Proximity Marketing Application

Analysis and Design Document (Semester 1)

Shane Walsh

20079607

Supervisor: David Drohan

BSc (HONS) in Software Systems Development

Acknowledgements

I would like to take this opportunity to thank my supervisor David Drohan for all his advice and support throughout the semester. Without his insightful feedback and creative suggestions this project would not be possible.

Along with my fellow students, I would also like to thank my SSD lecturers and WIT staff for helping me along the way and making the journey as smooth as possible.

I would also like to gives special thanks to my girlfriend Aoife who always steered me in the right direction with positivity and kindness. And also my mother Josephine and sister Jen who were always there when I needed help.

Proximity Marketing App:

Analysis & Design

07.12.2017

Shane Walsh
20079607
Software Systems Development
Final Year Project - WIT

Table of Contents

1. Introduction

The purpose of this document is to present the research, analysis and design for the development of a web application. This project is a requirement of the WIT Software Systems Development Final Year programme.

From reading this document, it is hoped a clear understanding of the development process can be gained. Beginning with research and analysis, the project will be explained under the following headings:

- **Project:** Description, Inspiration, Scope, Target audience, User stories
- **Feasibility:** Market, Commercial, Competitors, USPs, Technical feasibility, Risk
- **Technologies:** Hardware, Software, Methodology
- **System Architecture:** Overview, User states, Flowchart, User authentication
- **Component Design:** BLE, NearBy, Voucherify, Webhose, Firebase, Analytics
- **Data Design:** Class diagram, Object schemas
- **UX Design:** Wireframes, Mockups
- **Schedule:** Timeline
- **References**
- **Appendices**

The intention of this project is to develop a progressive proximity marketing web application. A detailed explanation of the development process is outlined in the subsequent sections.

2. Project

This section will outline a description of the Proximity Marketing App (PMA), the inspiration behind the idea, along with the project scope, target audience and user stories.

2.1 Description

Using Bluetooth Low Energy (BLE) beacon technology, it is the intention of this project to design and develop a progressive web application that instantly provides shoppers with relevant information about products they're physically looking at - straight to their mobile device. This app will also allow users to review and compare products and take advantage of special offers through digital discount vouchers.

Combining the in-store experience, where you can pick up, touch and get the feel of a product, with the online experience where reviews and product insight are at your fingertips - the PMA hopes to bring the benefits of both worlds together.

82% of smartphone users say they consult their phones on purchases they're about to make in-store, making it vital for companies to identify and engage the consumer at the right time and in the right way.[1]

[1] https://www.unacast.com/post/proximity-marketing-what-how-why

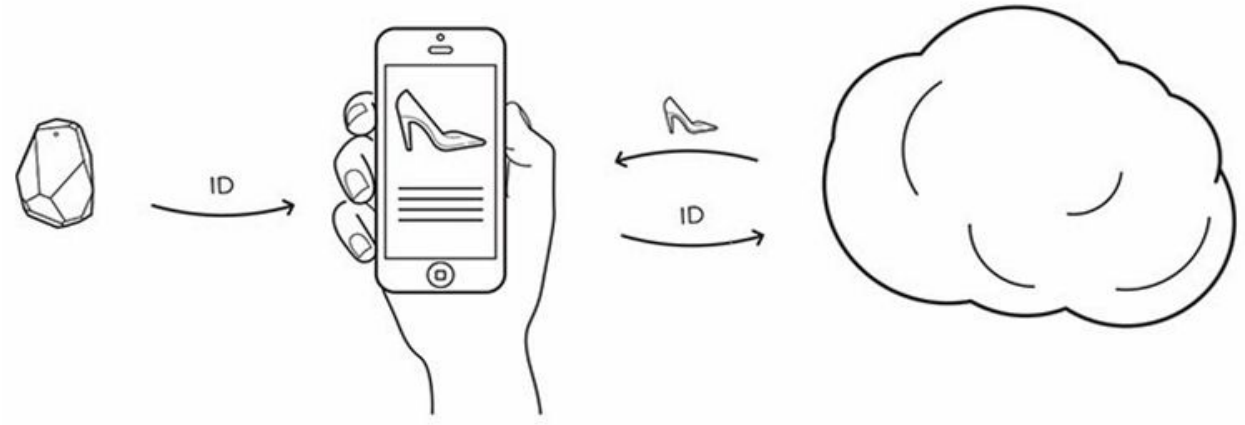


Fig 1. Beacon system overview

Take this typical scenario: A customer walks into a store and heads directly to a pair of shoes. Standing in front of the shoes she receives a browser notification on her smartphone. She clicks the link, upon which the PMA provides her with product info, reviews, special offers and a price comparison with similar products.

Customers increasingly expect to be pushed the right content at the right time based on the context, compared to the tedious process of diving into the phone to pull content out manually.[2] The PMA intends to deliver information that is timely, relevant and personal.

2.2 Inspiration

While visiting Amsterdam, I was very interested to find out the efforts made by local authorities to transform the capital into a smart city. It was recommended that I take a stroll down the 'Beacon Mile' for a real-time demonstration of innovation and smart city solutions. A route of up to 2km between Amsterdam's central station and the marina, the Beacon Mile is a public network of Bluetooth beacons offering digital tourist routes, public way findings and hyper local points of interest. From this, I got the idea to adapt beacon technology for the retail industry.

[2] Ibid.

2.3 Scope

The following table outlines the intended feature set of the PMA.

Feature Set Table

REF	Name	Description
FS-001	Product Review	View online reviews for the product
FS-002	Price Compare	Offers a price comparison with similar products online
FS-003	Vouchers	Send discounts and special offers to customers
FS-004	Analytics	Collect user data for brand owners

Fig 2. Feature Set Table

2.4 Target Audience

The target audience for this application are retail customers that like to make an informed decision using their smartphone before purchasing an in-store product. Location-based marketing is fast changing the way customers interact with brands around them. A 2014 consumer study conducted by beacon technology platform Swirl found that over 70% of shoppers who received beacon-triggered content and offers on their smartphone said it increased their likelihood to make a purchase during a store visit. More than 60% of respondents said they'd do more holiday shopping at brick-and-mortar stores that delivered mobile content and offers while they shopped, and 61% of people said they'd simply visit a store more often if they offered beacon marketing campaigns.[3] Figures like these would suggest customers are very receptive to the idea of brand owners pushing them content as long as it's timely and relevant. In other words, modern shoppers enjoy customized experiences and personalized recommendations and fit into the target audience for this application.

[3] https://www.shopify.com/retail/the-ultimate-guide-to-using-beacon-technology-for-retail-stores

2.5 User Stories

Persona 1:

Name: Carolyn Chappell

Age: 52

Location: Toulon, France

Work Life: Assistant Producer for French TV production company.

Scenario: Carolyn likes to shop online for clothes and jewellery because product reviews help steer her in the right direction in relation to quality. However, when it comes to buying footwear, she prefers to buy in-store and physically try on the shoes. Getting the correct shoe size is vital to having comfortable footwear. By using the PMA Carolyn can also access relevant online reviews about the shoes quickly.

Fig 3. Persona 1

Persona 2:

Name: Dr. Noah Andrews

Age: 38

Location: UK

Work Life: GP in busy local community clinic, along with three other doctors

Scenario: Dr. Andrews is a self-described 'gadget-head'. He loves to shop around for the latest electronics. So when he needed to buy a new TV, he went to the nearest electronics store to see actually *how* big a 52" HD plasma screen is in person. With the PMA, Dr Andrews was able to review the TV and price similar products on the market without having to dive deep into reviews on the net - it just popped up on his smartphone screen.

Fig 4. Persona 2

Persona 3:

Name: Eleanor Marsi

Age: 22

Location: Canada

Work Life: Medical Student

Scenario: As a student Eleanor loves to find a bargain. So whenever she is out shopping she makes sure to visit one specific store that offers personal digital discounts via coupons on her smartphone. She's not 100% sure how it works, all she knows is that she is guaranteed to save money.

Fig 5. Persona 3

Persona 4:

Name: Teddy Lawlor

Age: 46

Location: Ireland

Work Life: Construction

Scenario: Teddy likes to change his car and buy a new one each year. In previous years, he had to do meticulous research into the specs of each make and model. This year when he entered the showrooms - Teddy walked over to each vehicle, clicked the chrome notification and all the car specs popped up on his smartphone screen automatically.

Fig 6. Persona 4

3. Feasibility

This section will outline the feasibility of this project with regards to the current market and also look at market competitors along with the PMA's unique selling points.

3.1 Market

At the beginning of 2015, about 33% of the world wide web traffic came from mobile devices. Halfway through 2017, that number has risen to 52%. Half the world is accessing the web through their phones.[4] Given our primary target audience, shoppers with smartphones, this is a huge potential market.

It is important to stress at this point that the PMA will be a progressive web application and not a mobile application. This app will not be installed on a mobile device, instead it will live in the Cloud.

Major outlets and companies are adopting progressive web apps for their mobile web experiences, with two major examples being the *Washington Post* and Flipkart, India's largest ecommerce site.[5]

Retailers in particular have had a tough time getting consumers to download their apps, and that has limited the reach of beacons in malls and other retail environments.[6]

It is true that shoppers are increasingly moving toward their mobile devices for retail purposes, but the actual retailers have struggled to get customers to download their specific branded mobile apps, as the chart on the next page suggests.[7]

PMA intends to take a different approach by pushing Eddystone-URL browser notifications directly to the customer's phone, giving the user exposure to the app without a need for installation.

---

[4] Domes, Scott '*Insider Guide to Progressive Web Apps*', Web Designer, issue 266 p. 67
[5] https://www.oreilly.com/ideas/5-web-trends-for-2017
[6] https://marketingland.com/beacon-market-matures-marketers-drop-173651
[7] http://uk.businessinsider.com/shopping-app-usage-is-rising-but-retailers-still-have-a-glaring-problem-2016-6?r=US&IR=T

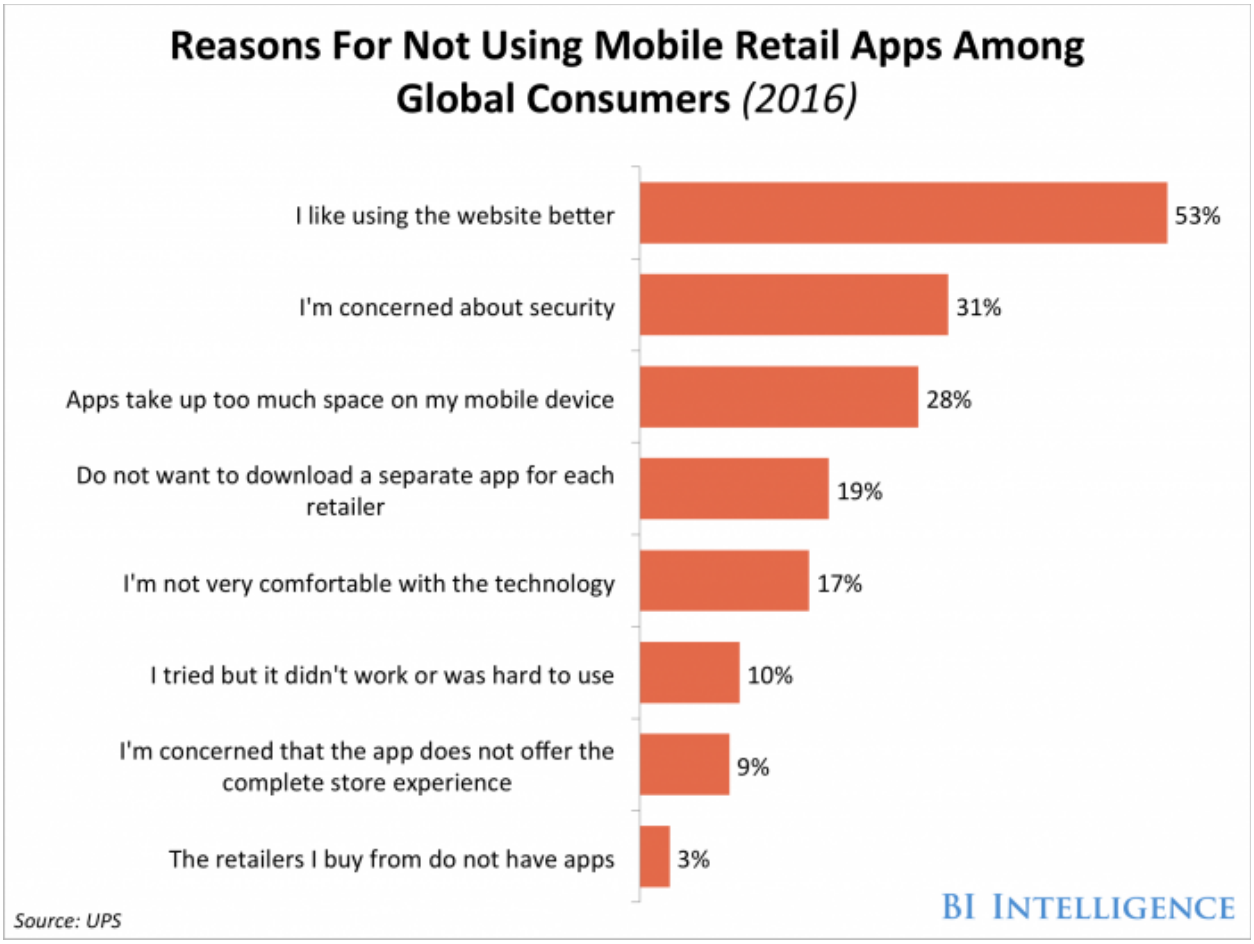


Fig. 7 Global consumer app usage

3.2 Commercial

In 5 years, over 70% of retail companies will use beacons.[8] With customers reluctant to download brand specific apps, beacon push notifications is certainly one solution. Speaking at a retail conference in Boston USA, veteran business journalist Deena M. Amato-McCoy comments that:

> *"By using beacon technology for retail to personalize the shopping experience, brick-and-mortar stores are driving customer engagement to turn shoppers into loyal brand advocates. Beacons are such a powerful engagement tool that 70 percent of companies plan to add the functionality over the next five years"*[9]

[8] https://www.onyxbeacon.com/7-major-trends-for-ble-beacons-in-2016/
[9] Ibid.

According to a Business Insider Intelligence report, a total of 4.5 million beacons are forecasted to be active in the U.S. by 2018. With such confidence for this technology in the business world, it would suggest to me that the PMA can be a commercial success.

3.3 Competitors

There are two major competitors in this market:

1. Beacon Stac

   https://www.beaconstac.com

2. Beacon Stream

   http://www.beaconstream.com

Both offer similar packages in that they run proximity marketing campaigns for marketers and business owners. Both these companies sell the physical beacons and configure them for their clients to suit their business needs.

Competitor's target audience:

- Retailer stores & supermarkets
- Bars and restaurants
- Amusement parks
- Dealerships

3.4 Unique Selling Points

Having investigated competitors in the proximity beacon market, two unique selling points have been identified for the PMA.

1. Product review feature.
2. Price comparison feature.

While competitors target customers using voucher redemption systems and are more focused towards marketing campaigns, it is the intention of the PMA to offer the user extended functionality in the form of a product review and price comparison feature as outlined in the above Feature Set table: FS-001 & FS-002.

3.5 Technical Feasibility

From analysis and research, it has been found that this project is indeed technically feasible. The proposed technologies for this project have been used to develop similar type applications in the recent past, giving the developer a positive indication that all technologies mentioned harmonise and function well together. A prototype will be developed to demonstrate the application's performance.

3.6 Risk Assessment

While there is always a certain degree of risk associated with software projects, it is hoped that a combination of strong design, in-depth analysis and a suitable methodology can counteract any potential development pitfalls.

4. Technologies

This section will outline *what* technology will be used to develop the Proximity Marketing App and a justification for their usage - which will include looking at alternative technologies.

4.1 Hardware

Bluetooth Beacons

Fig 8. Kontakt beacon

This project will utilize three Kontakt Bluetooth Low Energy (BLE) beacons. BLE beacons are essentially hardware transmitters that broadcast unique identifiers to nearby portable electronic devices. This technology enables smartphones, tablets and other devices to perform actions when in close proximity to a beacon.

Why choose Kontakt beacons?

BLE vendors are many and varied, however Kontakt support both iBeacon and Eddystone protocols, offer extended API functionality and an easy to use SDK for

integrating functionality into existing apps. Kontakt also offer reasonable pricing for their beacons along with great customer service.

A more detailed description of Bluetooth beacons and how they will work in the context of the PMA will be given later in the Component Design section.

4.2 Software

React.js

React is an open-source JavaScript library used for handling the view layer of web and mobile apps. It allows for the creation of reusable UI components.

Why use React?

The main purpose of React is to be fast, scalable and relatively easy to learn given prior knowledge of JavaScript ES6. As a student developer, it is my intention to broaden my horizons regarding front-end development frameworks. Having experience using Angular I wish to expand that knowledge base and use React for this project in order to add another 'bow to my string' so to speak.[10]

[10] See Section 5 for a more detailed discussion on React.js.

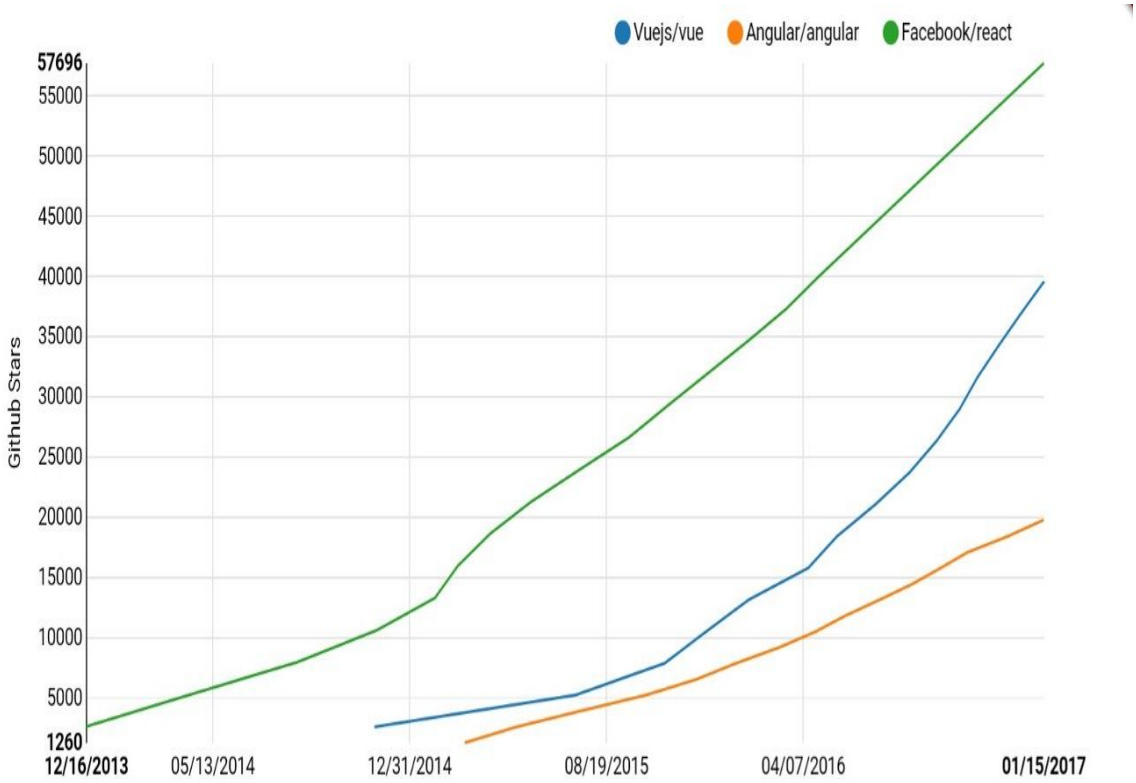


Fig.9 Github stats: front-end frameworks

React certainly is a favourite with big business dealing with data heavy server-side rendering. Facebook, Instagram, Netflix and Yahoo all use React components. The figure above shows React's popularity on Github in relation to other prominent front-end frameworks. Vue is another popular progressive framework for building user interfaces. It is similar in ways to React, both use virtual DOM's and both are solutions to the same problem. However, React fully embraces JavaScript and Vue does not. Using Vue templates creates scoping issues, whereas with React all the standard JavaScript scoping rules apply.[11]

Benefits of using React include:

- Components - UI is made up of reusable components.
- One-Way Data Flow - allows for better scalability and debugging.
- Virtual DOM - more efficient UI update.

[11] https://medium.com/@CalinLeafshade/why-i-chose-react-over-vue-3dd9a230b507

- Ecosystem - great support community

Stack-overflow illustrates the rapid growth and decline of other web frameworks over the last few years in the graph below.

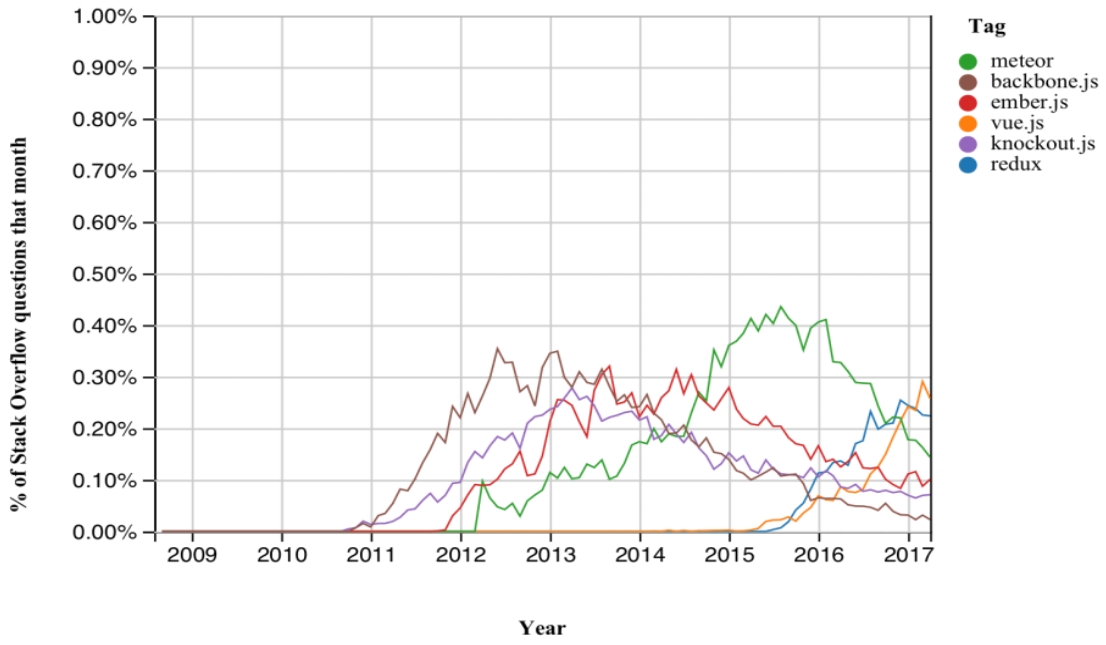


Fig. 10 Web UI Frameworks

Firebase

Firebase is a web application development platform supported by Google. It will form the backbone of the PMA as it will be used to provide server-side functionality, real-time data storage and deployment. A more in-depth analysis of Firebase will be covered later in the Component Design section.

Why use Firebase?

- Real-time database
- Built-in authentication process
- Large data storage capacity
- Notifications
- Free to use

Other platforms such as Amazon Web Services were considered to provide server-side support for the PMA. However, seen as several components of the app will use Google technologies (Eddystone, NearBy, Analytics) it was decided that Firebase, also a Google technology, would be the most suited and help minimise the risk of conflict and data communication problems.

Google NearBy Notifications API

The NearBy API enables a web site or web app to be associated with a BLE beacon. Essentially, NearBy facilitates a notification to be pushed to a web browser on a mobile device. A URL opens in the web browser when the user clicks the notification.

Why use NearBy?

- No app install required
- Supports Eddystone
- Addition tags for analytics

Webhose API

Webhose offers two API's that are critical to the feature set of this application.

1. Reviews API - returns online reviews. It will allow the PMA to get comprehensive, up-to-the-minute access to millions of online reviews from thousands of sources. This API will be used to implement the product review component of the PMA feature set.
2. eCommerce Product Data API - get access to a database of millions of products from thousands of online retailers and e-commerce sites.[12] This API will be used to implement the price comparison component of the PMA feature set.

[12] https://docs.webhose.io/docs/ecommerce-product-data-api

Why use Webhose?

- All data returned from this API will be in JSON format, which is ideal seen as this app will be written entirely in JavaScript.
- Free to use RESTful API token.
- After investigating similar alternative technologies such as Import.io and Parsehub, it was clear Webhose offered a more extensive service with better functionality and more in depth documentation and support.

Google Analytics (analytics.js)

The Google Analytics Platform enables the measuring of user interactions with web applications across various devices and environments. It provides all the computing resources to collect, store, process, and report on these user-interactions.[13] For the purposes of the PMA, analytics.js will be used, which is a JavaScript tracking snippet that allows for the capture of user data, such as: total time a user spends on the app, what internal links were clicked, geolocation and more.

Why use Google Analytics?

While there are many popular and capable analytics application available, such as Moz Pro, Piwik and KISSmetrics, it was decided to incorporate Google Analytics for the following reasons:

- Insight into user behaviour.
- Automated data collections and integration with other tools.
- Extensive documentation and Google support community.
- It's relatively easy and free to implement.[14]

[13] https://cloud.google.com/appengine/docs/standard/python/google-analytics
[14] https://developers.google.com/analytics/devguides/collection/analyticsjs/how-analyticsjs-works

Voucherify (voucherify.js)

Voucherify is a cloud-based API that helps integrate loyalty and discount functionality into web applications. This platform gives total flexibility in the way coupon and loyalty systems are designed, implemented and maintained.

Why use Voucherify?

- The Voucherify RESTful API allows a voucher redemption system to be plugged into any web application, in this case the PMA.
- Extensive support documentation.
- API integration.

4.3 Methodology - Agile SCRUM

For this project an Agile SCRUM approach will be applied towards development and design. Scrum is a subset of Agile and is generally geared towards development teams with no less than 5 or 6 members. As this is an individual project, it is the intention of the developer to extract the most relevant and appropriate components of SCRUM and apply them to the development process. After considering other methodologies such as the Waterfall model, it was decided that agile development methods are best suited to this project.[15]

[15] For further discussion on agile application & alternative methodologies see appendix A

5. System Architecture

This section gives a high level overview of the PMA system and its interconnected components. Also discussed are the user states and the architectural design.

5.1 Systems Overview Diagram

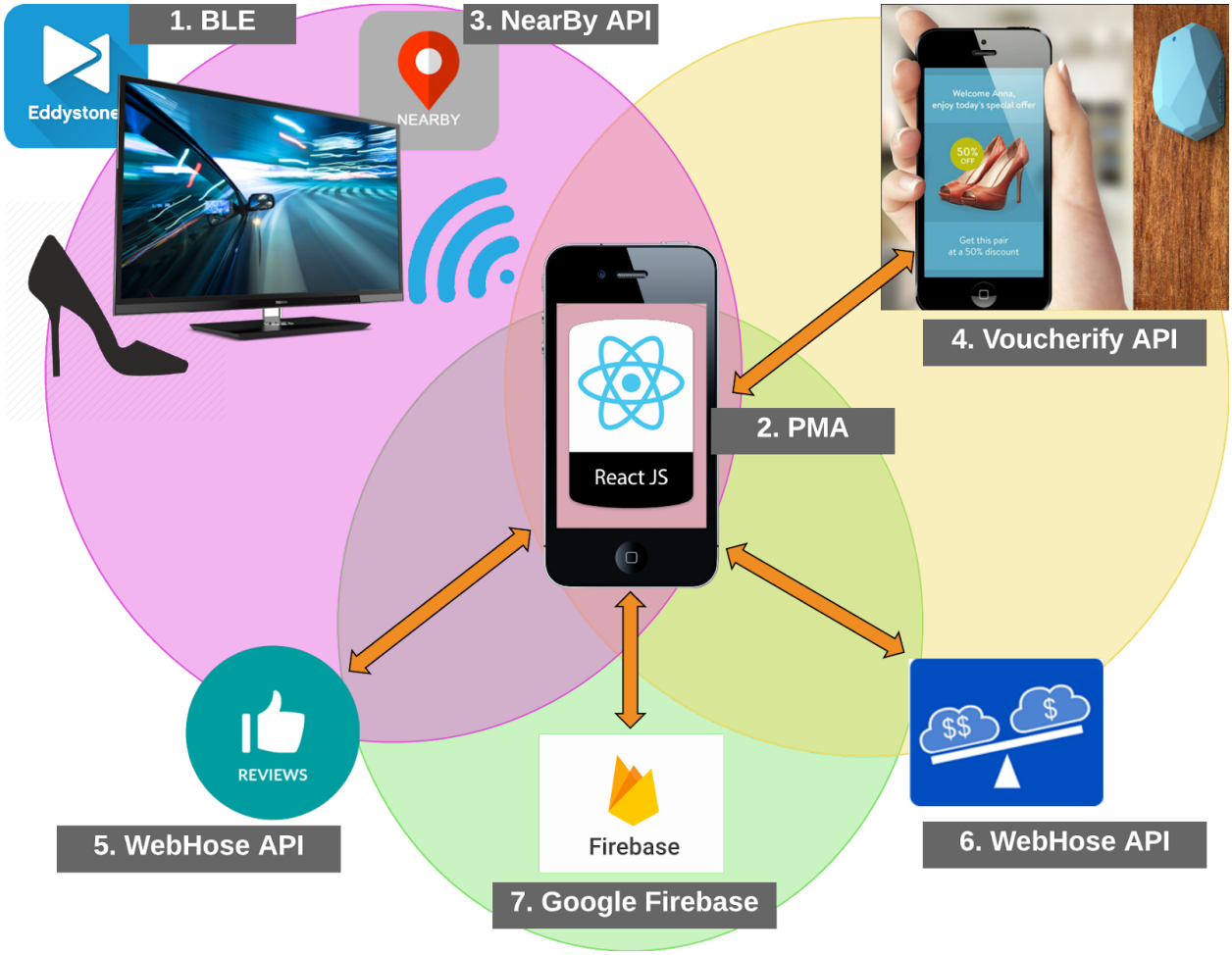


Fig. 11 Systems Overview Diagram

(1) BLE - Eddystone formatted Bluetooth low energy beacon will send a push notification to mobile device web browser. User clicks on notification to be taken to the PMA.

(2) PMA - It is the intention of the developer to make this ReactJS web application as progressive as possible, meaning:

- Works for all users regardless of browser choice
- Fully responsive - adapts to desktop, mobile, tablet
- Connectivity independent - enhanced with service workers to work offline or on low quality networks.
- App-shell model to provide app-style navigation and interactions.
- Secure - served via HTTPS to enhance security

(3) Google NearBy API enables the beacon to be associated with the web app.

(4) The Voucherify API allows users to redeem digital vouchers.

(5) (6) Webhose API enables users to review products online, along with a price comparison.

(7) The app will be deployed and persistence maintained using Google's Firebase platform.

PLEASE NOTE: The integration of the above technologies (1-7) will be discussed at greater length in the next section Component Design.

5.2 User States

Two user states exist for the PMA

1. Register
2. Unregistered

Registered

A registered user will have access to the complete functionality of the PMA, which includes:

- Product Review
- Price Comparison
- Digital Vouchers

Unregistered

Unregistered users will have limited access to PMA functionality, which includes:

- Product Review
- Price Comparison

Below a flow chart illustrates the steps a user takes using the PMA - followed by a sequence diagram outlining the authentication process:

5.3 Flow Chart

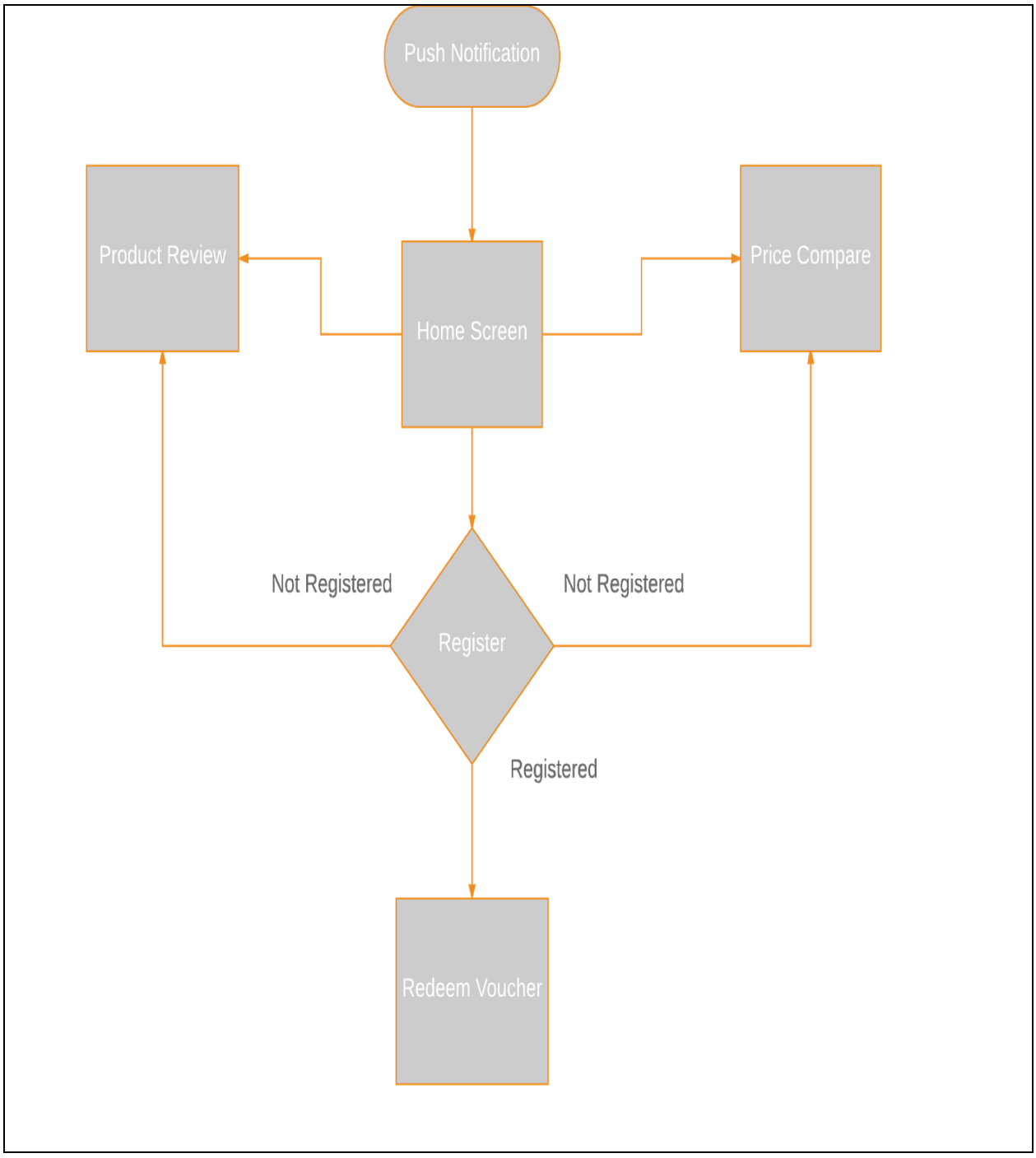


Fig. 12 User Flow Chart

5.4 Sequence Diagram: User Authentication

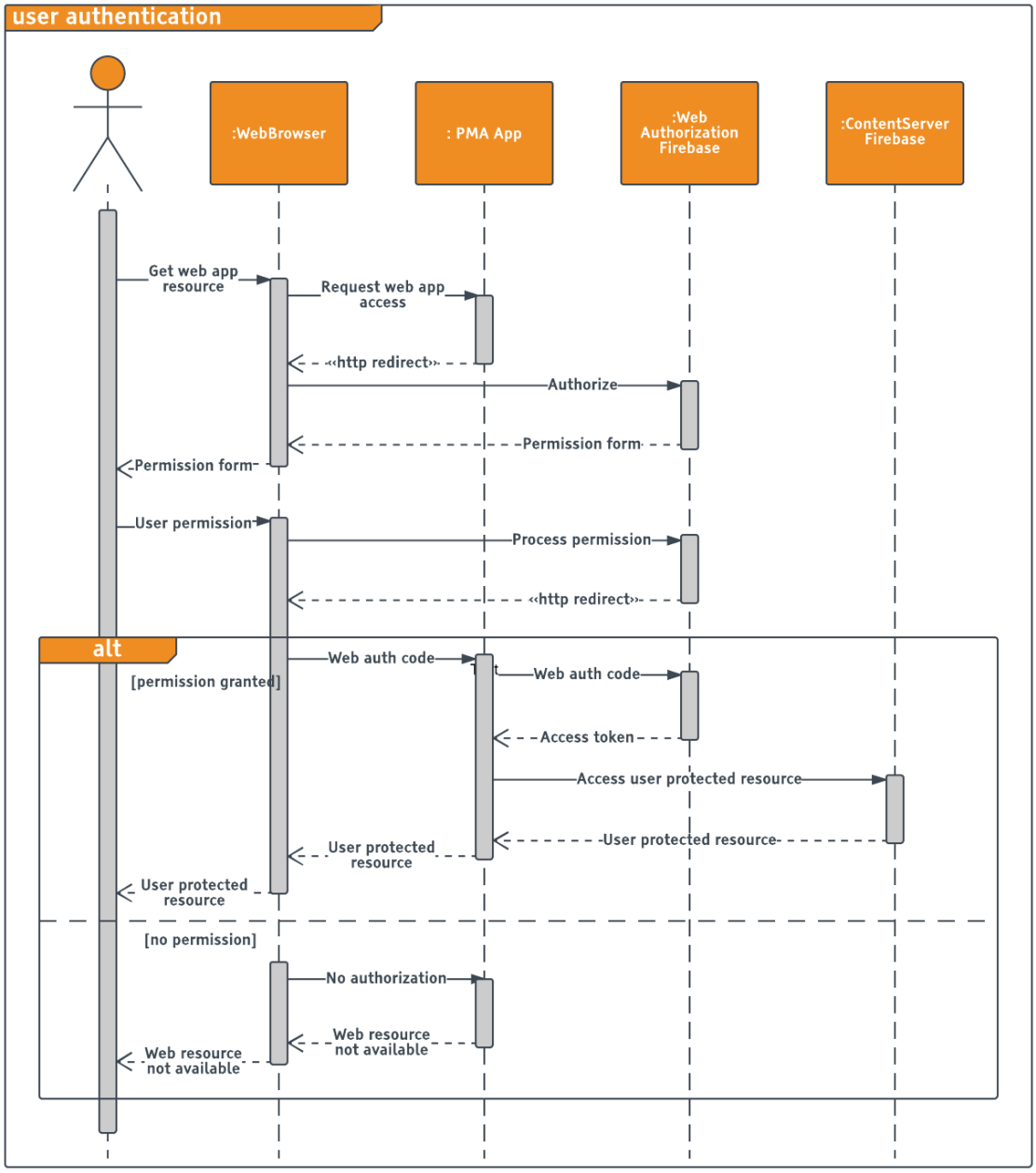


Fig. 13 User Authentication Sequence Diagram

5.5 Architectural Design

Flux

Flux is an architecture developed by Facebook as a solution to problems encountered using the traditional MVC design pattern. As the PMA will use React.js to render the front-end presentation layer, Flux will be employed to complement React by using a unidirectional data flow design pattern.

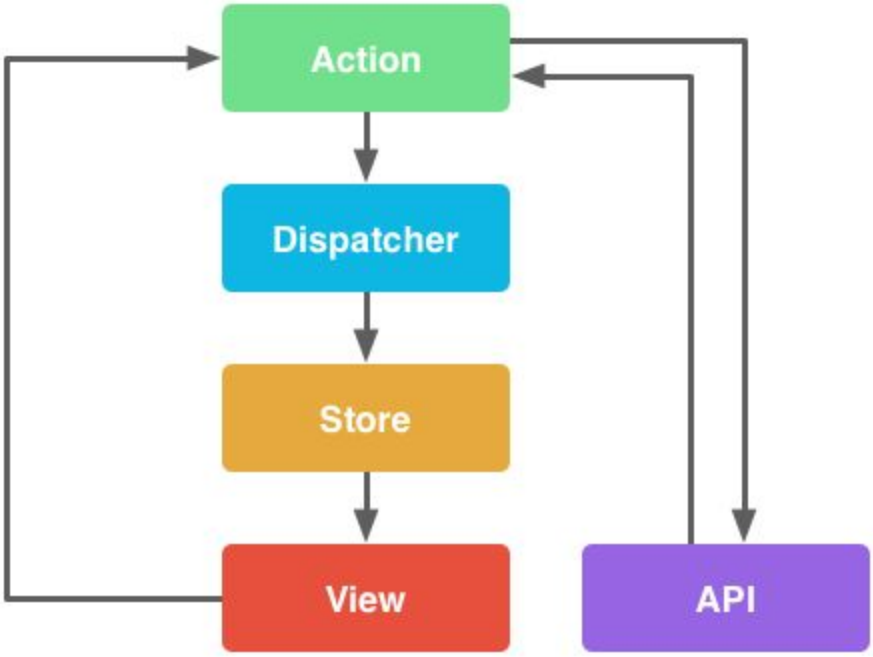


Fig. 14 User Authentication Sequence

- Actions - are methods that pass data to the Dispatcher
- Dispatcher - acts as a control centre receiving and broadcasting payloads to callbacks
- Stores - a collection of data storing the UI state to be used by the dispatcher
- Controller Views - UI components

6. Component Design

This section gives a detailed account of all the technologies that will be used to create the PMA and how they will work together to deliver a high value application.

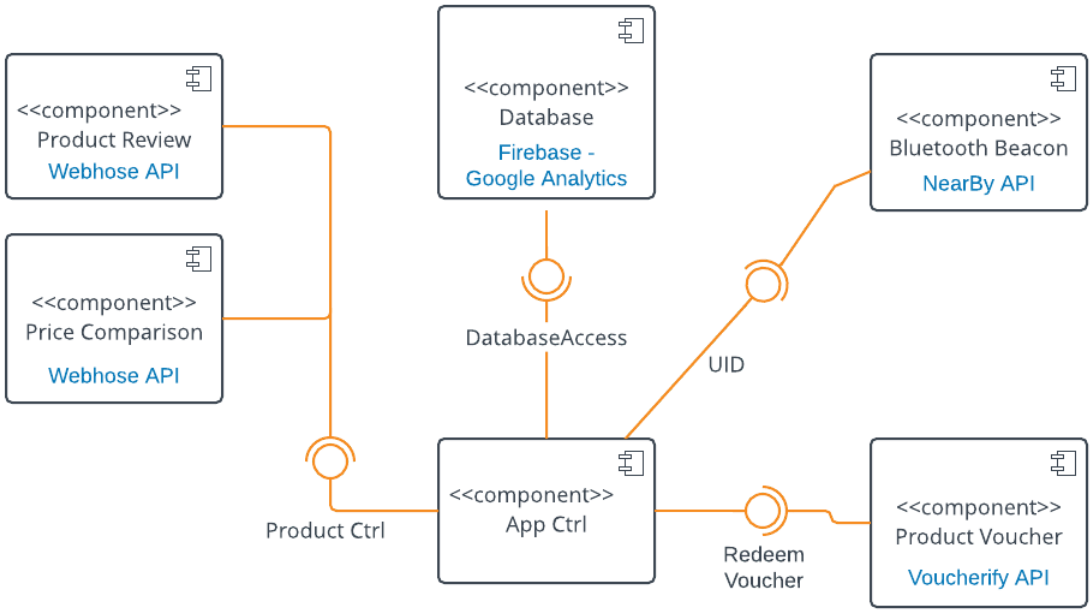


Fig. 15 Component Overview Diagram

6.1 Bluetooth Low Energy Beacon (BLE)

How will beacons work with the PMA?

Each beacon is assigned to a single product. That beacon is then broadcasting or pushing a browser notification to nearby mobile devices. Within that notification is a URL specific to that product. The user clicks the notification to activate the web application, which in turn retrieves the relevant product information from the Firebase servers and displays it onscreen. No app installation needed.

What is Eddystone and iBeacon?

Eddystone and iBeacon are communication protocols. However, the PMA will be using Google's Eddystone for the reasons outlined below.

The Eddystone protocol can transmit 3 different frame-types:

- Eddystone-UID (Unique Identification)
- Eddystone-URL
- Eddystone-TML (Telemetry: data from sensors)

Fig. 16 User Authentication Sequence

Important for this project is the Eddystone-URL which allows for a specific application URL to be pushed to a user's mobile device, in this case our PMA URL.

Attached to the URL will be a unique ID, as show above in Fig. 16. This UID tells us what beacon to look for on our servers and which product is associated with that particular beacon.

```
var BeaconSchema = new mongoose.Schema({
    uid:{ type: Number, required: true},
    name: { type: String, required: true, lowercase: true},
    venue: { type: String, required: true},
    platform: { type: String, required: true},
    active: { type: Boolean, required: true},
    product: { type: Number, required: true},
    date: { type: Date, default: Date.now }
});
```

Fig. 17 Beacon Schema UID and Product relationship

Eddystone is a cross-platform protocol which means it not only works well with iOS and Android, but any processor that supports Bluetooth beacons.

iBeacon does not offer the same functionality as it only transmits a UID. This means iBeacon requires an installed app to receive, process and/or track beacons.[16]

6.2 Google NearBy API

Nearby Notifications allows the PMA to be associated with a BLE beacon. It provides the ability to create contextual notifications from nearby Bluetooth technology - even with no app installed.

How will NearBy work?

Nearby uses low-power Bluetooth to detect broadcasts around a mobile device. Users in the proximity of a bluetooth beacon will see a message pop-up in the notifications section of Chrome or whatever NearBy ready browser they are using. Contained in the message will be a link to the PMA.

[16] https://kontakt.io/blog/extensive-guide-to-bluetooth-beacons/

6.3 PMA Architecture Overview

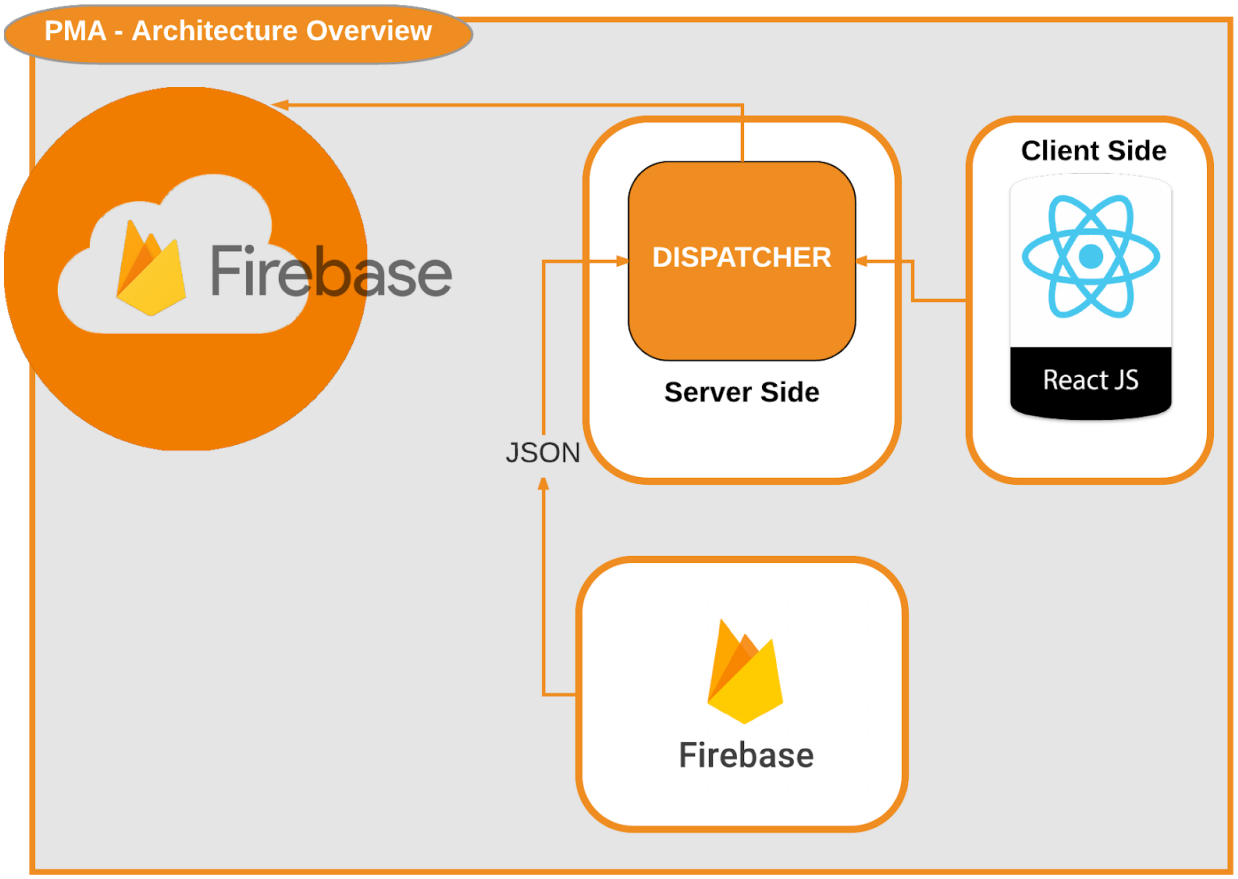


Fig. 18 Architecture Overview

As shown in Fig. 18 above the PMA will use React to implement our 'View' or presentation layer, with Firebase acting as the 'backbone' on the server side and also adding persistence and hosting services.

HTML5 and CSS3

It is important to keep in mind that this web application will be predominantly used on mobile devices, so it needs to be fully responsive to cater for all screens. All PMA pages will be written in fully responsive HTML5 and CSS3 to cater for various size mobile devices, tablets, phones etc.

React.js

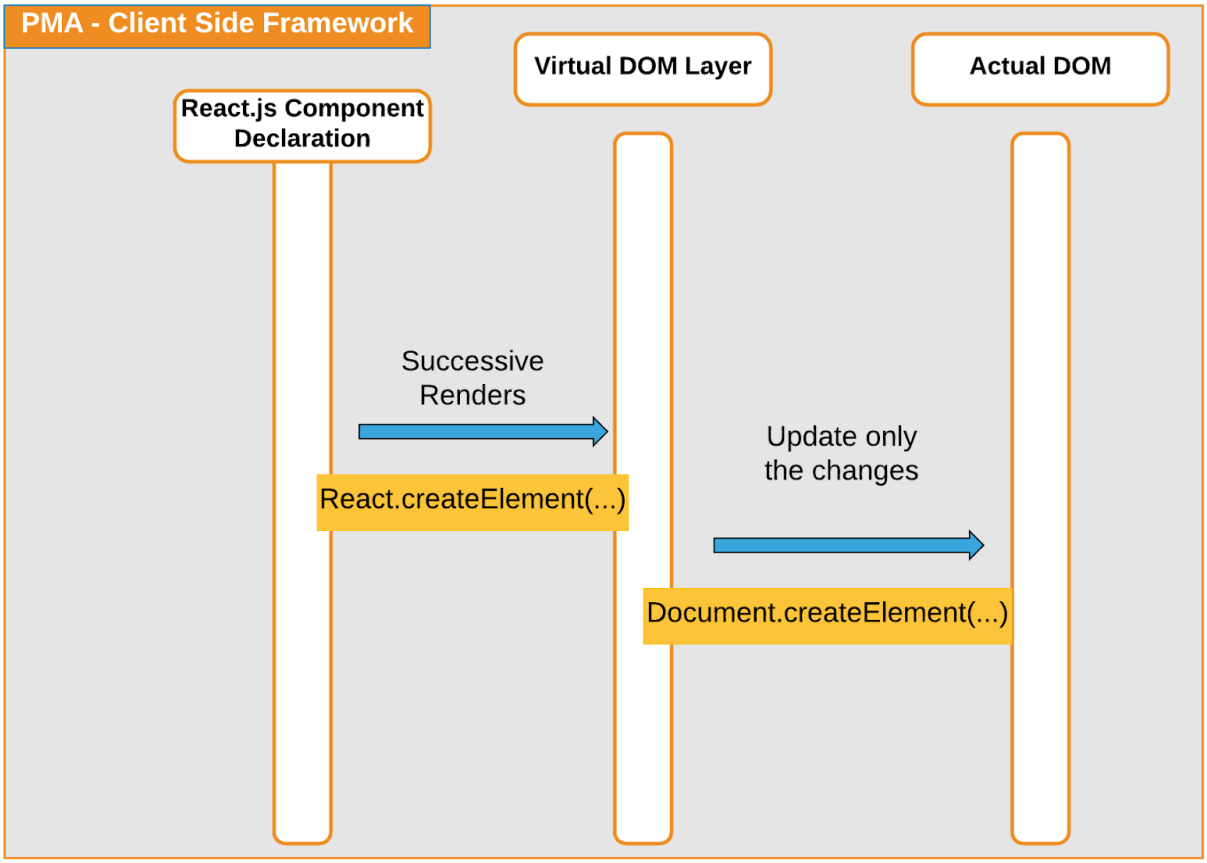


Fig. 19 React Client Side Virtual DOM

React's use of a virtual DOM will help render the PMA pages rapidly. React's render() method updates the DOM in response to changes in the data model. Every time the underlying data changes in the PMA - React checks for differences against the Virtual DOM and so only the changes are rendered.

React follows a 3 step process that will handle our application's view layer - as follows:

- If anything changes on the client-side are made, the entire UI will be re-rendered in the Virtual DOM.

- The differences between the previous Virtual DOM and the new one will be calculated. This is called 'diffing'.
- The real DOM will be updated with only the new changes.

This process will help render the PMA front-end quickly because the entire actual DOM is not re-rendered every time there is a minor change in the UI.

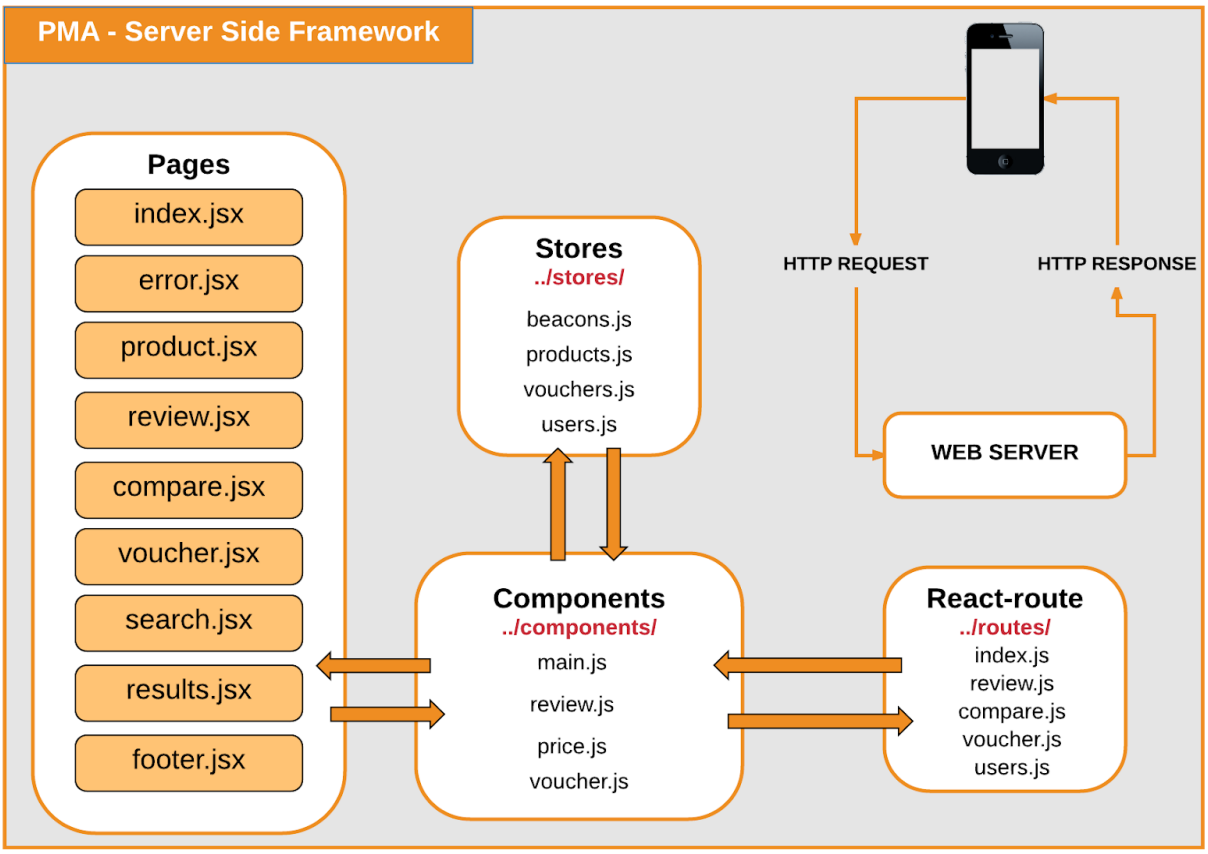


Fig. 20 Server Side Framework

Fig. 20 above shows the integration of components within the PMA. React-router is the standard routing library for React. It will help keep the PMA UI in sync with other application components. The stores are containers for the logic states to be used by the application dispatcher.

6.4 Voucherify API

The diagrams below illustrate the voucher redemption process in UML format.

Redeem Voucher Use Case Diagram

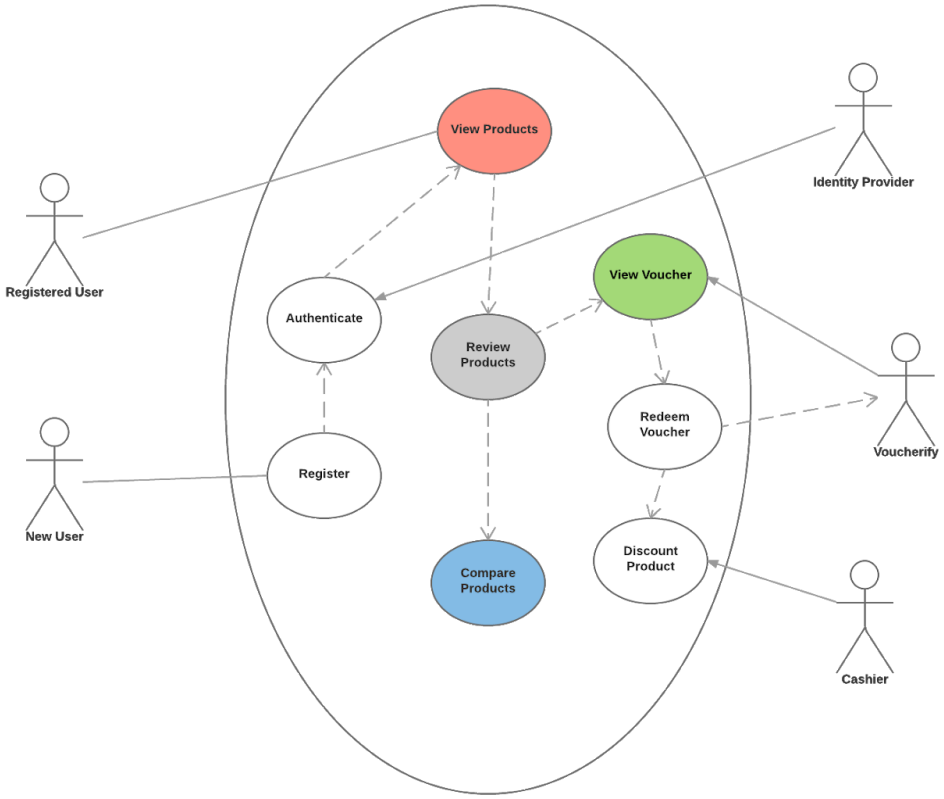


Fig. 21 Redeem Voucher Use Case Diagram

Redeem Voucher Sequence Diagram

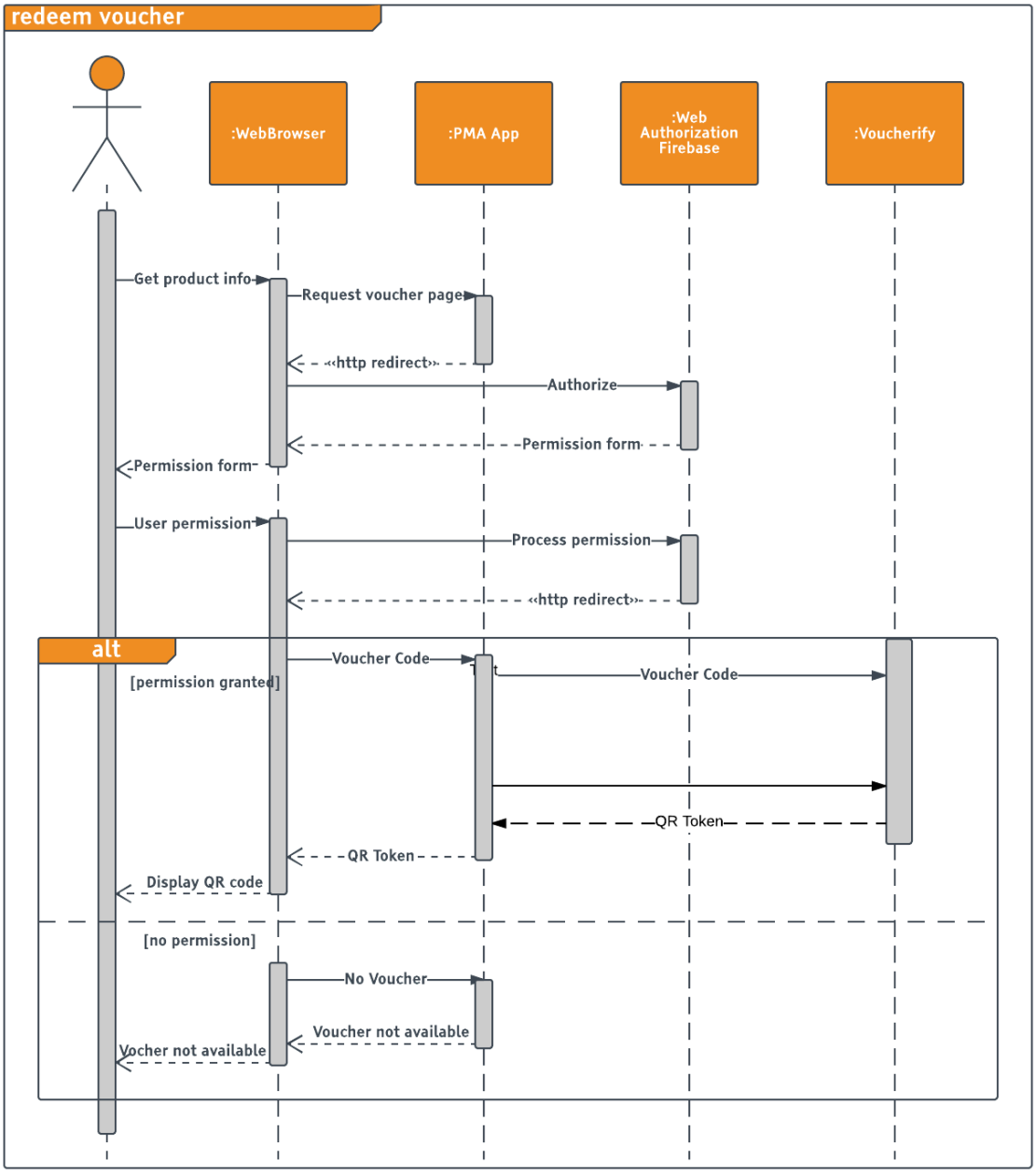


Fig. 22 Redeem Voucher Sequence Diagram

How will vouchers work in the context of the PMA?

Vouchers will be assigned to products based on a loyalty system. If a registered user interacts with a product via the PMA on more than 5 occasions, the voucher will be issued as a reward and as an incentive to make the purchase. The PMA will keep track of the amount of times a user interacts with a product using a visit_counter variable (see below) and automatically trigger the voucher allocation when the count is greater than 5.

```
//User Schema
var mongoose = require('mongoose');

var UserSchema = new mongoose.Schema({

    firstname: { type: String, required: true, lowercase: true},
    lastname: { type: String, required: true, lowercase: true},
    email: { type: String, required: true},
    visit_count: { type: Number, required: true},
    passwordHash: { type: String, required: true},
    passwordSalt: { type: String, required: true}
});

module.exports = mongoose.model('User', UserSchema);
```

Fig. 23 visit_counter in User Schema

To redeem a voucher, the user taps the 'Redeem Voucher' button at which point a QR code is displayed on screen. The user can then take the code to the cashier's desk, where the assistant scans the QR Code - implementing the product discount.

Technical Implementation

First we obtain the API keys from the Voucherify.io web interface.[17] The API keys are implemented below:

```
var voucherifyClient = require('voucherify')

var client = voucherifyClient({
   applicationId: 'YOUR-APPLICATION-ID',
   clientSecretKey: 'YOUR-CLIENT-SECRET-KEY'
})
```

We can now use the Voucherify modules within our application. It is the intention of the developer to utilise JavaScript promises instead of callbacks throughout this application, mainly to avoid 'callback hell' (waiting for one async request to finish before performing another async request) and also to make the app code more readable. A Voucherify interaction can be written as a promise - as seen below:

```
client.vouchers.get('vExample')
   .then((result) => {
      console.log(result)
   })
   .catch((error) => {
      console.error("Error: %s", error)
   })
```

Common voucher methods are listed below. These methods will handle the majority of actions concerning the voucher object - including:

```
client.vouchers.create(voucher)
```

[17] All Voucherify documentation is found at www.voucherify.io

```
client.vouchers.get(code)
client.vouchers.update(voucher)
client.vouchers.delete(code)
client.vouchers.list(params)
client.vouchers.enable(code)
client.vouchers.disable(code)
```

6.5 Webhose API

Technical Implementation

To use the WebHose.io API for product reviews and price comparison it is necessary to register with Webhose in order to obtain an API key.[18] Now we can import the Webhose library and set the access token with the API key:

```
var webhoseio = require('webhoseio');

var client = webhoseio.config({token: 'YOUR_API_KEY'});
```

There are two primary API endpoints required for the PMA:

1. `filterWebContent` - gives access to the reviews API
2. `productFilter` - gives access to the eCommerce price comparison API[19]

[18] All documentation can be found at www.webhose.io
See use case & sequence diagrams for product review and price compare in appendix B + C
[19] For an example of an endpoint returning JSON - see appendix D

6.6 Google Firebase

The PMA will be hosted and deployed on Google's Firebase servers which will add a layer of persistence to our application. In other words, Firebase will be used to handle all of our data in terms of:

- Storage
- Synchronization
- User authentication

Technical Implementation

Before integrating the PMA with Firebase, it is necessary to register and obtain an API. To use the module in our application, we simply 'require' it like so:

```
var firebase = require("firebase");
```

Next we initialise the Firebase SDK in our app and configure using our API key.[20]

The PMA can now use multiple Firebase services. Each service can be accessed as shown below:

`firebase.storage()` - Cloud Storage
`firebase.auth()` - Authentication
`firebase.database()` - Realtime Database
`firebase.firestore()` - Cloud Firestore

[20] See example in appendix E

Storage

In terms of storage, PMA data will be stored in Google Cloud Storage buckets. Each bucket is a logical unit of measurement used by Firebase to store data. The data storage service can be referenced by the PMA as follows:

```
var storage = firebase.storage();
```

Database

Firebase Realtime Database is where the Cloud Storage buckets are hosted. It is a NoSQL database in which all data is stored as JSON. The database is initialised in the PMA config file and can be referenced as follows:[21]

```
// Get a reference to the database service

var database = firebase.database();
```

Authtication

Firebase will also be used to handle authentication as if offers backend services to leverage industry standards like OAuth 2.0 and OpenID Connect .

User credentials can be an email address and password, or an OAuth token - which is then passed to the Firebase Authentication SDK for verification, before returning a response to the client.

Sign-in via other identity providers, such as Facebook, Twitter and Github will also be implemented for PMA users.[22]

[21] For a sample of what read and write operations for the PMA may look like see appendix F
[22] Full documentation can be found at: https://firebase.google.com/docs/auth/web/start

Deployment

Firebase will also be used as a hosting platform for the PMA. In order to deploy - it is necessary to first install firebase-tools globally:

```
$ install -g firebase-tools
```

From the PMA base folder execute the following code to initialise the application:

```
$ firebase init
```

To deploy the PMA we simply run:

```
$ firebase deploy
```

Our app will be deployed to the following domain, or something similar:

```
proximitymarketing.firebaseapp.com
```

6.7 Google Analytics

To use Google Analytics it is first necessary to create an account and register with Google. For analytics and web-tracking the PMA will use the analytics.js library. This is a JavaScript library for tracking user interactions with the PMA.

The JavaScript tracking snippet consists of the following code and must be inserted inside the <head> tag of the app's index page and before any other script files or CSS. The following code is available on Google's developer website.[23]

```
<!-- Google Analytics -->

<script>

(function(i,s,o,g,r,a,m){i['GoogleAnalyticsObject']=r;i[r]=i[r]||functio
n(){

(i[r].q=i[r].q||[]).push(arguments)},i[r].l=1*new
Date();a=s.createElement(o),

m=s.getElementsByTagName(o)[0];a.async=1;a.src=g;m.parentNode.insertBefo
re(a,m)

})(window,document,'script','https://www.google-analytics.com/analytics.
js','ga');

ga('create', 'UA-XXXXX-Y', 'auto');

ga('send', 'pageview');

</script>

<!-- End Google Analytics -->
```

[23] https://developers.google.com/analytics/devguides/collection/analyticsjs/

The code above will collect user data and send it back to the PMA Google Account. The information gathered on users and the application's performance can inform product owners on what products are hot, and what are not - and potentially help them to make smarter data-driven business decisions.

7. Data Design

This section outlines the data types, objects and structures required to handle the PMA's data storage needs. Also shown is a class diagram, highlighting entity relationships along with a list of object schemas.

7. 1 Data Type Tables

Beacon_Object

Field Name	Data Type	Required	Description
beacon_id	Number	Yes	Unique ID
name	String	Yes	Unique name
venue	String	Yes	Identifies beacon location
platform	String	Yes	Eddystone || ibeacon
active	Boolean	Yes	Set to active or dormant
voucher	Number	Yes	Voucher_id
product	Number	Yes	Product_id
date	Date	Yes	Date beacon activated

Product_Object

Field Name	Data Type	Required	Description
product_id	Number	Yes	Unique ID
brand	String	Yes	Product brand name
type	String	Yes	Type of product
description	String	Yes	Description of product
price	Number	Yes	Cost of product
designation	Number	Yes	Beacon_id
date	Date	Yes	Date product activated

Voucher_Object

Field Name	Data Type	Required	Description
voucher_id	Number	Yes	Unique ID
amount	Number	Yes	Discount rate
order_id	Number	Yes	Product order_id

Order_Object

Field Name	Data Type	Required	Description
order_id	Number	Yes	Identifies order
product _id	Number	Yes	Identifies product
voucher_id	Number	Yes	Unique ID
user_id	Number	Yes	Identifies user
beacon_id	Number	Yes	Discount rate
quantity	Number	Yes	Quantity of products ordered
order_date	Date	Yes	Date ordered

User_Object

Field Name	Data Type	Required	Description
user_id	Number	Yes	Unique ID
firstname	String	Yes	User's name
lastname	String	Yes	User's name
email	String	Yes	User's email
visit_count	Number	Yes	PMA usage
passwordHash	String	Yes	Authentication
passwordSalt	String	Yes	Authentication

Fig. 24 Data Type Table

7.2 Class Diagram and Entity Relationship

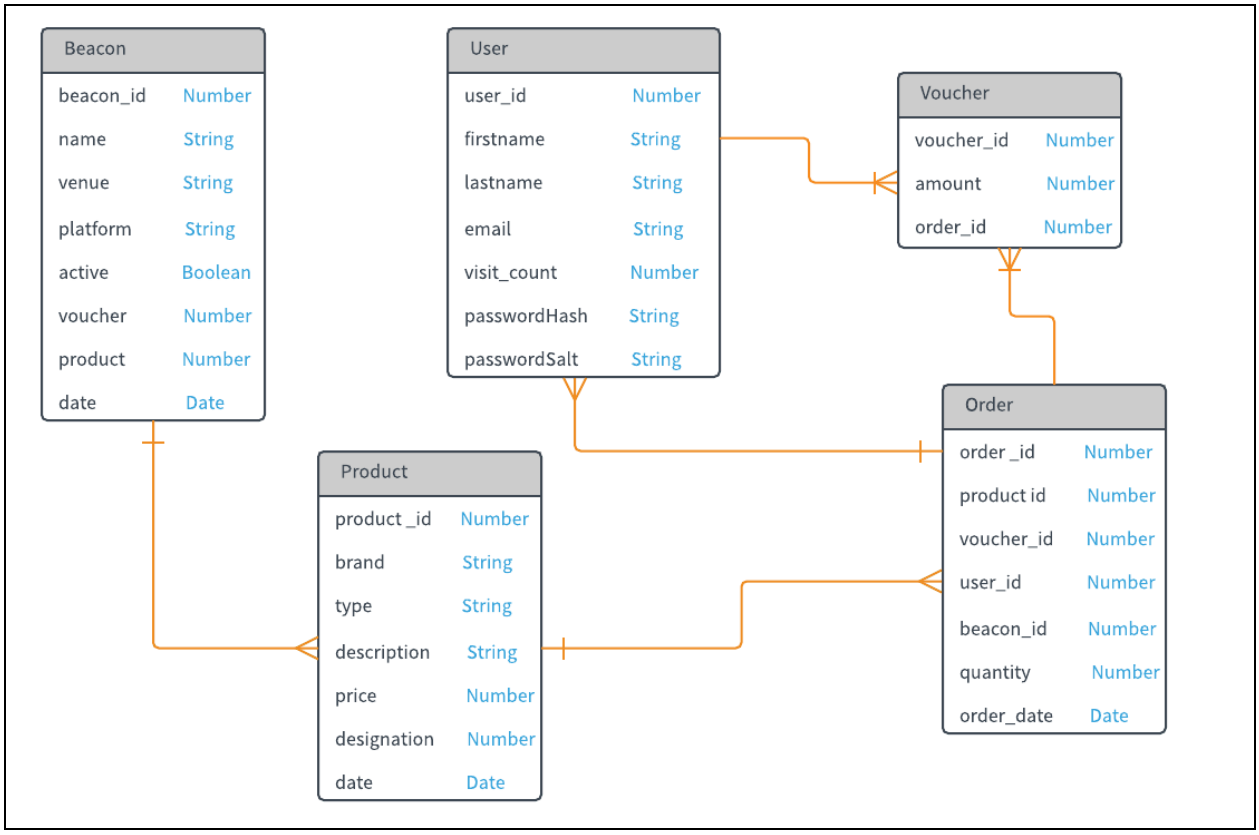


Fig. 25 Class Diagram + Entity Relationship

7.3 Object Schemas

The following are a list of schemas that define the structure of the objects to be used by the PMA:

Beacon_object

```
//Beacon Schema
var mongoose = require('mongoose');

var BeaconSchema = new mongoose.Schema({
    uid:{ type: Number, required: true},
    name: { type: String, required: true, lowercase: true},
    venue: { type: String, required: true},
    platform: { type: String, required: true},
    active: { type: Boolean, required: true},
    voucher: { type: Number, required: true},
    product: { type: Number, required: true}.
    date: { type: Date, default: Date.now }

});

module.exports = mongoose.model('Beacon', BeaconSchema);
```

Fig. 26 Beacon Schema

Product_object

```
//Product Schema
var mongoose = require('mongoose');

var ProductSchema = new mongoose.Schema({
    brand: { type: String, required: true},
    type: { type: String, required: true},
    description: { type: String, required: true},
    price: { type: Number, required: true},
    designation: { type: String, required: true, lowercase: true},
    date: { type: Date, default: Date.now }
});

module.exports = mongoose.model('Product', ProductSchema);
```

Fig. 27 Product Schema

Voucher_object

```
//Voucher Schema
var mongoose = require('mongoose');

var VoucherSchema = new mongoose.Schema({

    amount: { type: Number, required: true},
    order_id: { type: Number, required: true},
});

module.exports = mongoose.model('Voucher', VoucherSchema);
```

Fig. 28 Voucher Schema

Order_object

```
//Order Schema
var mongoose = require('mongoose');

var OrderSchema = new mongoose.Schema({
    product_id: { type: Number, required: true},
    voucher_id: { type: Number, required: true},
    user_id: { type: Number, required: true},
    beacon_id: { type: Number, required: true},
    quantity: { type: Number, required: true},
    order_date: { type: Date, default: Date.now }
});

module.exports = mongoose.model('Order', OrderSchema);
```

Fig. 29 Order Schema

User_object

```
//User Schema
var mongoose = require('mongoose');

var UserSchema = new mongoose.Schema({

    firstname: { type: String, required: true, lowercase: true},
    lastname: { type: String, required: true, lowercase: true},
    email: { type: String, required: true},
    visit_count: { type: Number, required: true},
    passwordHash: { type: String, required: true},
    passwordSalt: { type: String, required: true}
});

module.exports = mongoose.model('User', UserSchema);
```

Fig. 30 User Schema

8. UX Design

This section will illustrate the human interface design and describe the look and feel of the application from the user's perspective through the use of wireframes and high fidelity mock-ups.[24]

8.1 Notification screen to Product screen

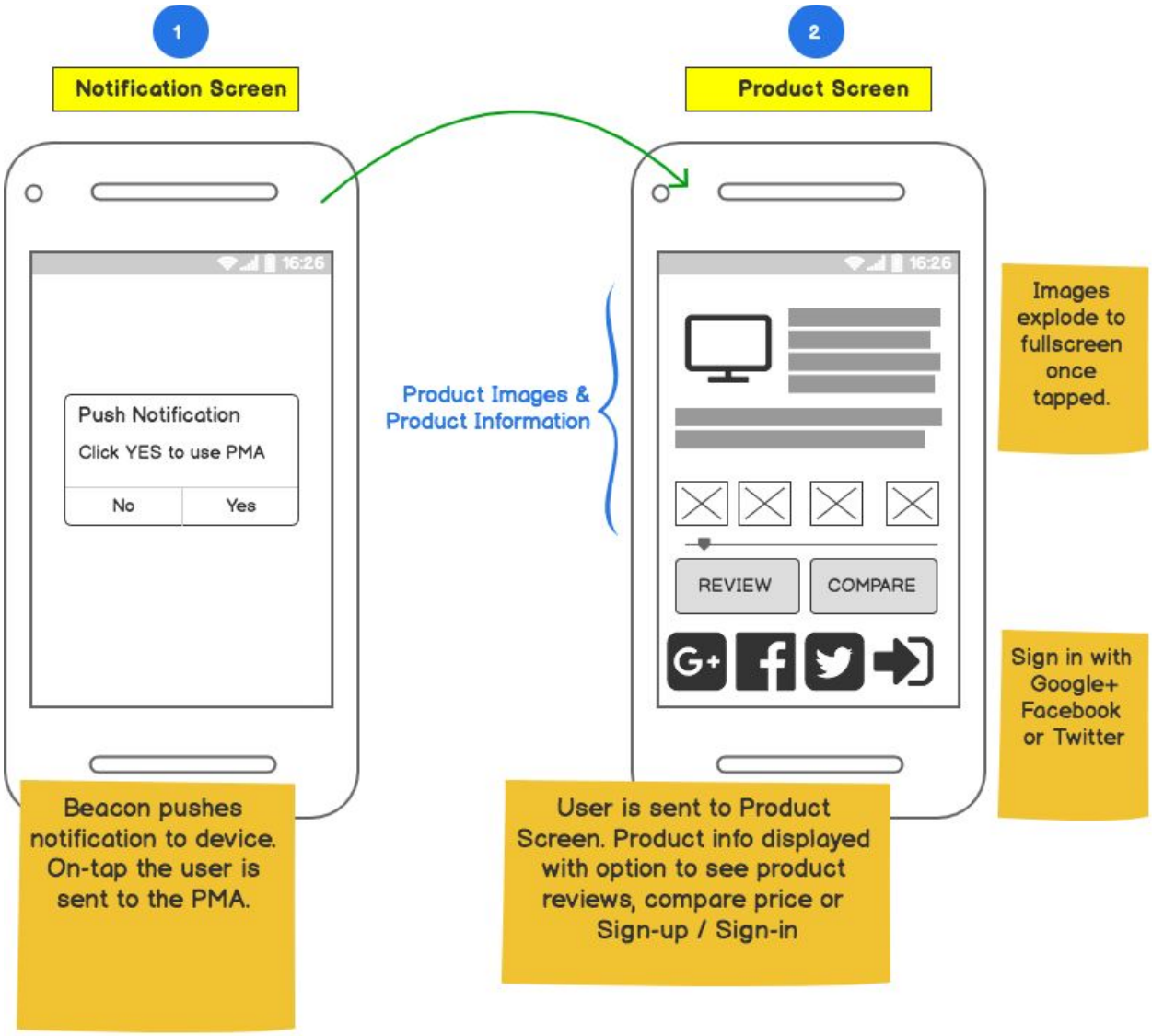


Fig. 31 Notification + Product Screen Wireframes

[24] All wireframes were created with Balsamiq & Mock-ups with Lucidcharts

8.2 Product Review and Price Comparison Screens

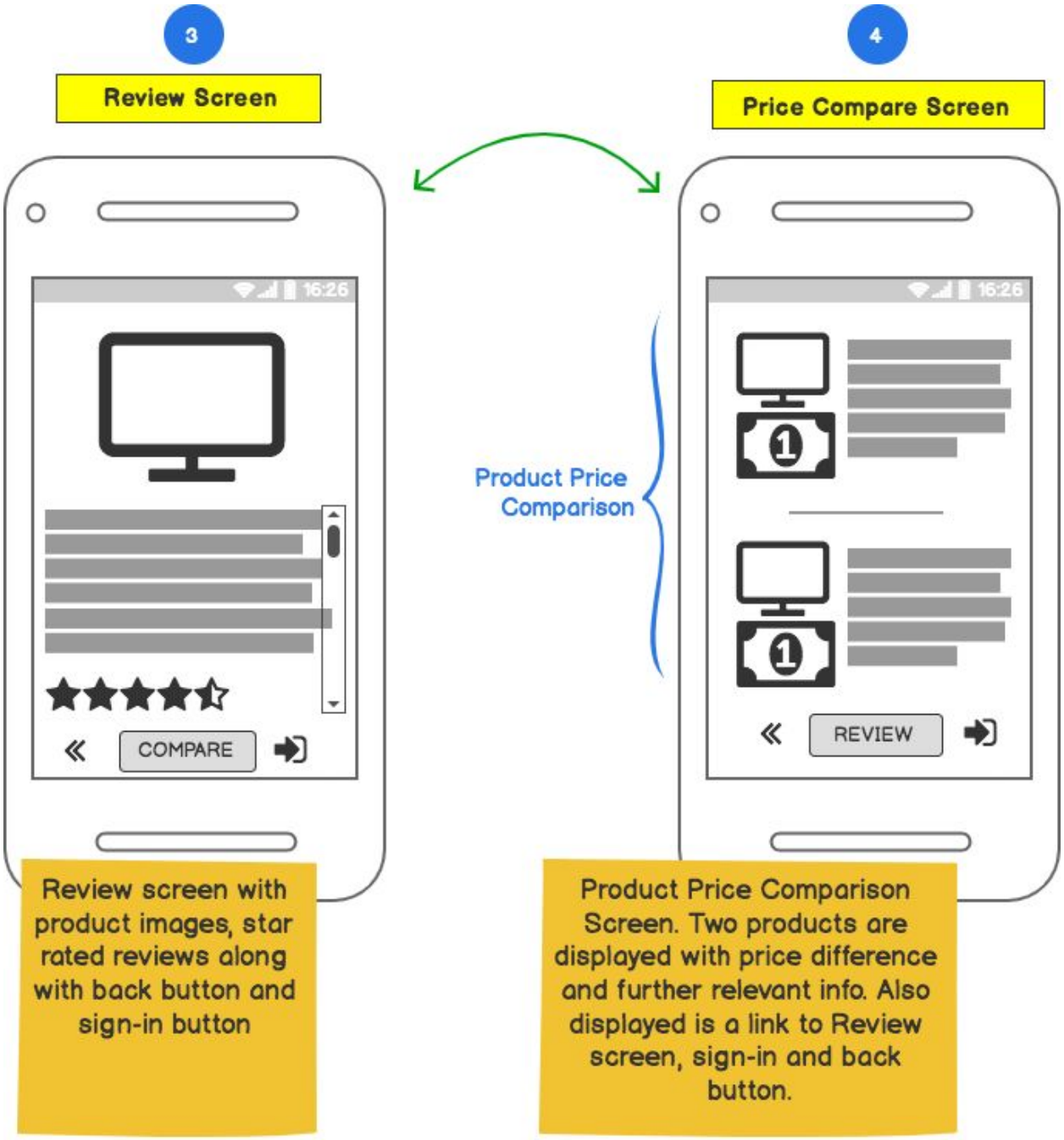


Fig. 32 Product Review + Price Compare Wireframes

8.3 Sign-in Screens

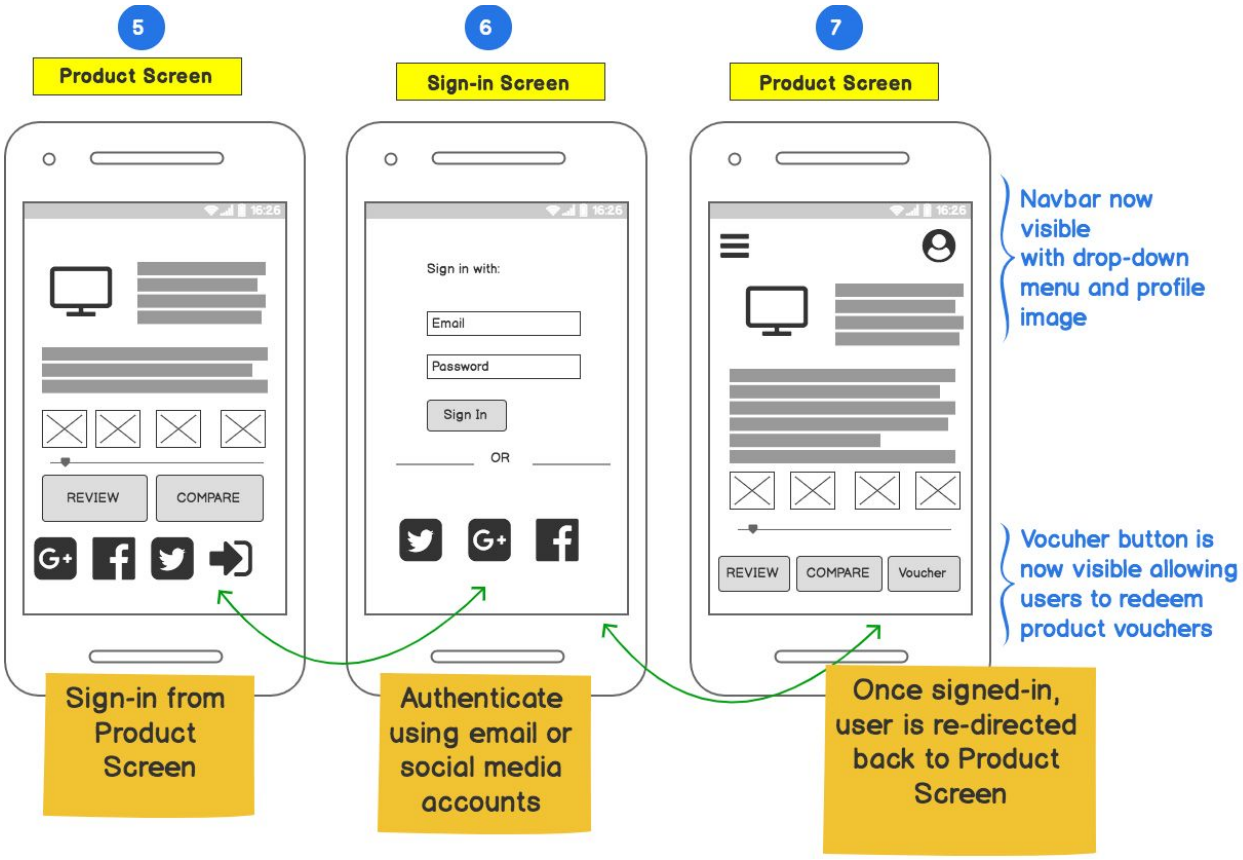


Fig. 33 Sign-in Wireframes

8.4 Navbar - Drop Down Menu and Profile Settings

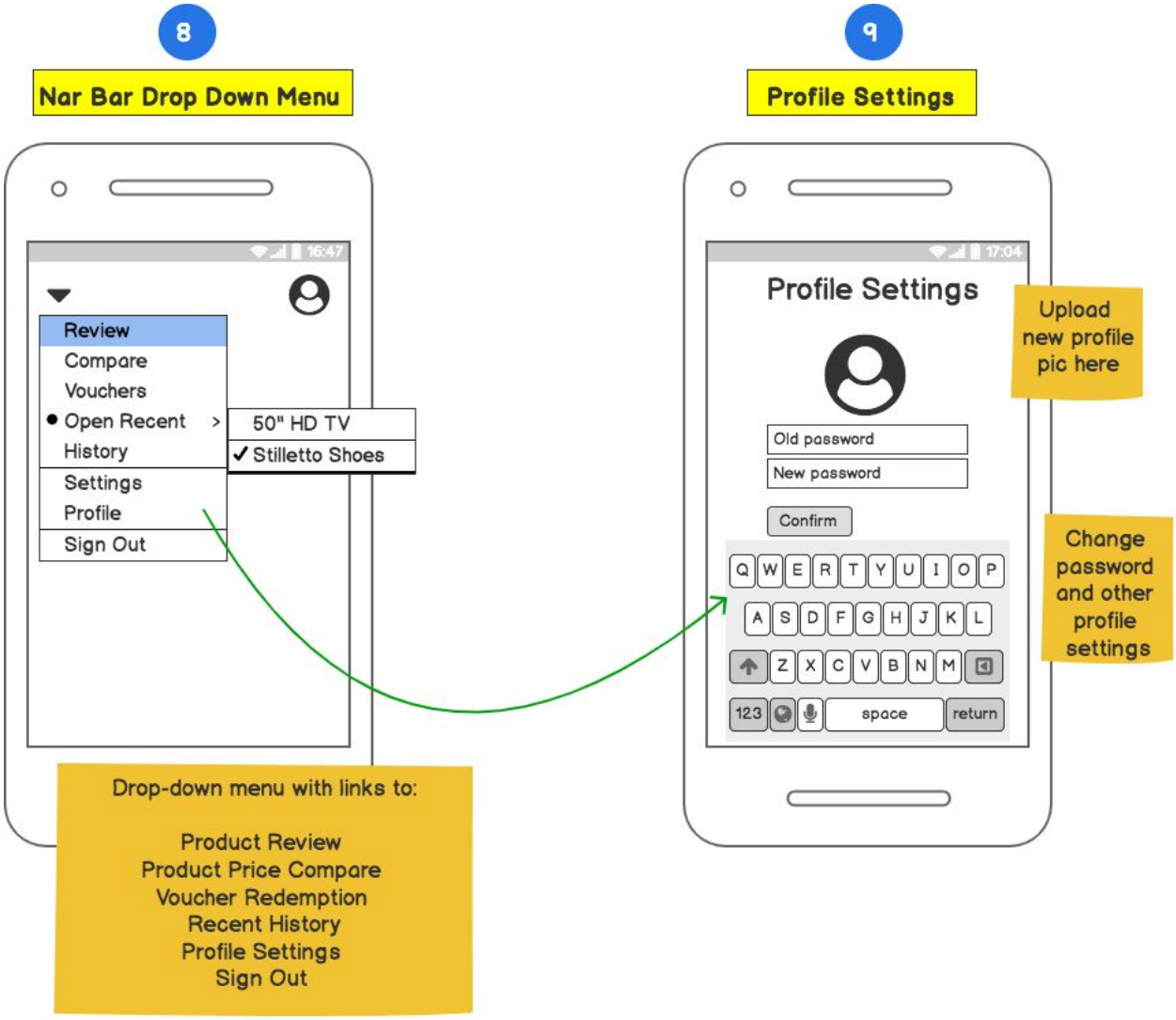


Fig. 34 Nav Menu + Profile Settings Wireframes

8.5 Voucher and QR Code Screens

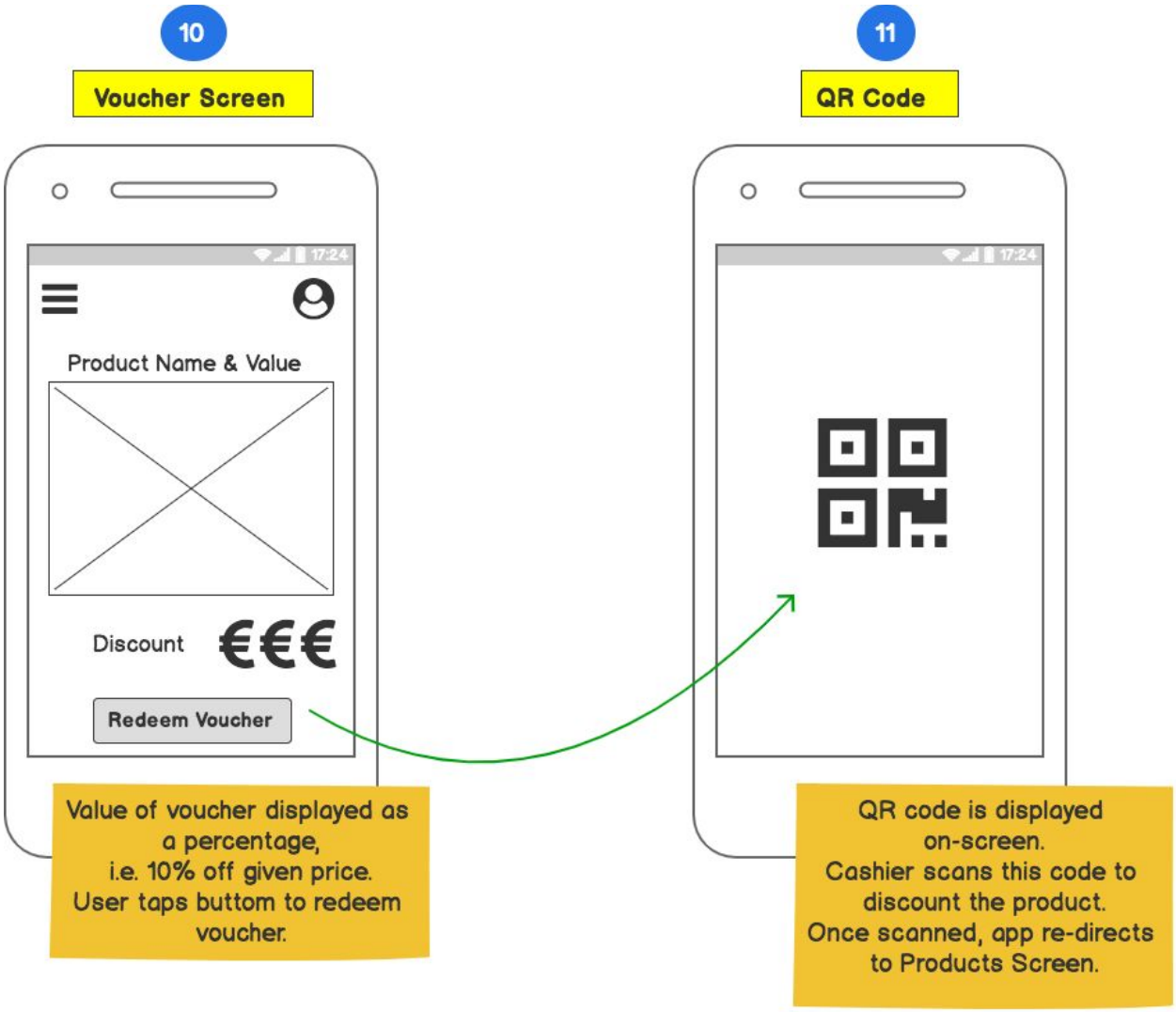


Fig. 35 Voucher + QR Code Wireframes

8.6 Mock-ups

The following are a sample of high fidelity mock-ups illustrating what the PMA will look like. The numbers above the images correspond to the numbered wireframes in the previous section.

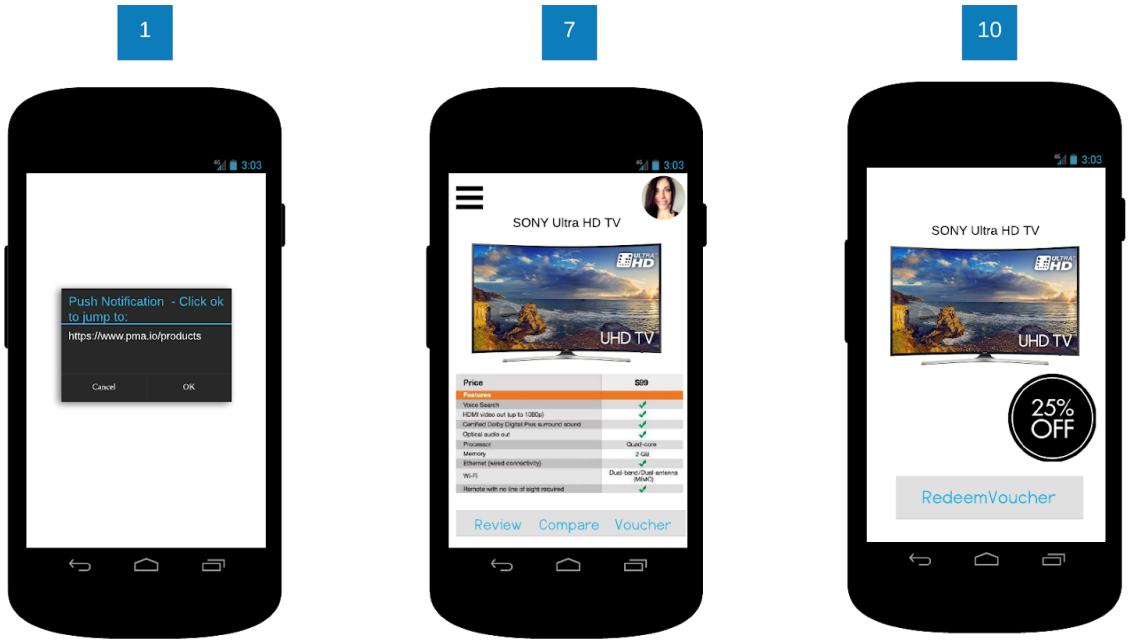


Fig. 36 Voucher + QR Code Wireframes

9. Schedule

9.1 Design and Development Schedule

The schedule outlined below gives a proposed timeline for design and development, highlighting important milestones and deliverables.[25]

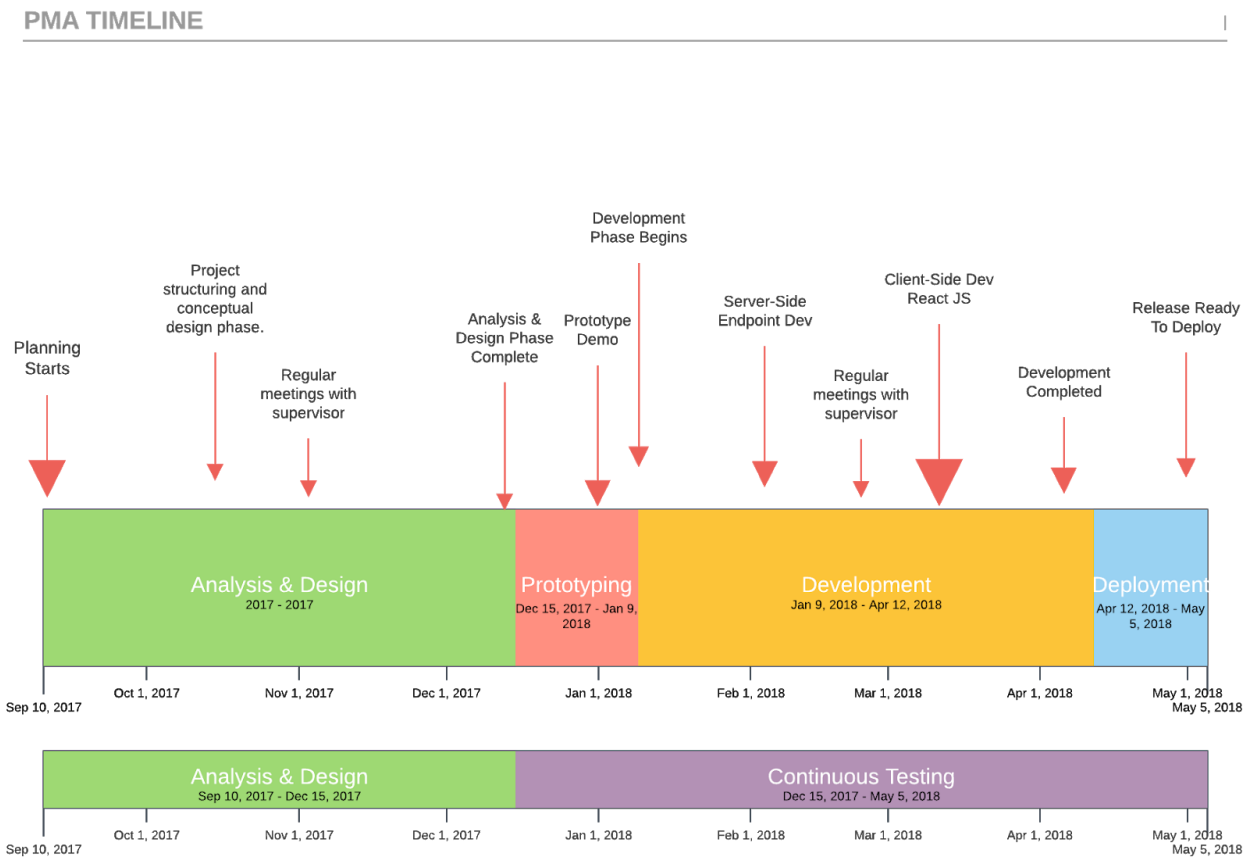


Fig. 37 Design + Development Schedule

[25] Time created with Lucidcharts http://www.lucidchart.com/documents

References

https://www.unacast.com/post/proximity-marketing-what-how-why

https://www.shopify.com/retail/the-ultimate-guide-to-using-beacon-technology-for-retail-stores

Domes, Scott '*Insider Guide to Progressive Web Apps*', Web Designer, issue 266 p. 67

https://www.oreilly.com/ideas/5-web-trends-for-2017

http://uk.businessinsider.com/shopping-app-usage-is-rising-but-retailers-still-have-a-glaring-problem-2016-6?r=US&IR=T

https://www.onyxbeacon.com/7-major-trends-for-ble-beacons-in-2016/

https://medium.com/@CalinLeafshade/why-i-chose-react-over-vue-3dd9a230b507

https://docs.webhose.io/docs/ecommerce-product-data-api

https://cloud.google.com/appengine/docs/standard/python/google-analytics

https://developers.google.com/analytics/devguides/collection/analyticsjs/how-analyticsjs-works

https://kontakt.io/blog/extensive-guide-to-bluetooth-beacons/

http://www.voucherify.io

http://www.webhose.io

https://firebase.google.com/docs/auth/web/start

https://developers.google.com/analytics/devguides/collection/analyticsjs/

http://www.lucidchart.com/documents

http://www.mybalsamiq.com/

Appendices

Appendix A

Methodology Alternatives

The Waterfall model places emphasis on a logical progression of steps taken throughout the software development life cycle (SDLC). Each phase must be completed before the next phase can begin. Development phases never overlap and software testing begins only when development is complete. Disadvantages of this model include:

- Inflexibility - lack of adaptability across all stages of the development life cycle
- Predefined requirements - not suited for projects with a high risk of changing requirements
- No deliverables until late in SDLC
- Late testing period - if a fundamental flaw is found at a late stage, a dramatic leap back in stages is required
- Late user feedback - user input is not iterative in nature

How Agile will be applied

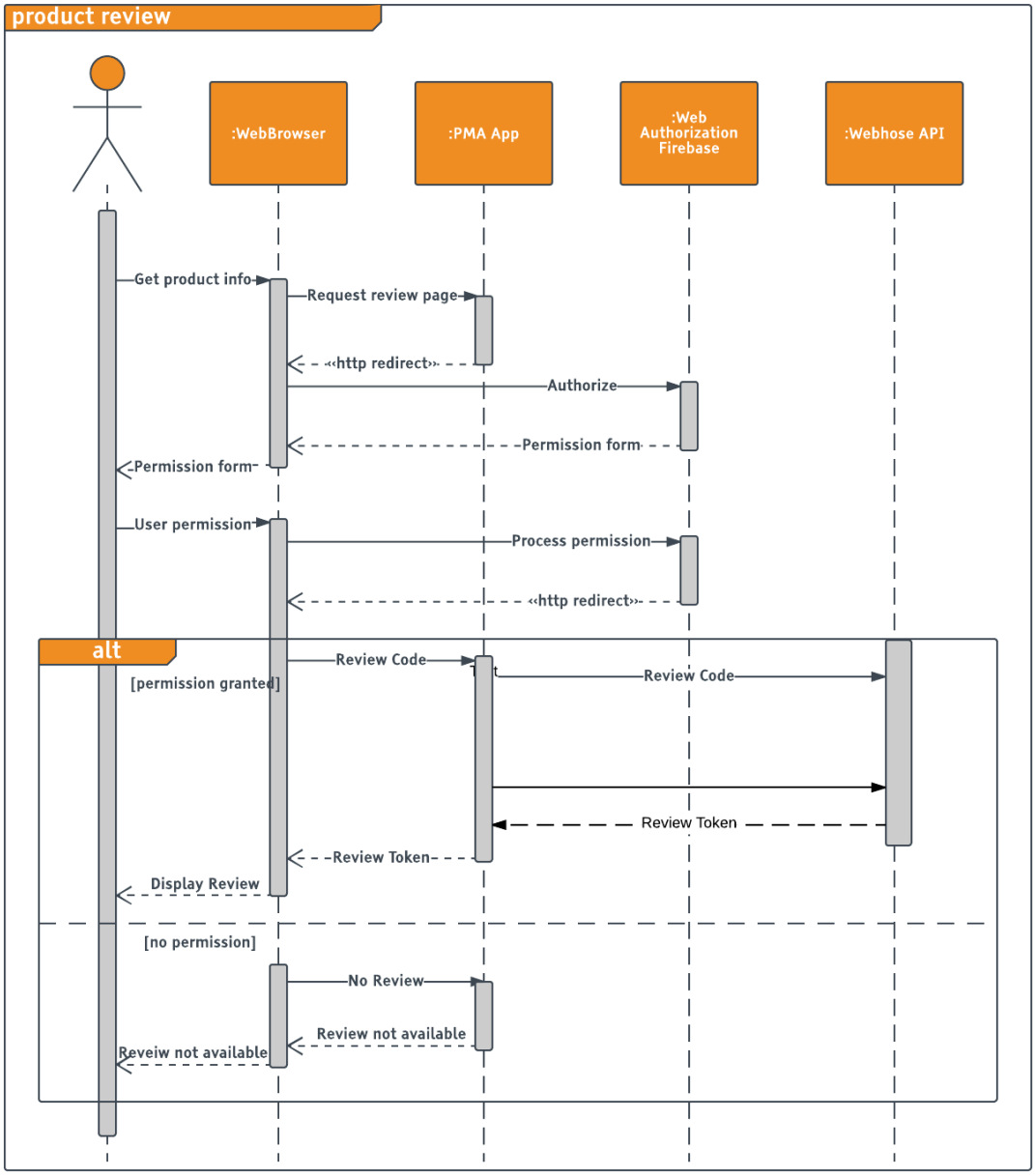
User-stories will be created to capture user needs and goals. User-stories are descriptions of a feature from the perspective of the user or customer. Each user-story will be allocated a 1 - 2 week sprints in which the following steps will be completed:

1. Analysis
2. Develop
3. Test
4. Validate
5. Integrate
6. Deploy

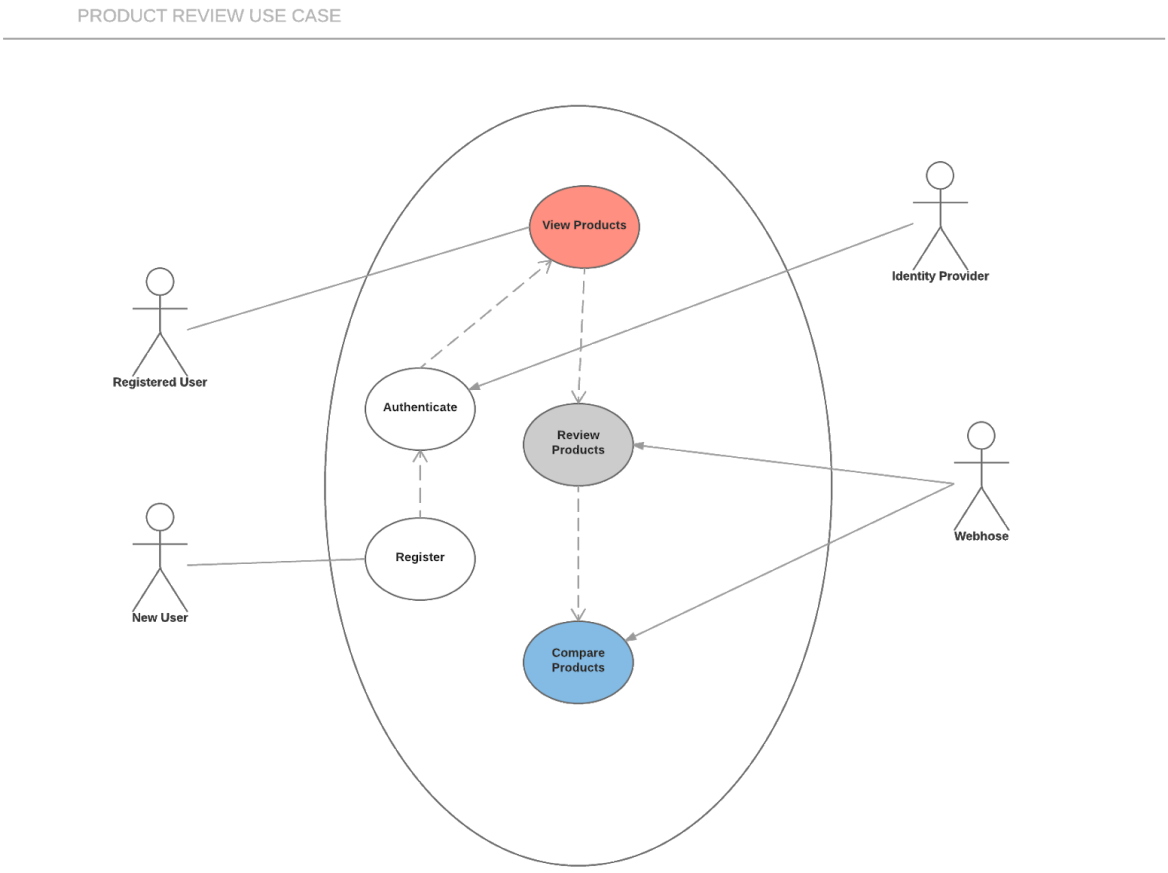
After deployment - the developer will move on to the next user-story. A breakdown of the development schedule can be found later in the Scheduling section.

Appendix B

Product Review Sequence Diagram

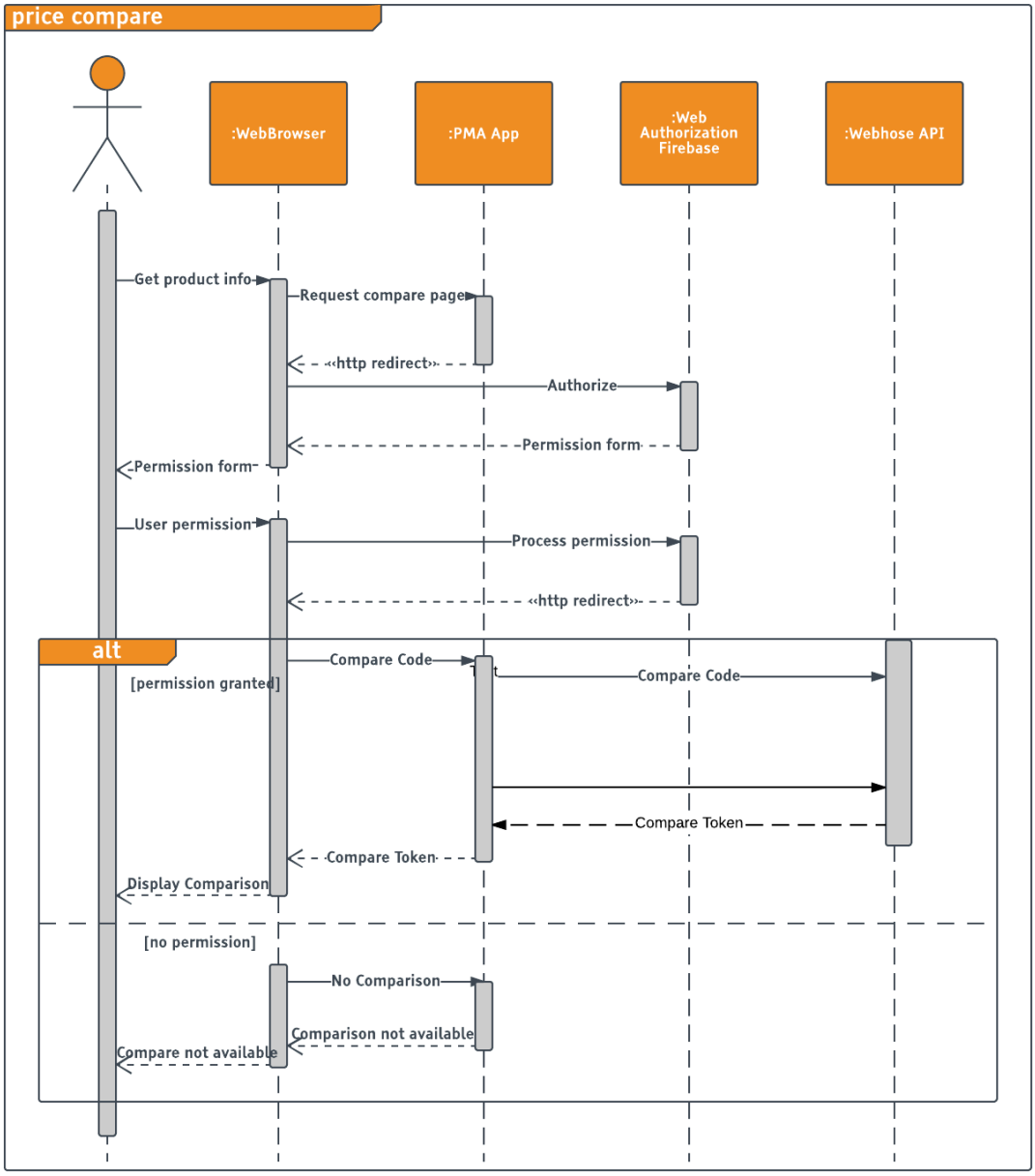


Product Review Use Case

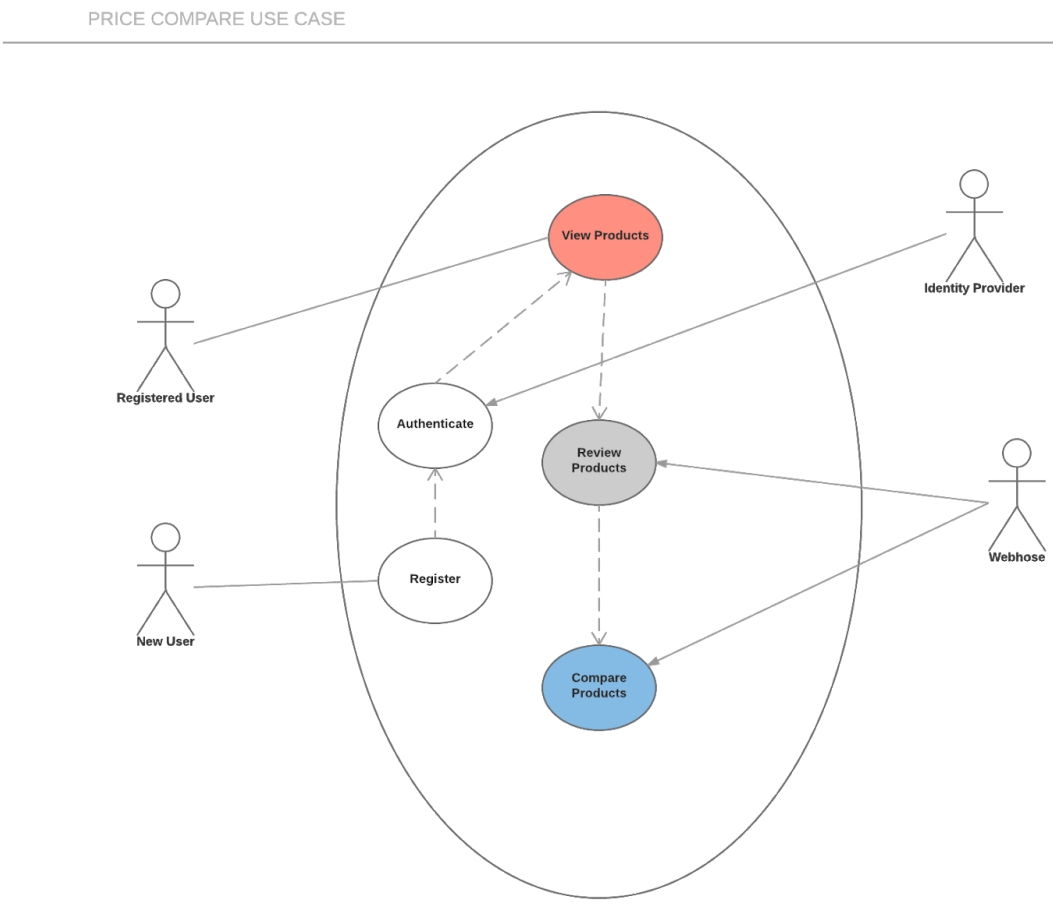


Appendix C

Price Comparison Sequence Diagram



Price Compare Use Case Diagram



Appendix D

For example: searching reviews on amazon.com would look something like the code snippet below - which would return a JSON object: **REF:** www.webhose.io

```
const webhoseio = require('webhoseio');

    const client = webhoseio.config({token:
'485b74f2-1b38-4a80-9fb3-19819afae4e2'});
    const query_params = {
     "q": "site:amazon.com",
     "sort": "crawled"
    }
    client.query('reviewFilter', query_params)
    .then(output => {
        console.log(output['posts'][0]['text']); // Print the
text of the first post
        console.log(output['posts'][0]['published']); // Print
the text of the first post publication date});
```

Appendix E

REF: https://firebase.google.com

Configure API key

```
// Initialize Firebase
var config = {
  apiKey: "<API_KEY>",
  authDomain: "<PROJECT_ID>.firebaseapp.com",
  databaseURL: "https://<DATABASE_NAME>.firebaseio.com",
  storageBucket: "<BUCKET>.appspot.com",
};
firebase.initializeApp(config);
```

Appendix F

Read and write operations (sample code)

```
// Write Product Data

function writeProductData(productId, brand, type, description, price) {

  firebase.database().ref('products/' + productId).set({

    prod_brand: brand,

    prod_type: type,

    prod_desc : description,

    Prod_price: price

  });

}
```

```
// Read one Product

var productId = firebase.auth().currentProduct.uid;

return firebase.database().ref('/products/' +
productId).once('value').then(function(snapshot) {

  var prod_brand = (snapshot.val() && snapshot.val().currentProduct);

  // ...

});
```