features/issueboard/
[Figur 3.1] Skapad av examensarbetarna
[Figur 3.2] Skapad av examensarbetarna
[Figur 3.3] Skapad av examensarbetarna
[Figur 4] Skapad av examensarbetarna
[Figur 4.1] Skapad av examensarbetarna
[Figur 4.2] Skapad av examensarbetarna
[Figur 4.3] Skapad av examensarbetarna
[Figur 4.4] Skapad av examensarbetarna
[Figur 4.5] Skapad av examensarbetarna
[Figur 4.6] Skapad av examensarbetarna

[Figur 4.7] Skapad av examensarbetarna
[Figur 4.8] Skapad av examensarbetarna
[Figur 4.9] Skapad av examensarbetarna
[Figur 4.10] Skapad av examensarbetarna
[Figur 4.11] Skapad av examensarbetarna
[Figur 4.12] Skapad av examensarbetarna
[Figur 4.13] Skapad av examensarbetarna
[Figur 4.14] Skapad av examensarbetarna

Kapitel 8

8 Bilagor

8.1 Bilaga 1. Mall för intervjuer

Uppgifter om personen som intervjuas:

- Namn:
- Foo Café deltagare, organisatör eller talare:
- Yrkestitel:

Frågor som användes i intervjuerna:

1. Vilka anledningar finns det för att en deltagare inte kommer till ett event de anmält sig på?
2. Kan ni tänka er använda vår mobilapplikation ?
3. Kan man möjligtvis öka deltagandet med dessa badges?
4. Finns det andra möjligheter för spelifiering förutom badges?
5. Har ni några förslag på annan funktionalitet för mobilapplikationen?
6. Övriga tankar kring FooBadges?

8.2 Bilaga 2. FooBadges specifikation

I denna bilaga finner ni en specifikation för Foo Cafés badges, där beskrivning av nivåer för en badge beskrivs och kraven som behöver uppfyllas för att få badgen.

Attendee badges
There are different tracks that can be taken to get attendee badges.

All attendees gets the first badge as soon as you attend an event at Foo Café:
Taking care of my Competence

Back-end track:

After attending 3 events of various types of;

- Software Development
- Architecture

You gain:
Back-end Geek
After attending 6 events of this type you gain:
Back-end Serious Geek

After attending 10 events of this type, where at least one of them should be a hands-on event, you gain;
Back-end Master Geek

Front-end track:

After attending 3 events of various types of:

- JavaScript
- User Interaction

You gain:
Front-end Geek
The badges has the same levels as the Back-end track.

Full-stack track:

After attending 5 events of various types:

- Software Development
- Architecture
- Functional Programming
- JavaScript

You gain;
Full-stack Geek
After attending 10 of this types you gain;
Full-stack Serious Geek
After attending 15 of this types, where at least two of them should be a hands-on event, you gain;
Full-stack Master Geek

Architecture track:

After attending 5 events of various types:

- Architecture
- Methodology
- Software Development
- Cloud

You gain:
Architecture Geek
The badges has the levels as Full-stack track

Designer track:

After attending 5 events of various types:

- User Interaction
- User Experience
- Design

You gain:
Designer Geek

The badges has the same levels as the Full-stack track

Tester track:

After attending 3 events of various types:

- Test
- Quality & Assurance

You gain:

Tester Geek

The badges has the same levels as the Back-end track

Leadership track:

After attending 5 events of various types:

- Leadership
- Methodology
- Ethics

You gain:

Leadership Geek

The badges has the levels as Full-stack track

Foo Kids mentor

People that mentoring kids writing software will be able to get badges through following program:

First time mentor;

I teach kids programming

After 10 times;

Foo KIDS teacher

After 20 times;

Foo KIDS regular teacher

After 40 times;

Foo KIDS master teacher

Printed by Tryckeriet i E-huset, Lund 2017

LUND
UNIVERSITY

Series of Bachelor's theses
Department of Electrical and Information Technology
LU/LTH-EIT 2017-578

http://www.eit.lth.se