Final Major Project Report

IMD Final Year, May 2015

Aaron Colton
B00589391

PSG: 5
Mentor: Zheng, Huiru

Gimme Web App

http://46.101.2.98/

Acknowledgements

I would like to express my deepest appreciation to all who helped and assistanced in the completion of this project.

There are a few individuals who deserve a special thank you. First, to my mentor for the final year project, Dr Huiru Zheng for her contribution in providing guidance, suggestions and feedback which proved valuable to coordinating my project.

Dr George Moore, who provided guidance the whole way through the Major Project module.

Further more I would also like to extend a very special thank you to DR Peter Nicholl & Professor Jonathan Wallace. Both Peter & Jonathan have contributed extensive mentoring and support over the past few years, which continued throughout final year.

Finally, a special thanks goes to my business partners at Amigo Studios, Ali Coleman and Andrew McKenna, who contributed valuable help and support throughout the project.

Contents

1. Introduction

1.1 Overview

This report will cover the process and the work undertaken to design and develop a web application for both users and business administrators. The report will make use of visuals were possible. All other visuals than are important to the understating of the project will be included within the appendices.

The major project being reported can be viewed at http://46.101.2.98/
Login details: - **Username:** Demo **Password:** Demo123

1.2 Background

Internet advertising is a huge market. According to research from the advertising association in 2013, more money was spent on internet advertising than any other form of advertising. The total expenditure was £6,300 million for the UK alone. Businesses are continually looking for new ways to be seen in an overcrowded market. In 2013 mobile advertising saw a year-on year growth of 95.2%. The data shows that there is an increasing demand for mobile advertising. (Advertising Industry Statistics, 2015)

Currently, several applications exist that allow shoppers to search for promotional offers by location or interest:

Groupon (http://www.groupon.co.uk/)
Living Social (https://www.livingsocial.com/gb)
Tippr (http://www.tippr.com/)
HomeRun (https://homerun.com)

The examples are some of the more popular applications. The common denominator in each is the ability to filter content by location. The key factors missing from all of the examples above are (1). The ability to tailor the experience to your personal preferences (2). Neither of these apps attempts to bridge the gap between internet shopping and high street shopping.

A problem has risen from the current model were desired content is now becoming spam due to the lack of control. It also does nothing to encourage shoppers to return to the high street where there is a potential to increase sales further. (Digitalstrategyconsulting.com, 2015)

1.3 Aims and Objectives

The aim of the project was to create a web application that would enable users to search for offers within their current location, based on chosen categories of interest. Offers are then claimable by the user and redeemed in store. Users would be able to connect with friends and notify one another about offers. A backend system would allow businesses to manage store accounts, create offers and view statics for claimed offers.

Objectives for the project.

- Create an easily memorable brand
- Navigational structure and user journeys
- Wireframes and conceptual designs (paper based)
- Functional layout and structure (html/css)
- Database designs
- Creating the backend administrative functions in Laravel (MVC Model)
- Testing functions
- Design Implementation
- Deploy to live server

1.5 Target Market

Shopping is a task undertaken by almost everyone. It would however be unrealistic to target every shopper so the focus would be on the “mainstream shoppers”, the ones that carryout the day to day shop for the entire household. Majority female, aged 20-40 and are likely to be mums. Mums make purchases on behalf of their children and their partners. Research shows that mums account for 70% of all visits to the supermarket. (Magazine, 2015)

2. Concept Definition

2.1 The Idea

The project would see a web application built that would allow businesses to create store offers and target them towards a specific region or store. Users of the application would be able to able to view and claim offers that are available within their location.

Details

Business owners can register for an account via the website. Once registered the administrator can then add store locations to their account within the backend system. Offers can then be created and applied to a store location. The administrator would then choose which category or categorises the offer is related to. Depending on the categories chosen a series of images related to each are loaded into the view. The administrator would then choose an image to represent the offer.

Users of the application would register for an account and select to receive offers based on their own interests. Offers that are available within a users location are loaded into the offer feed. Users may also search for offers based on a specific location. The offers within the user feed be will updated as the user travels from location to location. When new offers are available a notification will be sent. The option to 'turn notifications off' would be available to users within their profile settings. Users would have the ability to share offers with friends that have also signed up to use the application. A user would be able to view friends recently claimed offers within the application.

2.2 Management and Planning

Planning was an important part to managing the project. The objectives for the project became tasks which then saw sub tasks being created. 'Teamwork' project management tool was then used to create a gantt chart, mapping out all tasks and sub tasks to be completed. (Teamwork.com, 2015) Each task/subtask was set an estimated date and time to completion. Once completed, tasks could then be marked off and the gantt chart provided an up to date representation of the progress. This approach to management was key to delivering the project on time.

**See gantt charts in Appendix A*

2.3 Methodology

There are several well know methodologies that could be chosen to help manage the project.

Waterfall

Beginning at the start and working towards the end in a liner fashion, requirements are set out at the beginning and doesn't allow room for changes during the design and build of the project. (Segue Technologies, 2013)

Modified Waterfall

Modified Waterfall follows the same stages as the conventional waterfall approach. The key difference with modified waterfall is that it allows for overlap. This allows for changes to be made during the development.

Agile

The agile methodology is to build small parts of the project quickly for testing. This gets projects off the ground fast and works well for team based development. Has no real structure. (Segue Technologies, 2013)

Prototyping

The idea is to create prototypes for testing then to go back and make refinements and build into a working version. Flexible and good for applications with a high volume of users, though, it can be complex and time consuming.

These are just a few of the possible methodologies. For this project a **Modified Waterfall** approach was chosen as it was best suited, seeing each step in the project completed in a linear fashion but allowing for small changes to be made during the development.

2.4 Requirements Specification

To help consider the requirements of the application the Volere Requirements Specification Template was used. Snow cards are used to state a requirement for the application. (Volere.co.uk, 2015)

Figure 1

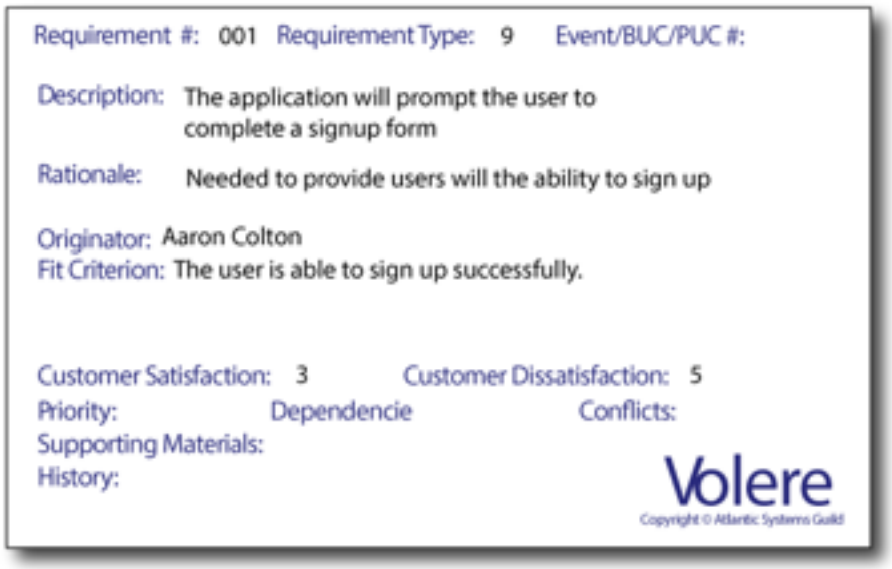
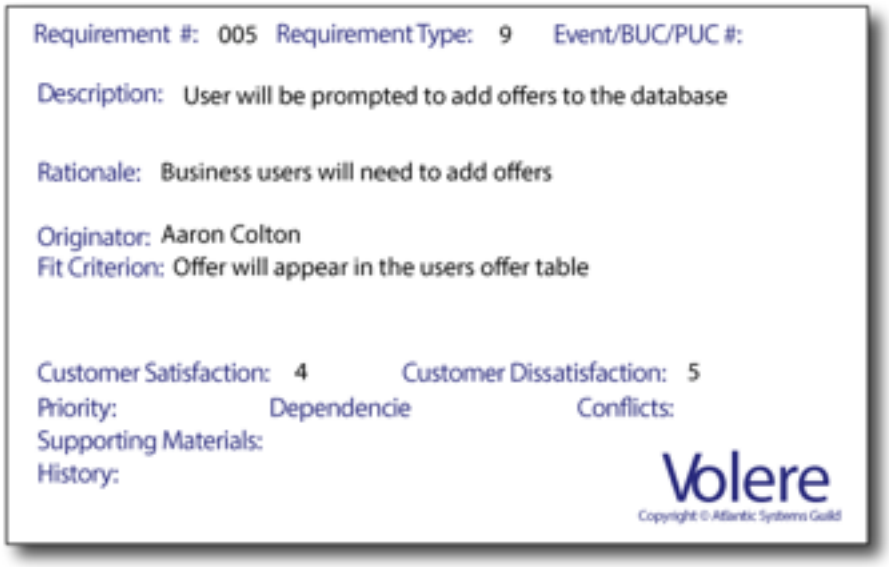


Figure 2



Both examples are functional requirements, type 9 within the Volere specification template.(Volere.co.uk, 2015)

Tables for the requirements specification can be seen in Appendix B.

3. Design

The design of the application was considered, from the visual look and feel of the application all the way through to the systems structure and architecture.

3.1 Paper Prototyping

Paper prototyping is the process of creating rough sketches to encourage idea generation. Ideas can be easily thrown away or expanded upon. This process helps to avoid timely mistakes in the design process when creating the digital works. The process was followed to consider the look and feel as well as the interactions within the application.

The main page for the front-end of the application is a feed consisting of store offers. Good use of imagery is an ideal way to present the offers as there isn't a lot of space for details on mobile. Below are some of the ideas that were generated through the process of paper prototyping.

Figure 3

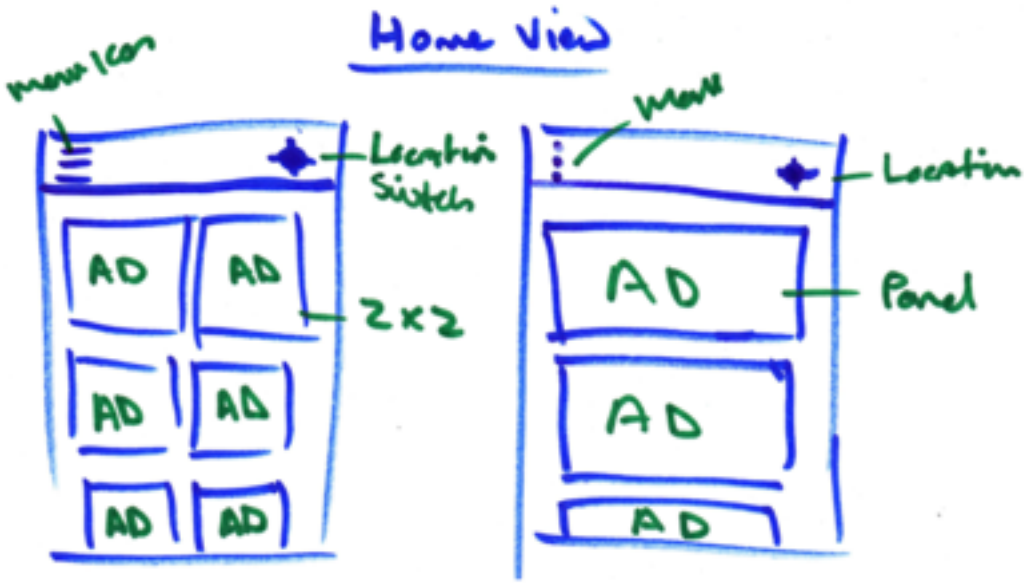
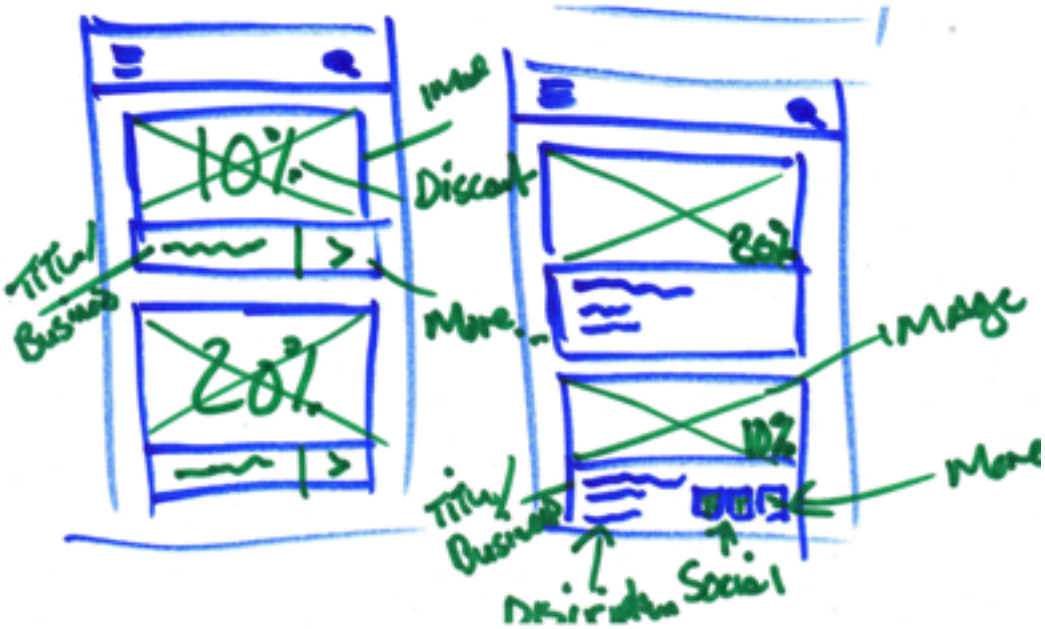


Figure 4



Detailed Sketchlng

Following on from from the rough sketches, a higher fidelity prototype was created to consider how the view might look when some colour and imagery were applied to the design.

Figure 5

The main noticeable aspect of this layout was that it created less clutter than the other layouts considered during the paper prototype process.

**Further evidence of the paper prototyping typing process can be seen in Appendix C.*

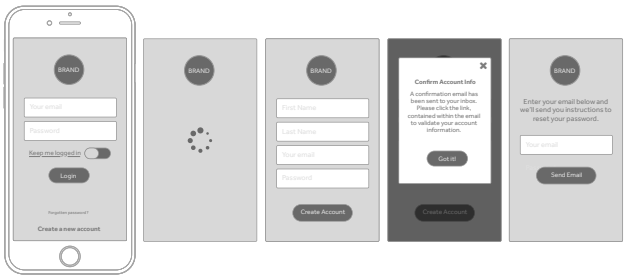
3.2 UX Design

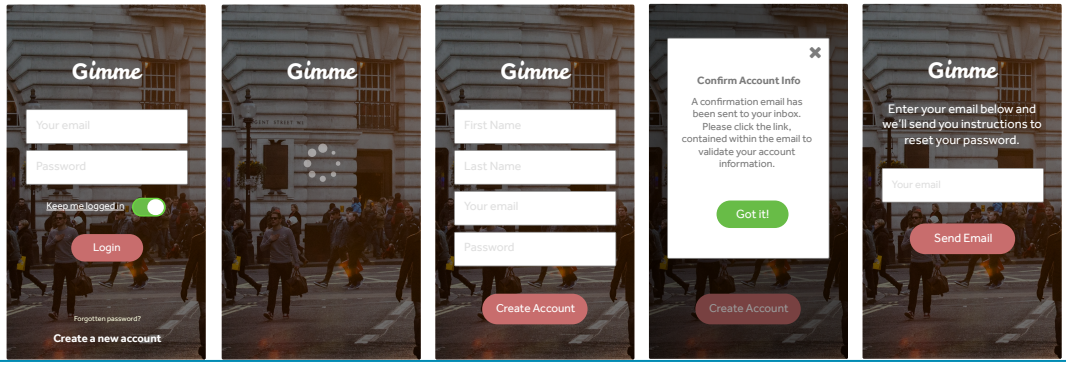
Leading on from the paper prototype design, was the experimentation with the user experience design.

The user experience design was an important step in the planing of the application. Through research, (Blog.karachicorner.com, 2015) wire framing and experimentation with colour, typography, iconography, imagery and brand personality, the user experience was tested and interactions within the application were considered.

User Login And Registration

Figure 6

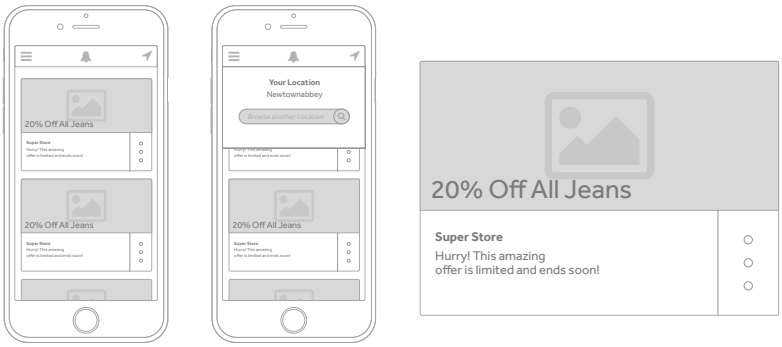


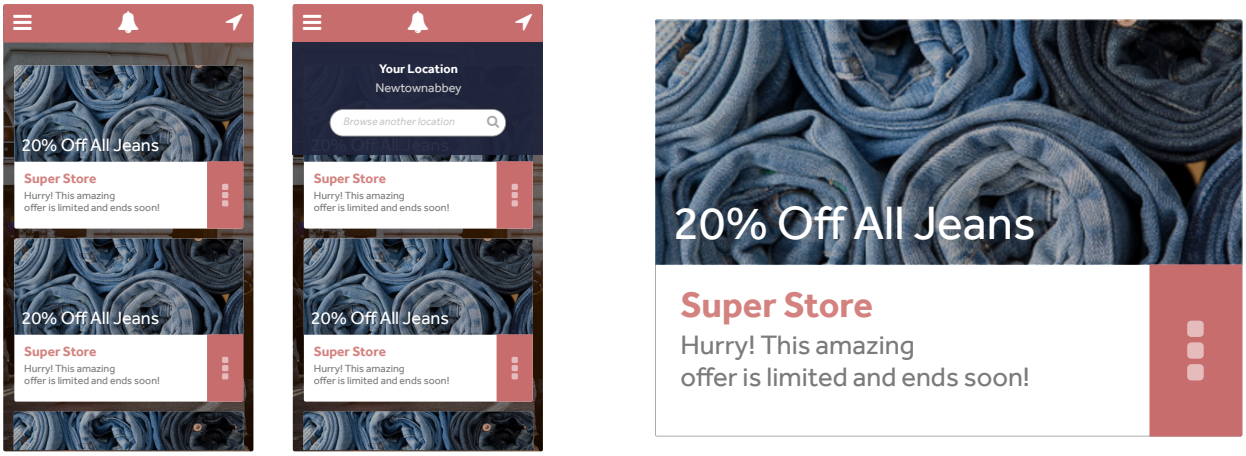


The brand ‘Gimme’ is accompanied by an attractive image of high street shopping that helps to communicate the purpose of the application. The toggle button was chosen in place of a checkbox to give a better user experience on mobile devices. Both the login and create account forms are simple and uncluttered. The primary button style and colour is soft and pleasing to the eye but prominent enough to be a clear call to action. Plain text is used for secondary actions on the login screen such as creating an account or the forgotten password option. Colour and font weights are used to communicate the hierarchy of these actions.

Offers

Figure 7





Offers consist of an image to visually communicate the category, a descriptive title, store name, short description and call to action button. Using a grid based design, all of this information is presented in a minimalistic panel. The call to action button uses the primary colour.

Figure 8

Offer Vouchers

Claiming an offer will load the voucher view. The voucher can then be redeemed in store. Using a good choice of colours, typography and icons the voucher is illustrated through minimal and flat design.

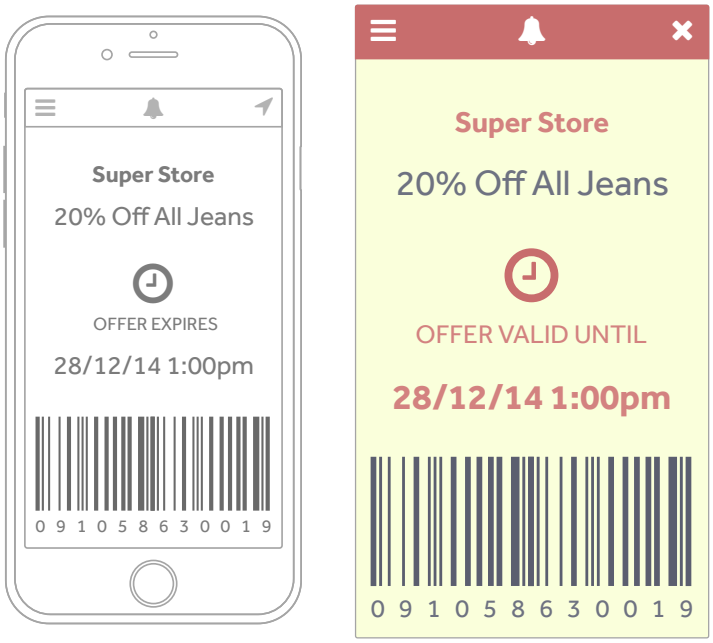
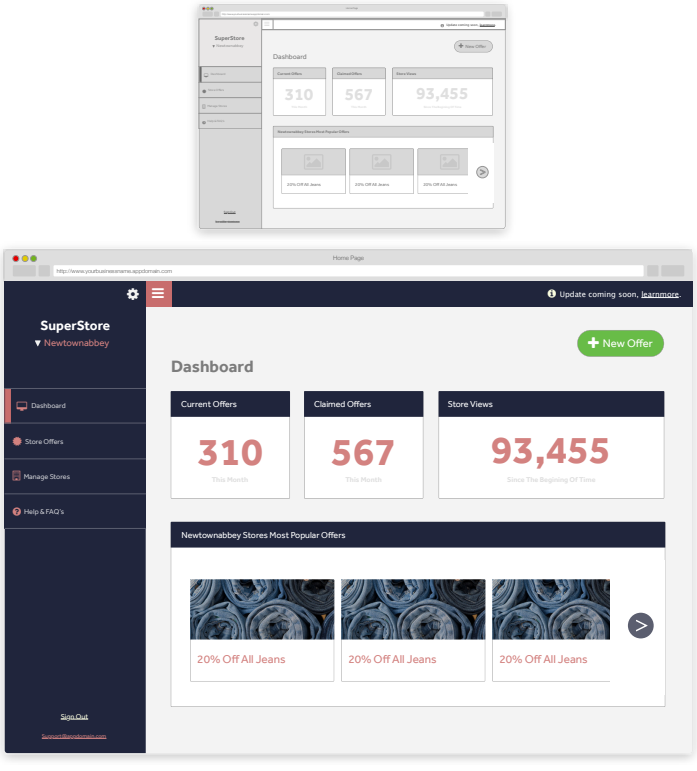


Figure 9



Admin Dashboard

The admin dashboard was designed to make the user experience simple and intuitive. The design takes into account all of the main assets of the system that the admin uses most frequently. These parts of the system need to be quick and easy access. A

large green call to action is displayed for the primary action, along with statistics for the account.

Figure 10

Stores

Store locations are created within the admin system. To improve the user experience when adding a store, it was decided that a map would be included on the page. This would allow for the user to verify the correct location has been set before submitting the form.

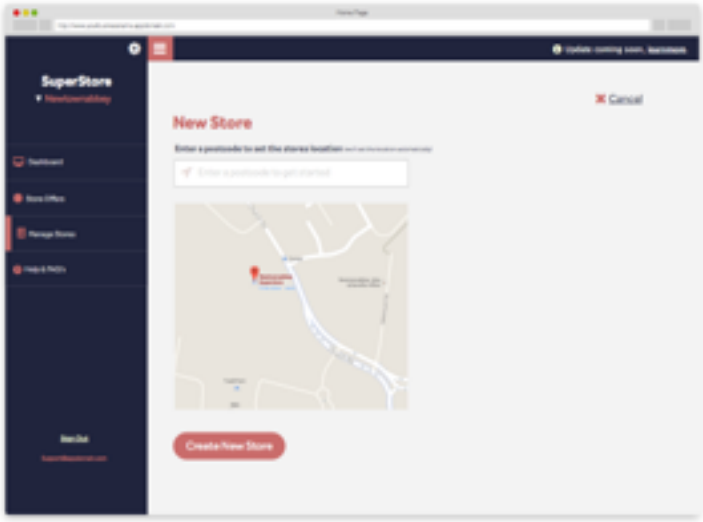


Figure 11

Offer Create

Clicking the new offer button will display a form.

The offer button is swapped out for a cancel call to action. The form is again minimal and echoes the theme styles.

Again, in place of a list, a scrollable slider is used for the category images section.

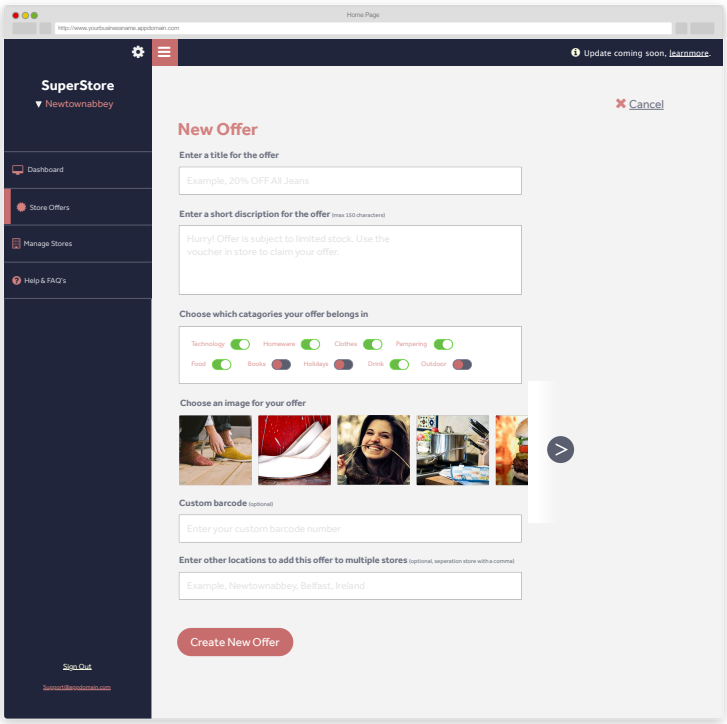
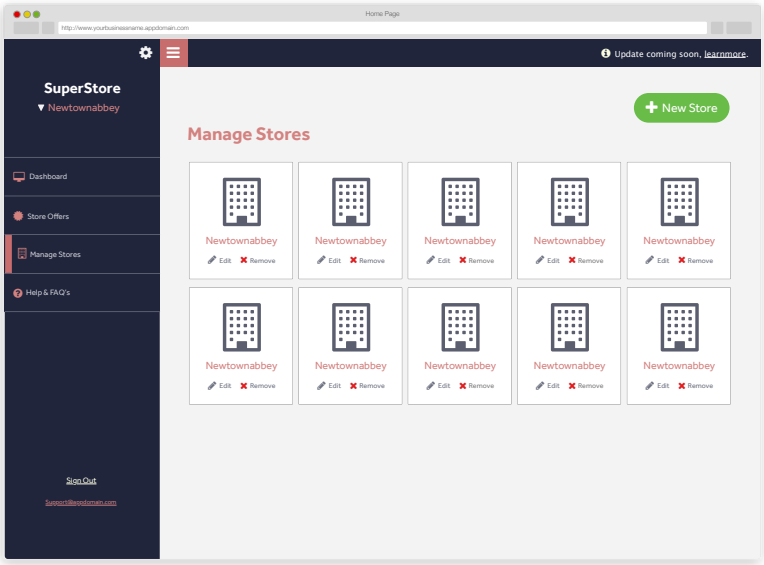


Figure 12

Manage Stores

Each Store is represented using a store icon with the options of edit and remove. The store region is displayed to quickly identify the store location.



**Further evidence of the UX Design can be seen in Appendix D.*

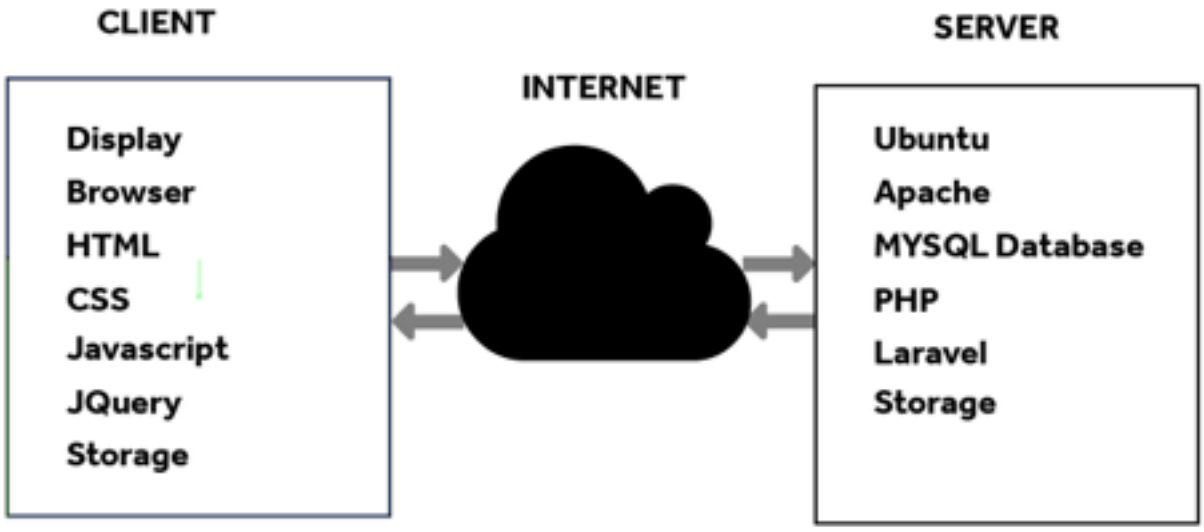
3.3 System Design

The structure of a system can be represented in many ways. Mapping out the system structure helped to see the various components and relationships between them in a visual diagram.

Client-Server Model

The client-server model outlined the network architecture of the system. The digram shows the various elements on the client side and on the server side of the system.

Figure 13



The digram shows how data flow between the client and server-side.The server for the application is setup using a LAMP stack (Linux, Apache, MySQL and PHP).

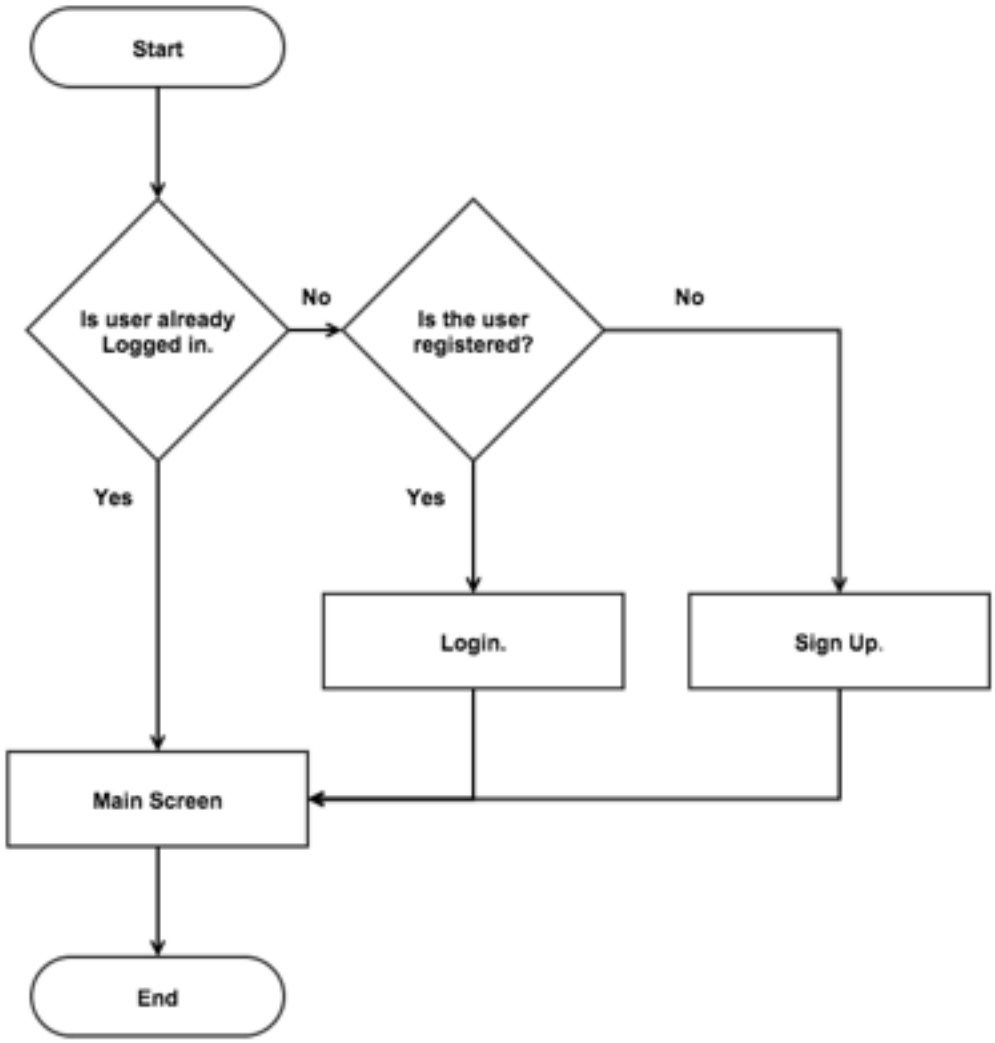
Work Flows

Diagrams were created to understand the workflows for parts of the application that needed to be developed.

User Login/Register Work Flow Diagram

The diagram below shows the workflow within the application for the login and signup process. This helped to understand the workflow before attempting to develop this part of application.

Figure 14

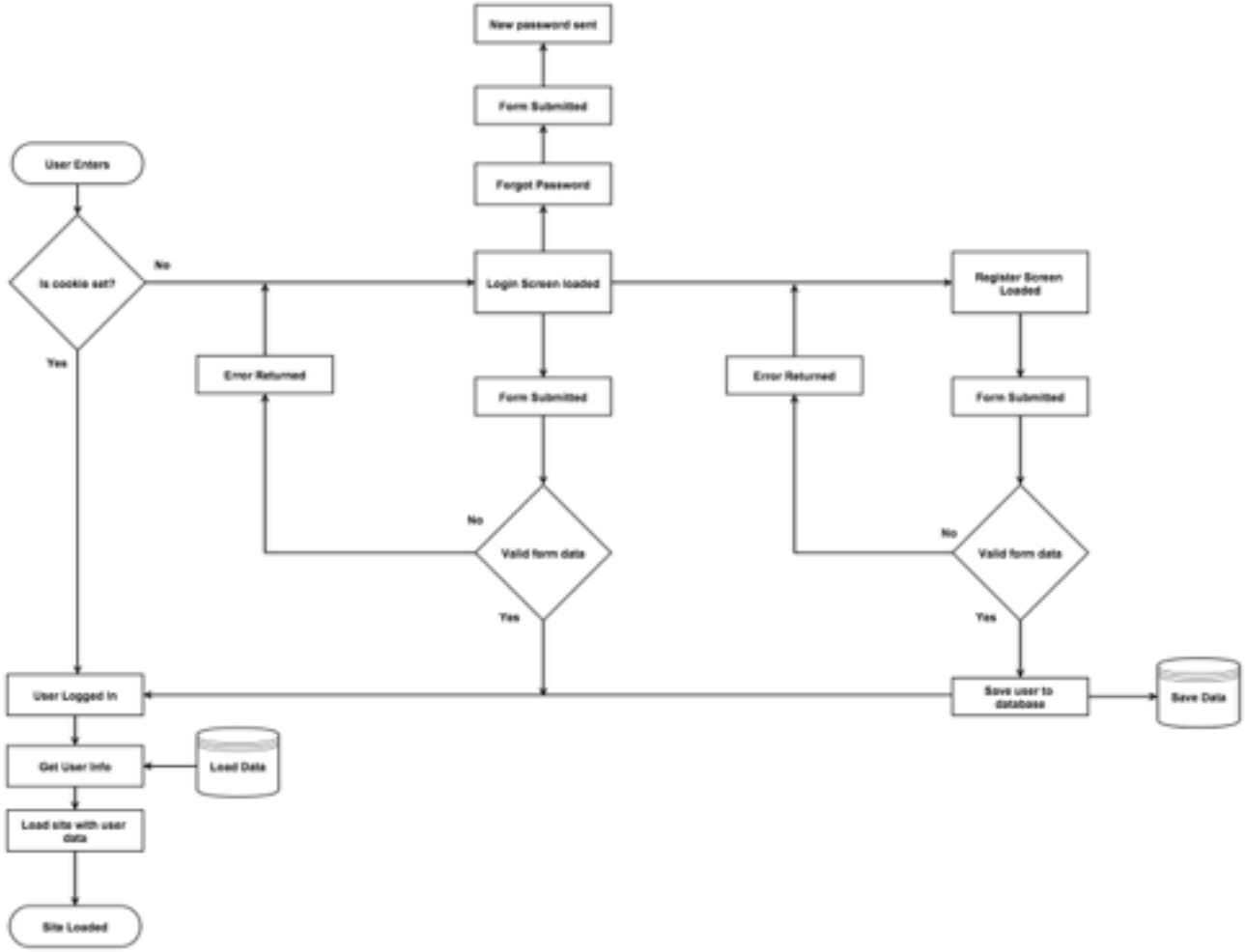


Data Flows

Data flows are an important part of the system design. Diagrams were created to map out the flow of data for workflows within the application. The diagram below represents the workflow above - User login/Register .

User Login/Register Data Flow Diagram

Figure 15



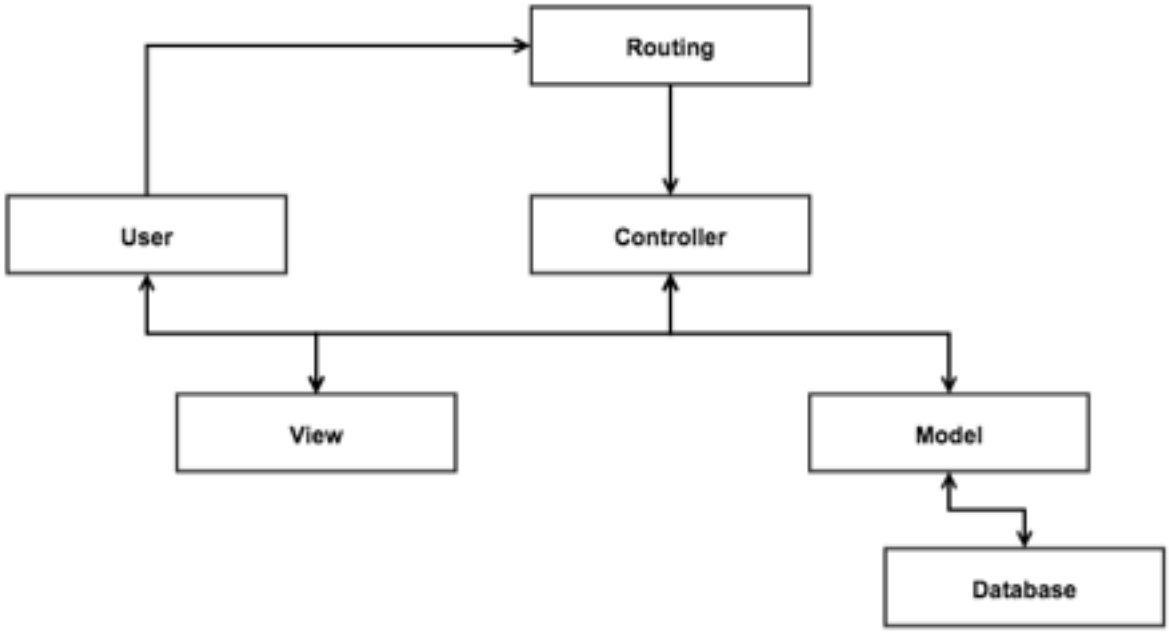
As you can see from the diagram, the application first checks to see if a cookie set. If true, the application will continue to log the user in, at which point a request will be made for the user data. The user data is then returned and the site will be fully loaded at this point. If the cookie is not set, the login screen is displayed. If the form data is submitted, The

data is validated and if valid, the application continues to logging the user in and loading the user data.

Model-View-Controller

The application is built using the Laravel PHP framework. Laravel uses an MVC model. The MVC model splits an application into three main parts - the model, the view, and the controller. Laravel also uses a powerful routing engine which matches the URL to the controller.

Figure 16

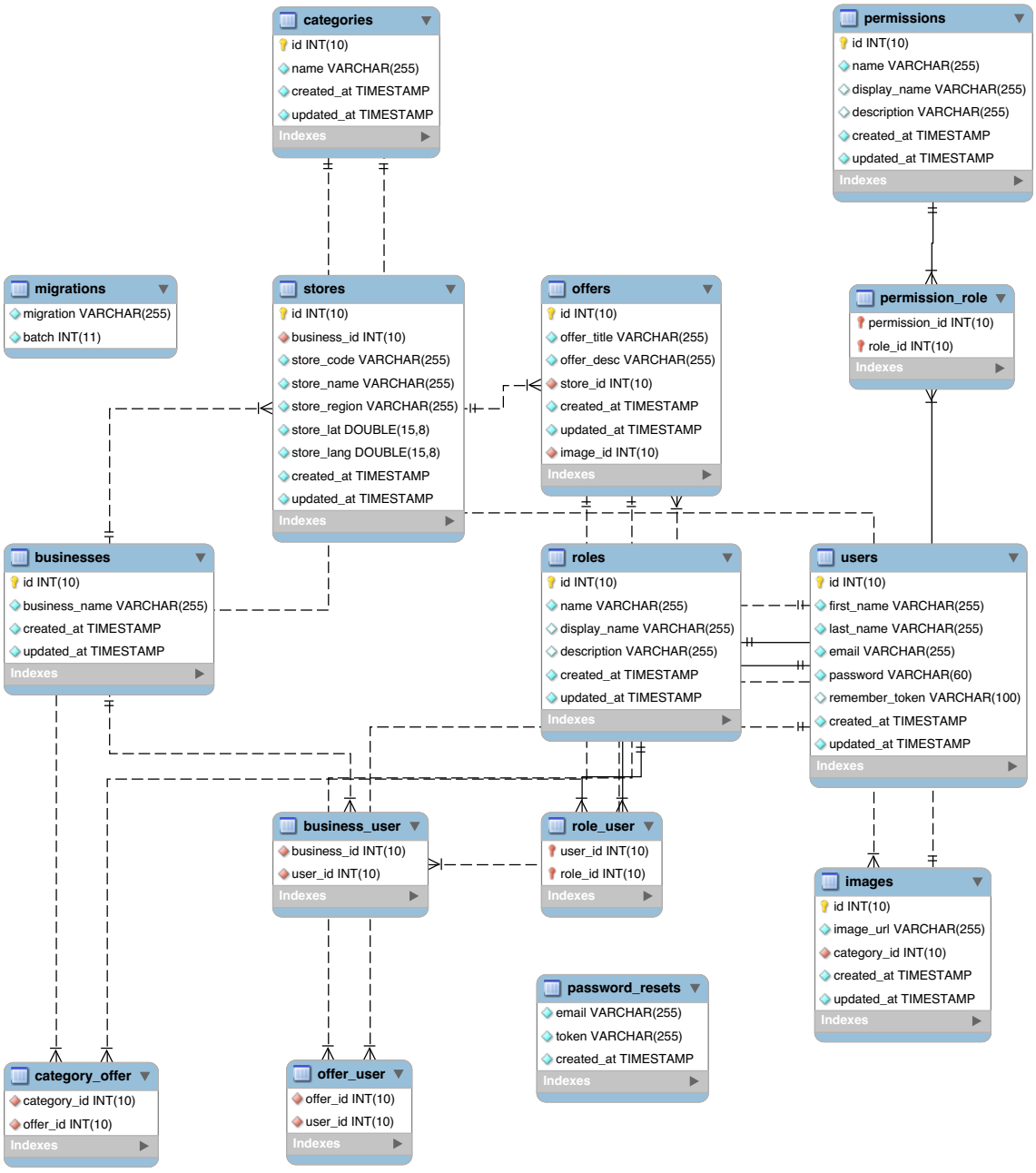


3.4 Database Design

The database design is an important part of the application. To create a visual representation of the database design, the database schema was created using MySQL Workbench. This presented an opportunity to consider any linked tables that could be used to assist the flow of data within the application. The relationships between tables

within the applications database i.e manyTomany, oneTomany etc. and foreign and primary keys were thought out when creating the digram. The diagram below shows the database design for the application.

Figure 17



*Further evidence of the paper prototyping typing process can be seen in Appendix E.

4. Implementation

4.1 Technology

To develop the project a combination of several technologies were used. Each technology used was chosen because of its suitability to the project.

4.2 Server-side

Php

PHP ("PHP: Hypertext Preprocessor") is one of the most used languages for building dynamic web applications. PHP is open source which means it's completely free to use on any type of project, personal or commercial. It is also worth noting that PHP can interact with many different database languages. (Phpbbhq.com, 2015) PHP was also the basis of a module taken while studying IMD.

PHP is compatible with most servers and the setup is easier than with other technologies such as Ruby on Rails. (Ruby-lang.org, 2015) Having previous experience with the language, it was chosen for the project because of it's capability to produce advanced web applications and also because of the community offering support and documentation which proved helpful when troubleshooting the build.

4.3 Google Maps API

Geocoding Api

Geocoding is a process that converts addresses into geographic coordinates (latitude and longitude).

Reverse geocoding is the opposite of Geocoding, it will take latitude and longitude values and reverse them back to a readable address.

When using the Geocoding Api, a new map element can be generated using the results.

The api was used within the application for two main aspects. The first of which was to add a store location for a business. The users current location is detected and a new map element is generated along with a marker to verify the location. The location can be updated via a form field or by dragging the marker to get a more accurate result. (Google Developers, 2015)

Google Distance Matrix Api

The Distance Matrix API is a web service that can be used to retrieve travel distances and total time of travel between origins and destinations. The Api is free to use but it does come with limitations for the maximum number of queries.

- 100 elements per query.
- 100 elements per 10 seconds.
- 2 500 elements per 24 hour period.

A paid service is also available that will increase the queries to:

- 625 elements per query.
- 1 000 elements per 10 seconds.
- 100 000 elements per 24 hour period.

The free service was chosen to be used for application as the query limitations are more than sufficient for the application in it's current stage of development. The system uses the distance matrix api to calculate the distance between store locations and a users location.

If the distance is less than 300 meters for stores that have offers, the available offers are then loaded into the users feed. (Google Developers, 2015)

4.3 Client-side

HTML5, CSS and JQuery were used to develop the client side of the system. Each of these technologies are considered the basic building blocks of a website. The technologies have been the basis of IMD in the past and so experience has been had using all of the languages.

Html

HTML (Hypertext Markup Language) refers to links used to connect pages to one another. As well as displaying Markup text, HTML can display images and other types of media. HTML is used to create a structure for a web page and the content of a web page. Content within the web page is contained within HTML elements such as <p> for paragraphs <img> for images, <title> page title, defines a block, etc. The combination of these HTML elements are what form the building blocks of a web page. The current version of HTML is HTML5. (W3schools.com, 2015)

Css

CSS was used to create the styles for the front-end of the application.
CSS stands for Cascading Stylesheets and is used to style webpages. With CSS you can style elements within your HTML document using CSS selectors. All modern browsers apply their own default CSS to HTML elements, CSS Stylesheets can be used to override the default styles. (Mozilla Developer Network, 2015)

Javascript

JavaScript or as it's more commonly referred to, JS is an object-oriented scripting language for building web sites. Javascript enables developers to make web pages more interactive and allow for certain tasks to be performed on the client side rather than the server side. Handling event actions such as onClick, onChange etc. is much faster on the client side. There are several popular libraries available in javascript, the most popular being JQuery. (Mozilla Developer Network, 2015)

4.4 Database

To enable the application to store and handle data, a database technology is required.

MySQL was chosen as the database technology to be used for the project because it is easy to use and the syntax is simple to understand. It is capable of handling huge amounts of data so it is suitable for use with large applications. MySQL is an open source database (RDMS – Relational Database Management System) it's free to download and use. It is a popular choice for web applications most widely used in combination with PHP. Some of the largest web applications on the web today use MYSQL such as Google, Facebook, You Tube, etc. The technology handles data in a fast and reliable manner even when being accessed by multiple users. MySQL runs on Windows, OS X and UNIX and is the standard on most hosting applications. (Dev.mysql.com, 2015)

4.5 Frameworks

To assist with the development of the project some frameworks were used. Using frameworks helps to speed up the development process. Most frameworks are well documented and have a large community supporting them.

Bootstrap

Bootstrap is a responsive design framework which takes a mobile first approach to web design and uses a mix of grids and media queries to create responsive websites. Bootstrap is the most popular choice of responsive web frameworks. One of the reasons why it's the most popular is the fact that there are more CSS, Jquery and UI assets within Bootstrap than any other framework, making it a great choice for rapidly developing websites. (Mark Otto, 2015)

Laravel

Laravel is an open source framework for PHP. It is the most popular choice for PHP developers. Laravel is both lightweight and robust. It's packed with built in functionality and there's plenty of add-ons to extend the framework even further. Laravel is easy to install and it makes the PHP much more readable and easier to write. It offers a higher level of security and it's not too difficult to learn, although it can be a little over whelming to begin. Larvel is well documented and there is even a Netflix style website packed full of training material (Laracasts.com, 2015) for developers of all skill levels. Larval's setup requires some PHP plugins to be installed and enabled on the server. It uses an MVC approach to development, separating logic from presentation. This makes is easier to manage and maintain the project. (Otwell, 2015)

4.5 Libraries

JQuery Framework

JQuery was used for the front-end development to manipulate data and improve the user experience within the application. JQuery is a javascript framework and as it is one of the

more popular javascript frameworks, there is a wealth of documentation and supporting threads across the internet to aid in troubleshooting. (jquery.org, 2015)

Typekit

To serve up fonts for the project, Typekit font library was used. Typekit hosts fonts that can be used on websites. Fonts are included with a simple line of JS and can then be applied using CSS. Simple. (Typekit.com, 2015)

4.6 Feasibility Testing

A functional prototype was created that enabled a feasibility test for the project. The prototype covered the installation of Laravel and saw a user authentication system built. Styles were also implemented for views within the functional prototype.

What was learn't from building the functional prototype was how Laravel uses the MVC model to separate logic from presentation. The file structure and how routes are used to map urls to controllers.

Shortly after the creation of the Functional prototype a newer version of the Laravel framework (5) was realised which included many bug fixes and improvements. The release of the new version had an impact on the project and so it was decided that it would be used for the final build.

4.7 Challenges

Learning Laravel

As larval uses its own built in classes and methods to handle a lot of the functionality it can be difficult to understand and troubleshoot, this is where the documentation (Otwell,

2015) and vast library of training videos via - Laracasts (Laracasts.com, 2015) proved valuable.

The main challenges with learning to use the framework were related to:

- Artisan and composer
- Learning to use the blade tempting engine
- Handling authentication for users.
- How to create and manage database migrations

Using The Distance Matrix Api With A Function To Retrieve Store Offers

This was the most significant challenge for the build. In order to load offers into the front end application, the users location needed to be tested against store locations within the database. If the distance between the locations is less than 300 meters from a store location, offers linked to the store are loaded.

Challenges that needed to be overcame to achieve this:

- Integrating the distance matrix api within Laravel.
- Detecting the users location and comparing against store locations for multiple stores within the database.
- Loading offers if distance between user and store is less that 300 meters.

Geocoding And Reserve Geocoding

In order for business users to be able to create and manage stores the google maps geocoding service needed to be integrated into the system. The following needed to be achieved:

- Detect the users location and place a marker on the map element displayed within the view.
- Allow the user to update the location manually if the store location is different from the users current location.
- Allow the location to be set by dragging the marker to improve accuracy.
- Reverse geocode store location to display a meaningful address for each store within the stores view.

Category Images Function

During the UX design process it was decided each offer would have a banner image displayed with the offer details in the users feed. This created a problem were business users would of needed to upload an image for each offer. This would become time consuming, sourcing images and uploading to the system. It would have also had an impact on the clean design of the system as there could be no control over the quality of images being uploaded.

It was decided then that choosing a category for the offer would load a series of stock photos related to the topic. One of the images is then selected as the banner image for the offer.

For this to work the system needed to:

- Load images based upon the category chosen
- Link the image with the offer in the database so that the url can be accessed

Risk Analysis

It was imperative to review the project scope, aims and objectives in order to identify the risks, this allowed for consideration of the impact each risk would have on the project, should it occur. Steps were then taken in order to plan ahead for anything that could of went wrong during the build.

	Risk source		Probability			Impact			Result	Impact Areas			Mitigation / Response plan	Status
			Low	Medium	High	Low	Medium	High		Cost	Schedule	Performance		
1	Using the Larval Framework	Aaron C.		5				9	45		x	x	Install and configure the framework and get comfortable will how it works. Watch Laracasts (tutorial videos) to gain knowledge of the framework and it's tempting engine (blade). Build a user authentication system and test.	New
2	Integrating the GeoLocation API	Aaron C.	3					7	21			x	Integrate the api early in the project and experiment	New
3	Data Loss	Aaron C.	1					9	9		x		The Project will be hosted on Github, a tool a that tracks changes to files so that the project state can be rolled back if needed. Local files for the project will be stored on a Dropbox folder as, so that all files will be backed up should there be a hard drive failure with the computer used to develop the application, this will also be a good contingency plan for version control as dropbox allows users to roll back file changes.	New
4	Data Gathering	Aaron C.	2			3			6		x		Generate dummy data for the database tables and test.	New
5	PHP and MYSQL Knowledge	Aaron C.	2			3			6		x	x	Further enhance ability using PHP and MYSQL through study. Be actively involved in the PHP online community and use the documentation.	New

Risk Item Details

	Risk	Date Created	Created By	Date Updated	Updated By
1	Using the Larval Framework	16/01/2015 11:59	Aaron Colton	16/01/2015 11:59	Aaron Colton
2	Integrating the GeoLocation API	16/01/2015 12:01	Aaron Colton	16/01/2015 12:01	Aaron Colton
3	Data Loss	16/01/2015 12:03	Aaron Colton	16/01/2015 12:03	Aaron Colton
4	Data Gathering	16/01/2015 12:03	Aaron Colton	16/01/2015 12:03	Aaron Colton
5	PHP and MYSQL Knowledge	16/01/2015 12:05	Aaron Colton	16/01/2015 12:05	Aaron Colton

4.7 Achievements

Learning Laravel

To assist in the learning of Laravel the framework documentation (Otwell, 2015) and training videos were used extensively throughout the project.

Artisan is the command-line interface that is included within Laravel. Artisan provides a number of helpful commands to be used while developing an application. Composer is a PHP dependancy manager made use of within Laravel. The commands assist in the setup of some of the features within Laravel, i.e controllers, database migrations, database schemas etc.

The first step to setting up the framework is to install composer globally.
(Getcomposer.org, 2015) ***Figure 18***

```
curl -sS https://getcomposer.org/installer | php
mv composer.phar /usr/local/bin/composer
```

Once the composer installation is complete, Larvel can then be installed.
As stated previously, composer is a dependency manager, it assists with the installation of extensions within Larvel. A popular extension used in Larvel is 'Laravel-5-Generators-Extended' which was installed to assist in the setup of the projects database schema.
(GitHub, 2015) ***Figure 19***

```
composer require laracasts/generators --dev
```

Then database schemas and migration could now be created and managed using:

make:migration:schema
make:migration:pivot
make:seed

make:migration:schema was then used to create a database schema for offers. (GitHub, 2015)

Figure 20

```
php artisan make:migration:schema create_offers_table --schema="offer_title:string,
offer_desc:string"
```

when completed the database table, model and migration are created.

(Otwell, 2015)

Laravel also has its own tempting engine 'Blade' the template engine makes it simple to load and manage views across the site. A view is created and given a name along with the extension of 'blade.php'. If loading the same styles or scripts across multiple views this can be very useful. For example you can create a default.blade.php template then a home.blade.php template. With default.blade.php you would include your document head, body, footer etc.

Then, using the blade commands you can include content from views using the @yield('content') command in the default template. Within the home view you would have @extends('default'). then you would create a section @section('content') any thing contained in this section is load into our default template where the @yield('content') is used.

Figure 21

```
1  @extends('default')
2
3  @section('content')
4
5
6      <div class="container-fluid">
7          <div class="row">
8              <div class="col-sm-12">
9                  <h2>Admin Profile</h2>
10
11                 <p>Update your profile information.</p>
12                 <div id="message"></div>
13             </div>
14         </div>
15         <div class="row">
16             <div class="col-sm-12">
17                 @include ('partials.profileForm')
18             </div>
19         </div>
20     </div>
21
22 @endsection
```

Figure 22

(Otwell, 2015)

Authentication is almost completely configured out of the box within Laraval. For example The Auth:: Method can be used to verify a user when logging in. (Otwell, 2015)

Figure 23

Password protection is made simple by using the built in password hashing class within Laravel 'Hash'. While there is an initial learning curve to these methods within the framework they make for a more robust system and assist in rapid development for large projects. (Otwell, 2015)

Using The Distance Matrix Api And Writing A Function To Retrieve Store Offers

The distance matrix service will return an object based on values passed to the api. Implementing the api is the first step. There is a vast amount of documentation (Google Developers, 2015) on googles website to aid in the implementation of the api, along with instructions on how to retrieve data form the returned object. However, the api wasn't

able to do what was needed out of the box. This saw a custom function being written to pass data to and from the api, that could then be used to load offers into the users feed or return a message of no offers found.

Below is the process followed to create the distance checking features within the application.

Figure 24

```
var origin1 = new google.maps.LatLng();
origin1.D = 0;
origin1.k = 0;
```

First, a new google maps object is created. Then the values for latitude (origin1.D) and (origin1.k) longitude are set to '0'.

Figure 25

```
function getOffersPage(stores) {
    $.ajax({
        url: "/offers",
        method: "POST",
        data: {
            store_ids: stores,
            _token: $('#token').val()
        }
    }).success(function(data){
        $('#outputDiv').html(data);
    });

    setTimeout(function(){ checkLocation() },20000);
}
```

A function getOffersPage was created to retrieve all of the offers from the stores passed to the function. The data returned contains a view constructed using a foreach loop that returns all of these offers from the database, each as an <li> element. setTimeout is a built in JS function, it was used to check the users location every 20 seconds.

Figure 26

```
function checkLocation() {

    if(navigator.geolocation) {
      navigator.geolocation.getCurrentPosition(function(position) {

          newOrigin = new google.maps.LatLng(position.coords.latitude,position.coords.longitude);

          if ((Math.round(origin1.D*10000)/10000 != Math.round(newOrigin.D*10000)/10000) &&
          (Math.round(origin1.k*10000)/10000 != Math.round(newOrigin.k*10000)/10000)) {

              origin1 = newOrigin;
```

The checkLocation function was then created to be passed to the setTimeout function. The function uses HTML5 geolocation to get the users current location. If a location is set, the newOrigin variable will be equal to the users latitude and longitude. An if statement was then created to compare the values against the previous location (origin1.D and origin1.k). Math.round was used to check the location accurately to meters as opposed to a higher degree of accuracy, as initial results indicated the user was moving by millimetres and so the application was constantly updating which was straining the server.

Figure 27

```
                $.ajax({
                    url: "/feed",
                    method: "POST",
                    data: {
                        lat: origin1.k,
                        lng: origin1.D,
                        _token: $('#token').val()
                    }
                }).success(function(data){
                    getOffersPage(data);
                });
            }

        }, function() {
            handleNoGeolocation(true);
        });
      } else {
        // Browser doesn't support Geolocation
        handleNoGeolocation(false);
      }
    }
```

If a change has been detected in a users position the ajax call will post the location to the 'feed Route'. The route is mapped to a controller containing a function **getOffersInArea.**

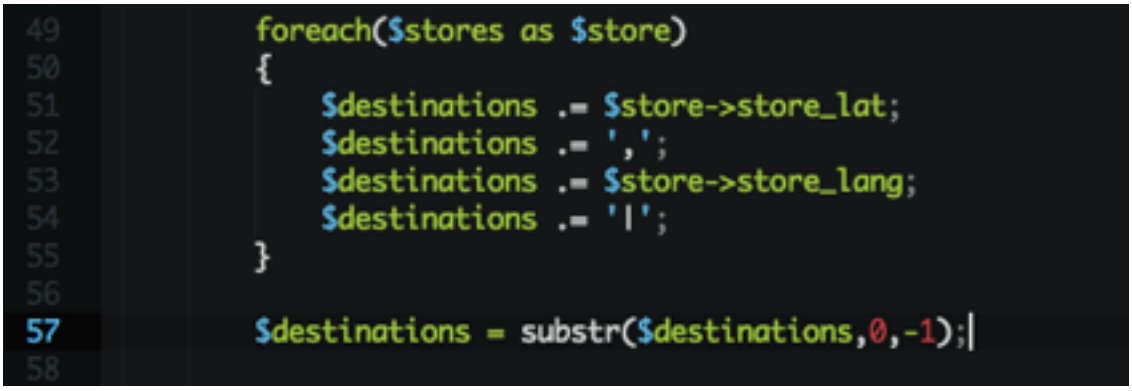
Figure 28

```
public function getOffersInArea()
{
    $data = Request::all();

    $origin = $data['lat'].','.$data['lng'];

    $stores = Store::all();

    $destinations = '';
```

The function **getOffersInArea** fetches offers linked to a store, where the distance between a store and a users location is less than 300 meters. First the data is requested using a built in Laravel method Request::all. $origin is set equal to the lat and lang values.

$stores is equal to the store data. The container $destinations has an empty value, the value is updated later in the function.

Figure 29



```
foreach($stores as $store)
{
    $destinations .= $store->store_lat;
    $destinations .= ',';
    $destinations .= $store->store_lang;
    $destinations .= '|';
}

$destinations = substr($destinations,0,-1);
```

A foreach then loops through all of the stores, concatenating each of the lat and lng values as a string . A string is the required format for passing the data to the distance matrix api. Substring is used to strip out the pipe character.

Figure 30

```
        $json =
        file_get_contents('https://maps.googleapis.com/maps/api/distancematrix/json?origins
        ='.$origin.'&destinations='.$destinations.'');

        $obj = json_decode($json);

        $nearbyStoresWithOffers = [];
```

The $origin which is equal to the users location and $destinations; equal to all of the store locations within the database are both passed to distance matrix url. The api then returns an object.

Figure 31

```
        $i = 0;

        foreach($obj->rows[0]->elements as $offerDistance)
        {

            if ($offerDistance->distance->value < 300)
            {
                array_push($nearbyStoresWithOffers,$i);
            }

            $i++;
        }

        $storesToReturn = [];

        foreach ($nearbyStoresWithOffers as $storeId)
        {
            array_push($storesToReturn,$stores[$storeId]->id);
        }


        return $storesToReturn;
    }

}
```

A foreach then loops through each of the elements returned in the object, then checks if the distance value is less than 300. For Each distance returned as true the index of the store is then pushed to the empty array $nearbyStoresWithOffers. A second foreach then loops

through the stores returned and gets the store id, based on the stores index within the array.

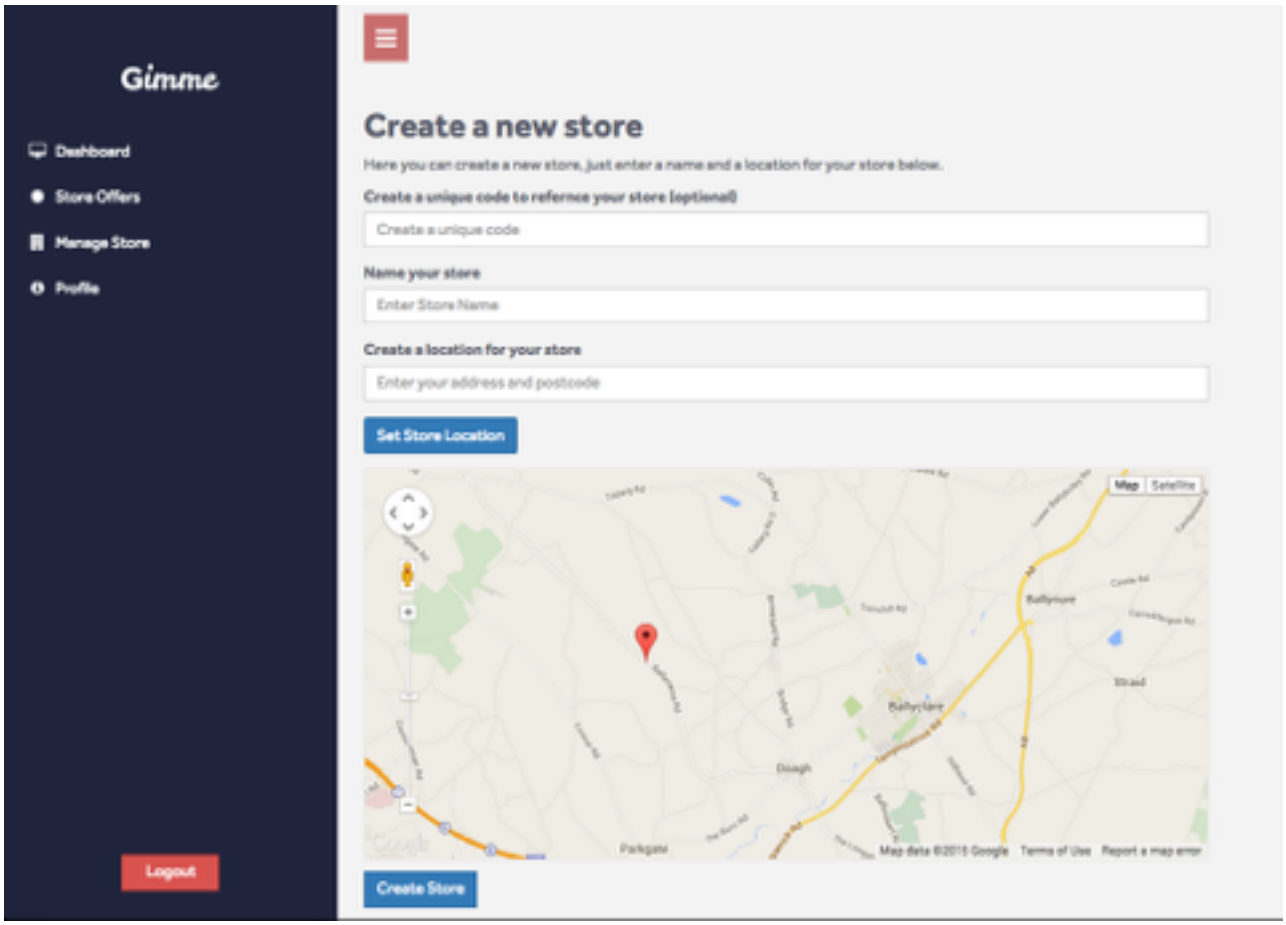
Figure 32

Finally within the JS, a function is created to handle errors with geolocation. The check location function is then called so that it runs on the first page load.

Geocoding And Reserve Geocoding

The google maps api (Google Developers, 2015) is fairly straight forward to implement so this section of the report will focus more on the unique way in which the api is being used, rather than the out of the box functionality.

Figure 33



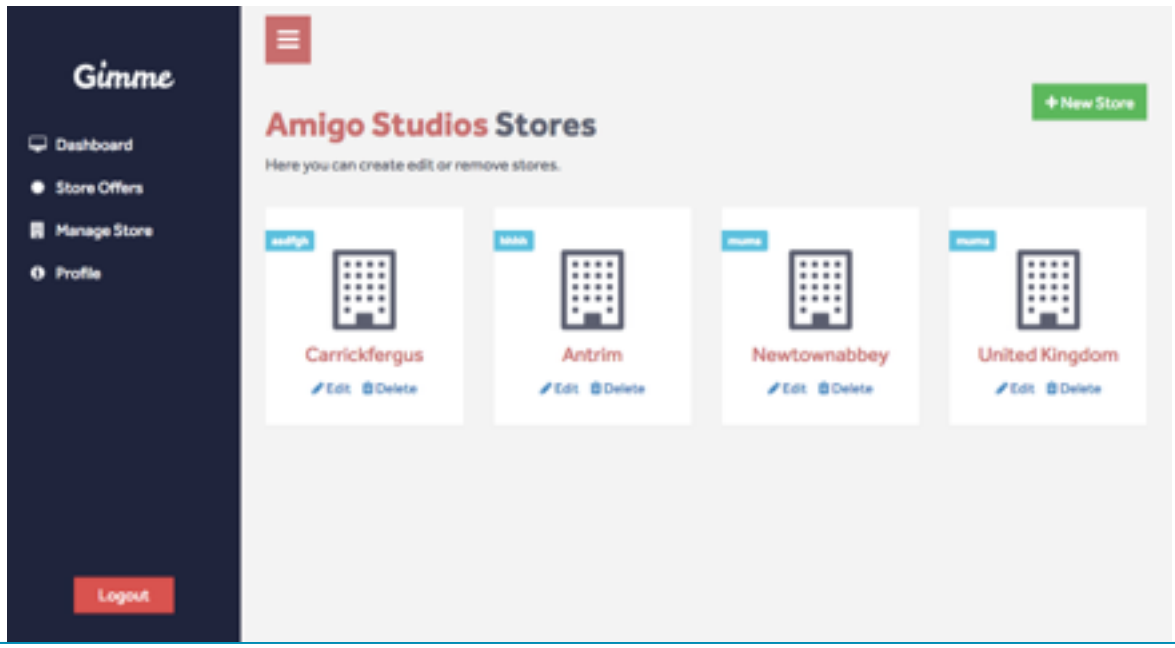
When creating a new store, a unique identifier can be set but is not required. This is useful if there are multiple stores within the same region. The next required field is the store name. The map marker is set automatically to the users current location using HTML5 geolocation. (W3schools.com, 2015) Entering the location into the address field and clicking the 'Set Store Location Button' updates the marker location on the map. The properties of the marker can also be updated by dragging the marker to a new location.

Figure 35

```
var geocoder;
var storeLocation = [];
var map;
var marker;

function setLocation(storeLoc) {
    var latlng = new google.maps.LatLng(storeLoc.k, storeLoc.D);
    geocoder.geocode({'latLng': latlng}, function(results, status) {
      if (status == google.maps.GeocoderStatus.OK) {
        if (results[1]) {
            storeLocation.lat = storeLoc.k;
            storeLocation.lng = storeLoc.D;
            storeLocation.region = results[1].address_components[3].long_name;
        }
      } else {
        alert("Geocoder failed due to: " + status);
      }
    });
}
```

The function updates the global storeLocation variable with an array containing the lat, lang and region. The region is obtained by reverse geocoding the lat/lang passed to the function and fetching the 4th field of the returned address. The values of the array are then stored within the database once the create store form is submitted.

Figure 36



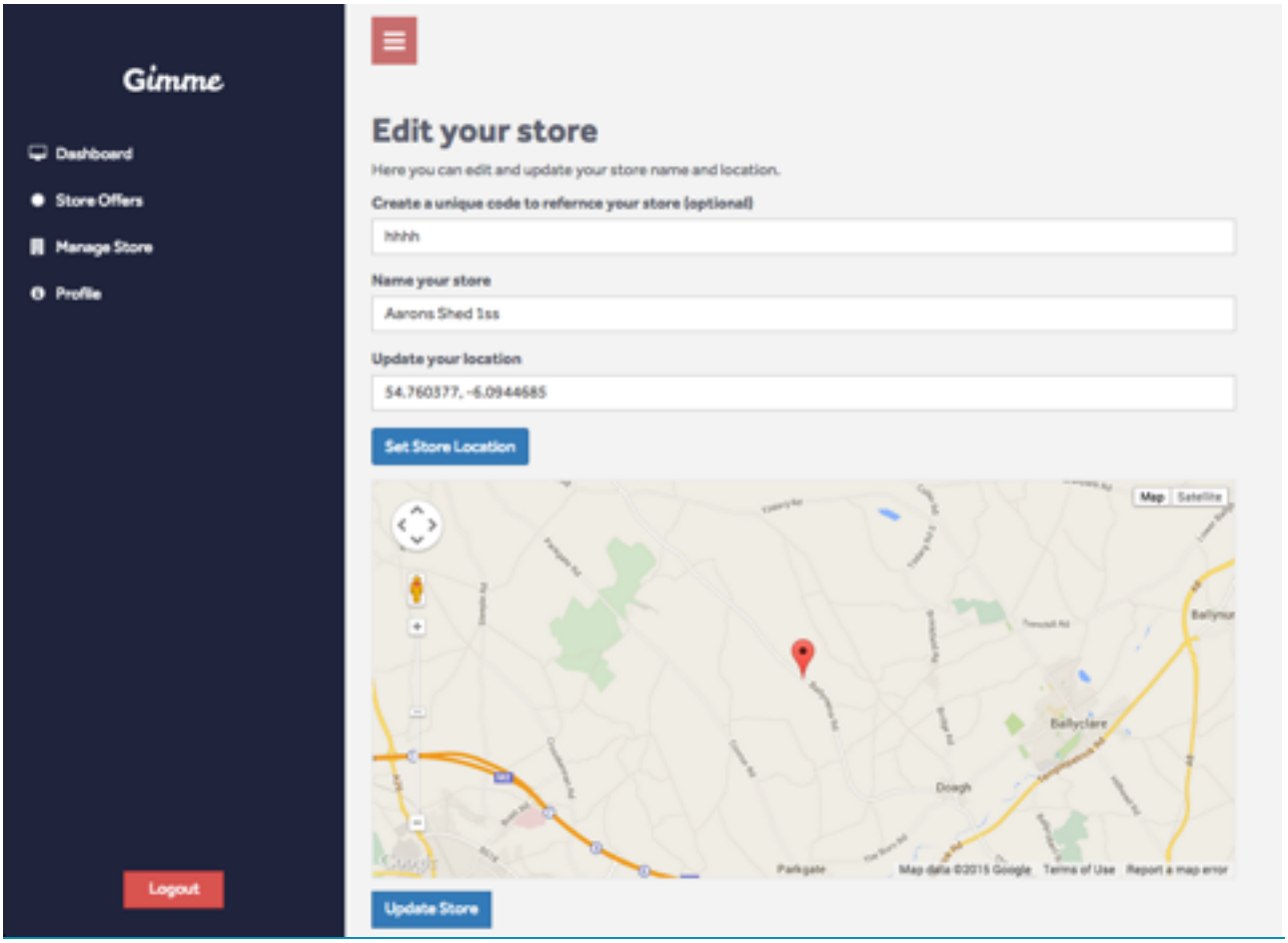
Existing stores can be edited or removed on the manage stores page. This view is where the stores region, created using the reverse geocoding, is displayed.

Figure 37

```
41
42  if ($('#store-address').val() !== "" ) {
43
44      $('#storeFormButton').click();
45
46      }
47
```

Editing a store location will load the store form. The form fields are populated with the information already stored within the database. An if statement is run to check if the location input value is set. If set the click function is run for the button, which will use the geocode function to update the marker on the map.

Figure 38



Category Images

Figure 39

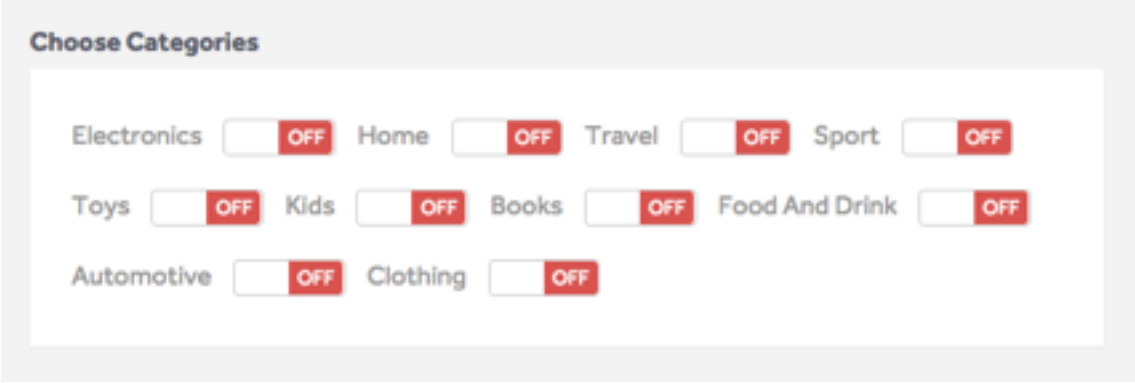
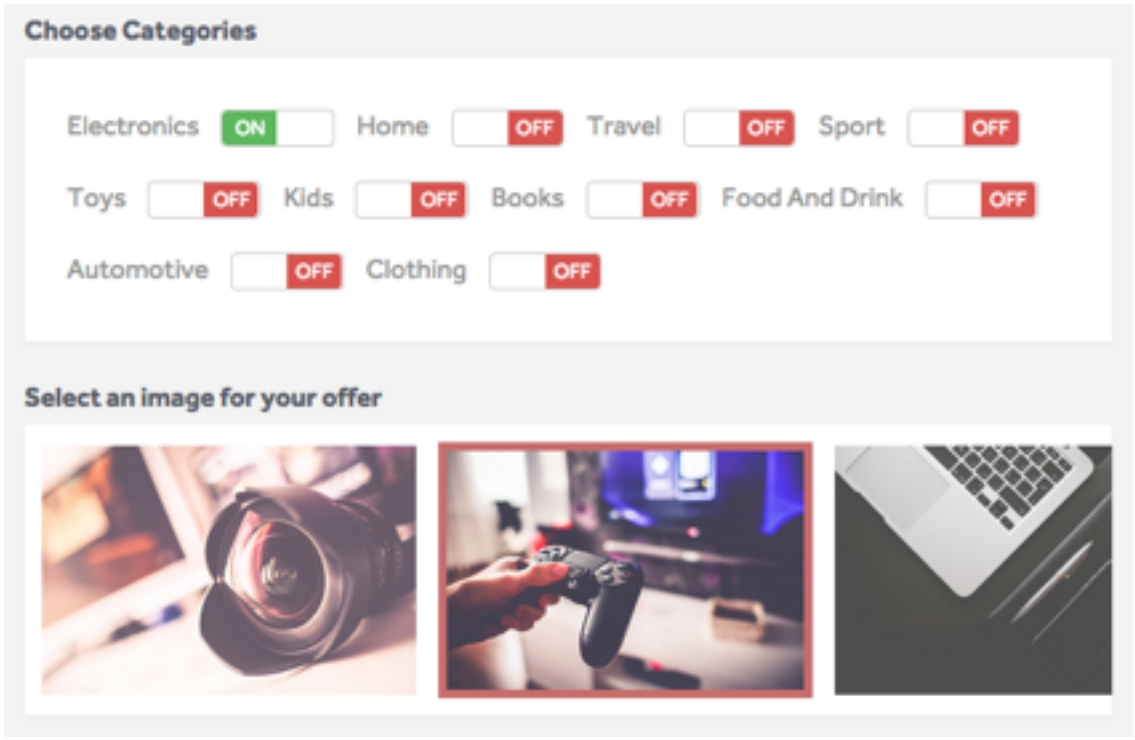


Figure 40



When creating an offer within the backend system, categories for the offer are set.

Depending on the categories set, a horizontally scrollable panel will appear containing a series of images that relate to the chosen category or categories.

Figure 41

```
function getImages(data) {
    $.ajax({
      url: "/fetch-images",
      type: "POST",
      data: {
        "_token": $("#token").val(),
        "categories": data
      }
    }).done(function(data) {
        $('#category-images').html(data);
    }).error(function(data) {

        $.each( data.responseJSON, function( index, value ){
            $('#category-images').append("<p class='alert alert-danger'>" + value + "<button type='button'
            class='close' data-dismiss='alert' aria-label='Close'>" + "<span aria-hidden='true'>&times;</span>" +
            "</button>" + "</p>");

        });
    });
}

$('input[name="cat[]"]').on('switchChange.bootstrapSwitch', function(e) {

    var data = [];

    $('input[name="cat[]"]:checked').each(function(){
        data.push($(this).val());
    });

    getImages(data);

    e.preventDefault();
});
```

Toggling the category switches will run a JQuery Ajax function. The function passes the data through the Route to the fetchImages method within the imagesController.

Figure 42

```
public function fetchImages($id = NULL) {


    $categories = Request::only('categories');

    $imagesCollection = [];

    if ($categories['categories']) {
        foreach ($categories['categories'] as $category) {
            $images = Image::where('category_id', '=', $category)->get()->toArray();
            $imagesCollection = array_merge($imagesCollection, $images);
        }
    }

    return view('partials.ImageSelect', ['images' => (object)$imagesCollection, 'activeId' => $id]);

}
```

A function **fetchImages** is contained with in the imageController, when called, the function will first request the categories posted form the ajax function and stores them in

the variable $categories. An if statement will check to see if categories have been passed to the function and then if true, loops through each of these categories. It then fetches the images from the database that have that categories id and adds them to the $imagesCollection before passing them to the imageSelect partial view.

Figure 43

```
@if (count((array)$images) > 0)
    <label>Select an image for your offer</label>
        <div id="image-sel-container" class="form-group">
            <div class="panel default-panel">
                @foreach ($images as $image)

                <input type="radio" name="image_id" value="{{
                $image['id'] }}" id="imgSel{{ $image['id'] }}" @if
                ($image['id'] == $activeId) {{ 'checked' }} @endif />
                <label for="imgSel{{ $image['id'] }}">
                    <img src="{{ asset($image['image_url']) }}"
                    alt="thumbnail" />
                </label>

                @endforeach
            </div>
        </div>
@endif
```

The data is now available within the view. The diagram shows the code constructed to populate the panel with the images. The images are contained within <label> elements for hidden radio inputs. This means that when an image is selected, the radio button connected to the label will be checked. This is how the image id is being posted with the form. The image id is then linked with the offer using a pivot table.

5. Testing

5.1 Testing approach

Testing is an important part in the development process. This part of the report will focus on the testing of the requirements for the application. Depending on the application to be tested, different methods can be applied.

White Box Testing

White box testing, also known as 'clear box' is used within software development to test the inner workings of an application. It is used to test the systems inputs and outputs when all of the code within the system is visible and clear. (Softwaretestingclass.com, 2015)

Black Box Testing

Black testing method is used when all of the inner workings of a system is not clear. Knowledge of programming is not required to carryout black box testing.
(Softwaretestingclass.com, 2015)

Gray Box Testing

Gray box testing is used whenever only parts if the inner workings of a system are clear. With grey box testing, access to the underlying code of the system is not required. It is best suitable for testing web applications.
(Softwaretestingclass.com, 2015)

5.2 Testing Process

Grey box testing was used to test the system against the requirements as not all of the inner workings of the system are clear. It was also chosen for the fact that it is best suited for testing web applications and can be used by both users and developers.

5.3 Test Results

Testing The Usability

Usability testing was carried out through the application to ensure the usability requirements were met.

The first step in the usability test was to ensure that all parts of the application were loading correctly and that the full application was available to users as stated in the requirements specification.

A simple check was performed by navigating to the web application and clicking thorough all of the pages and interactions to ensure everything appeared as expected.

The views all loaded with the correct data as expected :

Figure 44

Figure 45

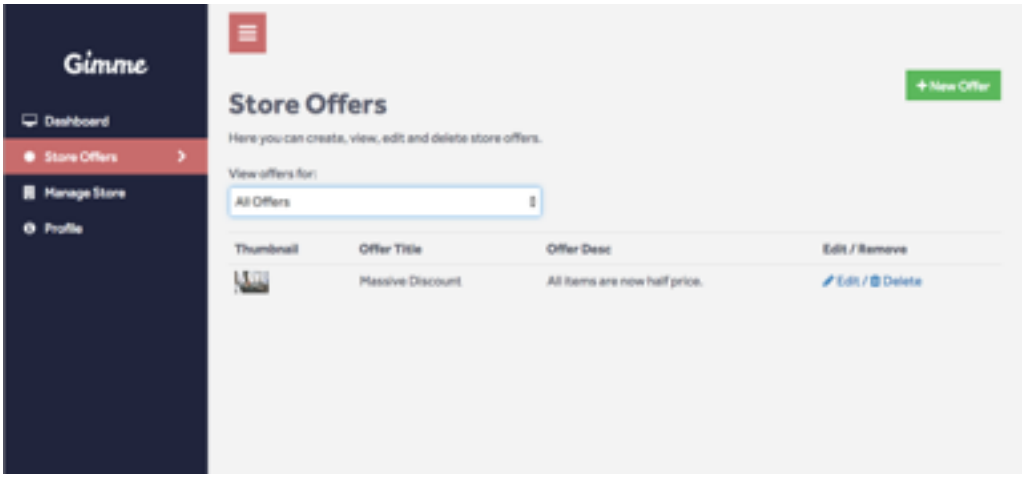


Figure 46

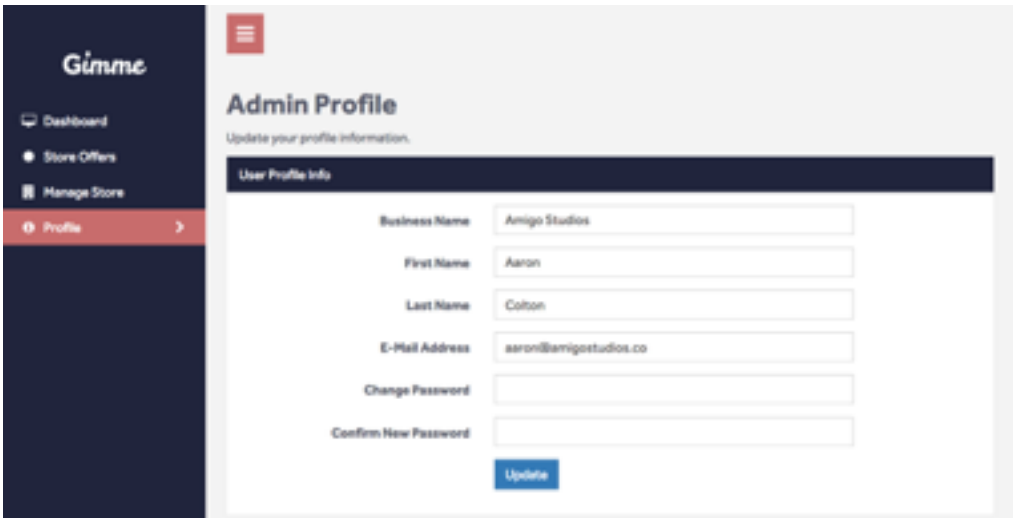


Figure 47

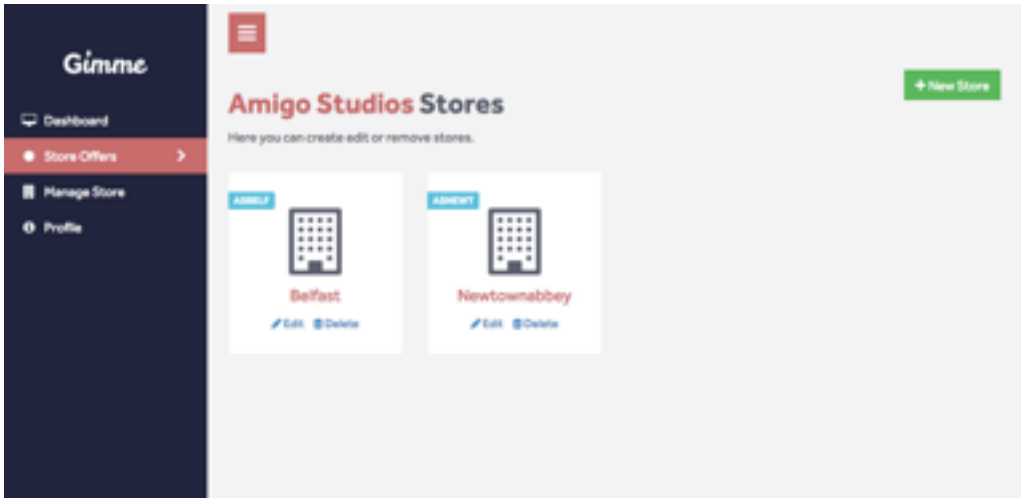


Figure 48

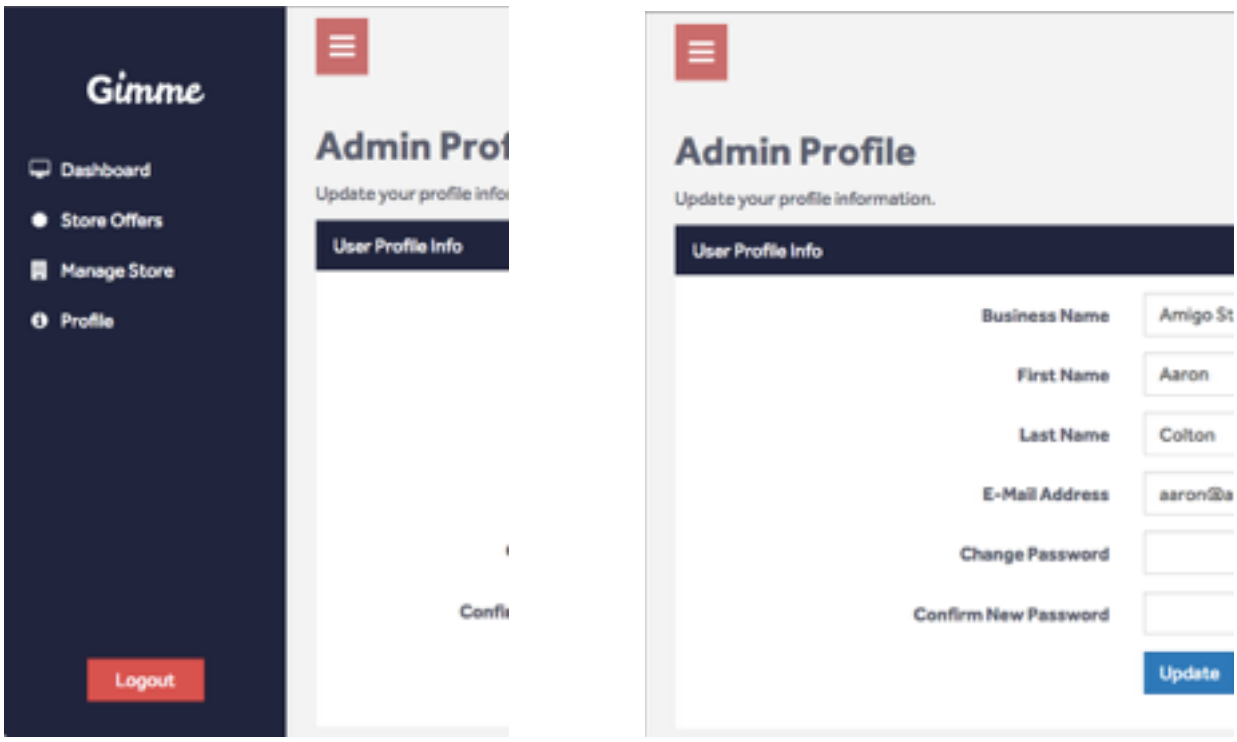


Figure 48 shows that pressing the menu button closes and opens the menu as is expected. The page column widths change as the menu closes and opens also as expected.

Figure 49

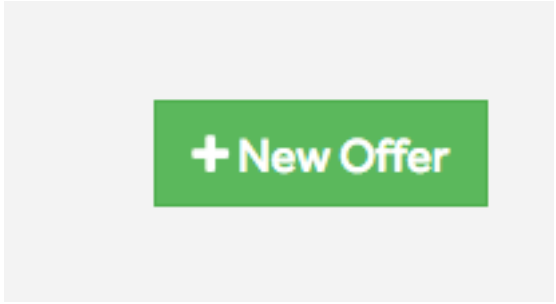


Figure 49 the user shows the call to action to create a new offer. When clicked the user will expect to see a form which they can fill out to create a new offer. The button click was tested to ensure the function was being run successfully. If the button done nothing, it would show that something had went wrong with the system.If the button was pressed and the form was returned successfully it could be concluded that the functionality was working as expected.

When pressed the button return the form successfully.

Testing Functionality

Form validation needed to be tested to ensure that relevant error messages were being displayed to the user if the form processing failed. The test was carried out on all forms by submitting the form with incorrect data, correct data, missing data and no data and then checking that each response returned was correct.

Figure 50

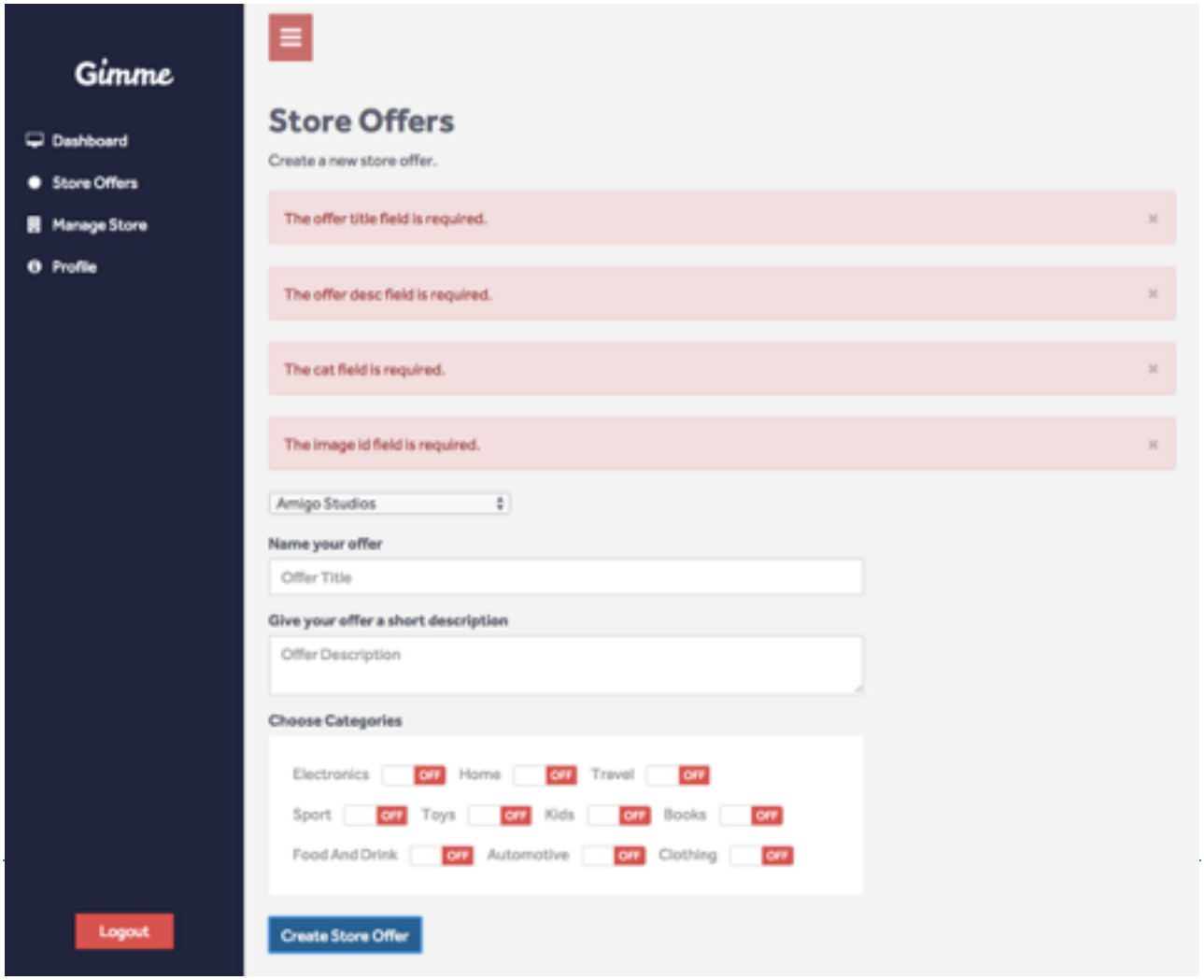
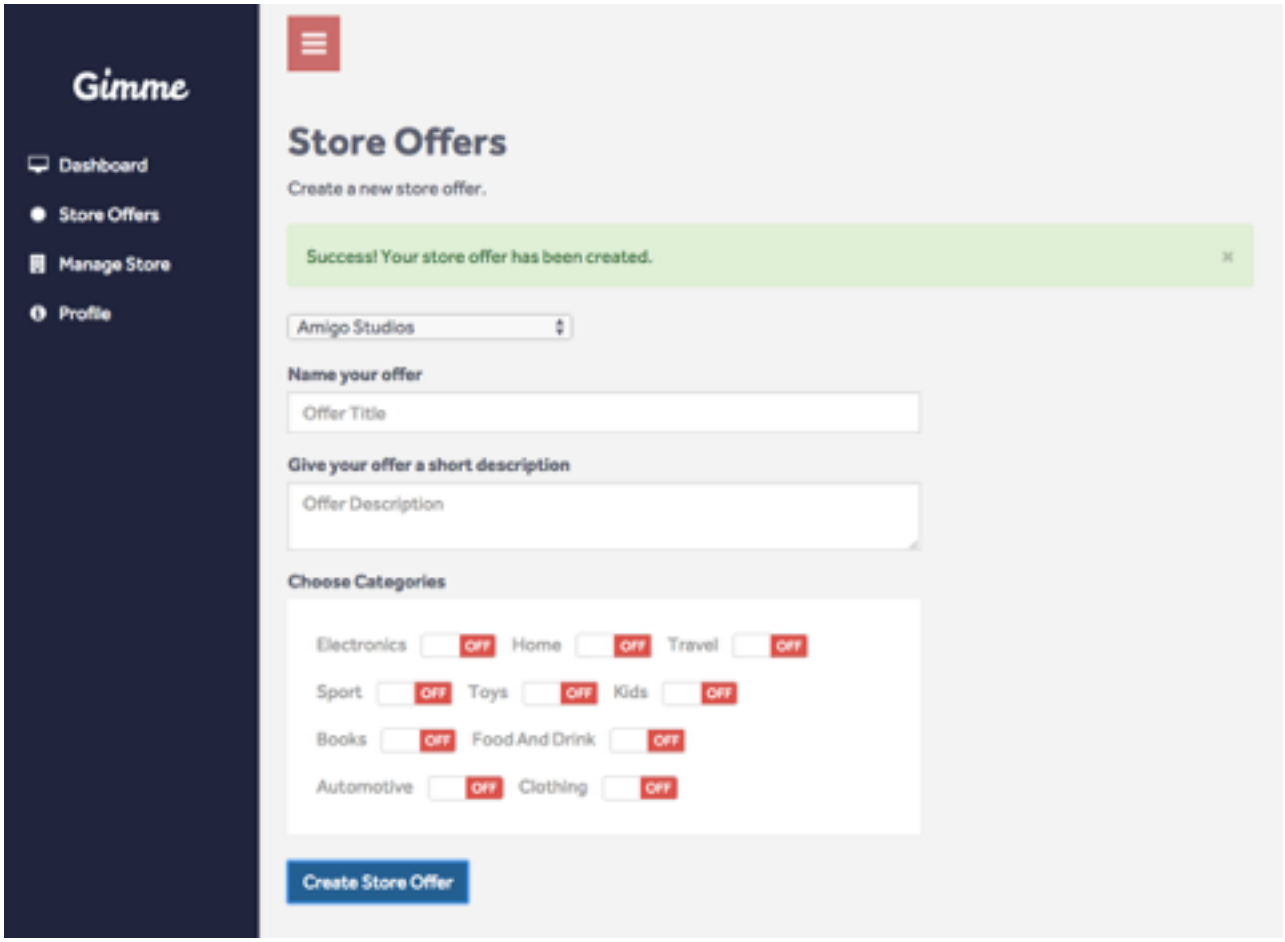


Figure 51



Api Testing

To test the functionality of the Api's used the functions were tested by using the application. The initial tests showed that the functions worked as expected. However, during further testing later in the project, it was discovered that functions making use of the distance matrix service and the geocoding service were no longer working.

After further testing and research it was discovered that google changes the naming of the lat and lang values within the api periodically. This meant that every function that made use of the pos.D and pos.k properties would no longer work as the values had changed to pos.A pos.D.

Anywhere these properties were used needed to be update to use Lat() and Lng() which would return the correct values anytime the api is updated.

Web Page Performance TestIng

Testing the web page performance is important to ensure that pages and resources are are being load with acceptable response times. The applications page performance was tested using the popular online testing tool 'Pingdom'. (Pingdom.com, 2015)

Figure 52

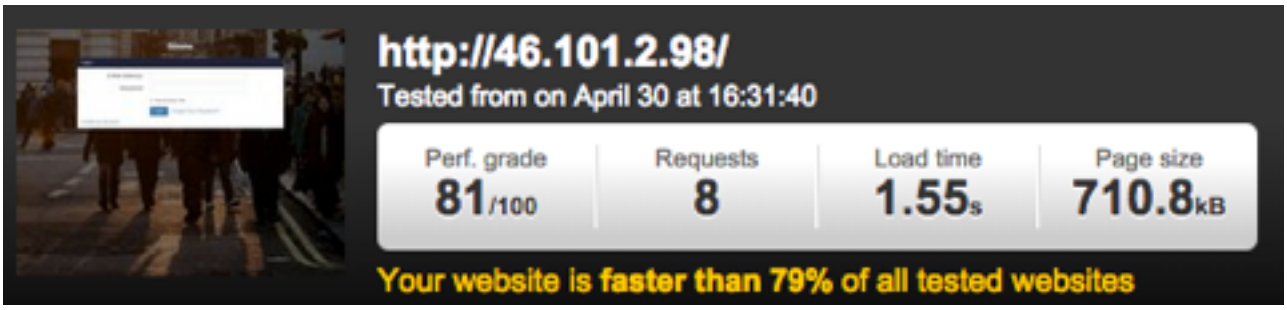
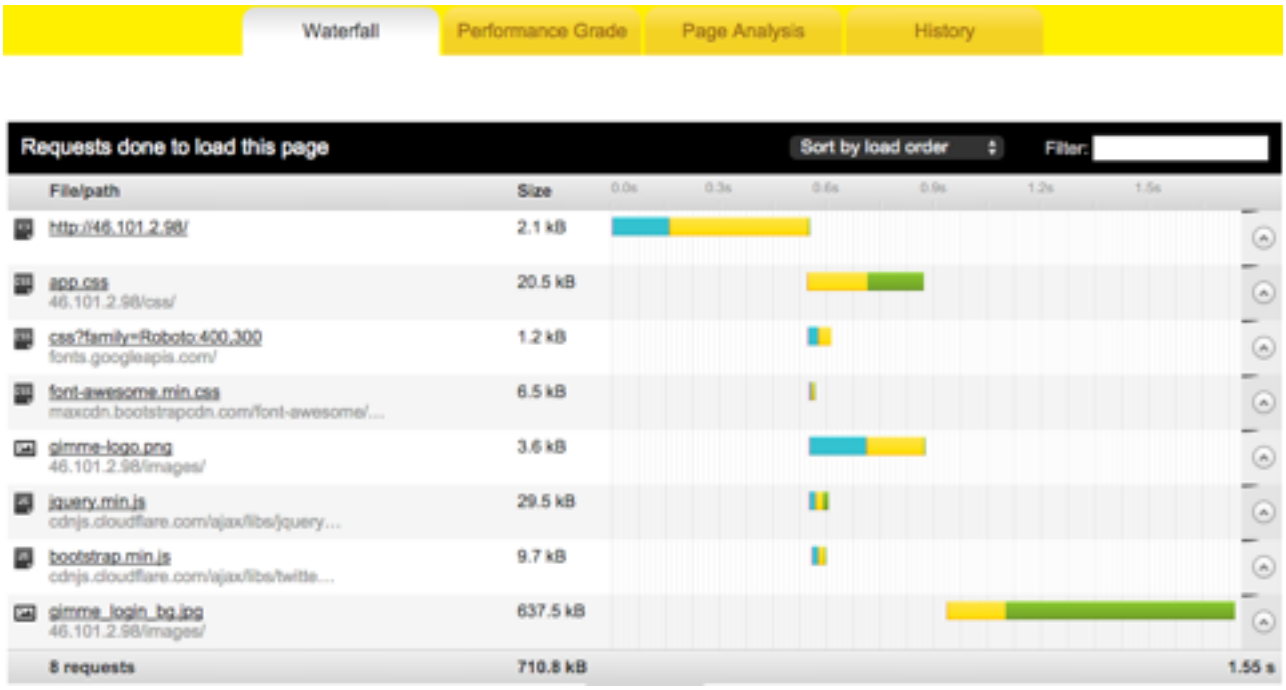


Figure 53



Testing The Web Pages For Cross-Browser Compatibility

To ensure that the user experience is the same across different browsers cross-browser compatibility testing was carried out.

The application was tested in the most current and modern web browsers. Some older versions of Internet Explorer were also tested. the results showed that the HTML and CSS for the application was robust and consistent across most browsers. Only two bugs were discovered.

Bug: Unexpected behaviour with navigation link
Description: When clicking the navigation item 'Manage stores' the link opens in a new window.
Browser: Internet Explorer 9
Solution: Correct the target attribute.

Bug: Positioning issue with menu container
Description: The menu appears on the right of screen, should appear on the left.
Browser: Mozilla Firefox 36.0.4
Solution: left positioning needed to be set to '0'.

5.4 User Surveys

To further test the usability and to measure the requirements of the application, a short user survey was created that asked 5 simple questions. The questions were carefully chosen to assess the core requirements.

Question 1 asked:

What are your thoughts on the look and feel of the application?

Tests requirement: 028

Figure 54

028	Non-Functional	M	The application needs to be ascetically pleasing	Users will want the application to have a nice look and feel	The design of the application will be pleasing.	

Question 2 asked:

Does the application work how you would expect it to?

Tests requirement: 029

Figure 55

029	Non-Functional	M	The application must be simple to use	Needs to have a good user experience design	User will be able to use the application without confusion.	

Question 3 asked:

Was the admin system clear and easy to use?

Tests requirement: 030

Figure 56

030	Non-Functional	M	The application must be simple to administer	Business user need a simplified admin experience	The admin system will be simple and will not lead to confusion.	

Question 4 asked:

Were you able to login successfully?

Tests requirement: 003

Figure 57

003	Functional	H	The application will prompt the user to login	Registered users will need to be able to login	The system will verify the user account or return an error	2

Question 5 asked:

Upon logging in, did the system redirect you to the admin dashboard?

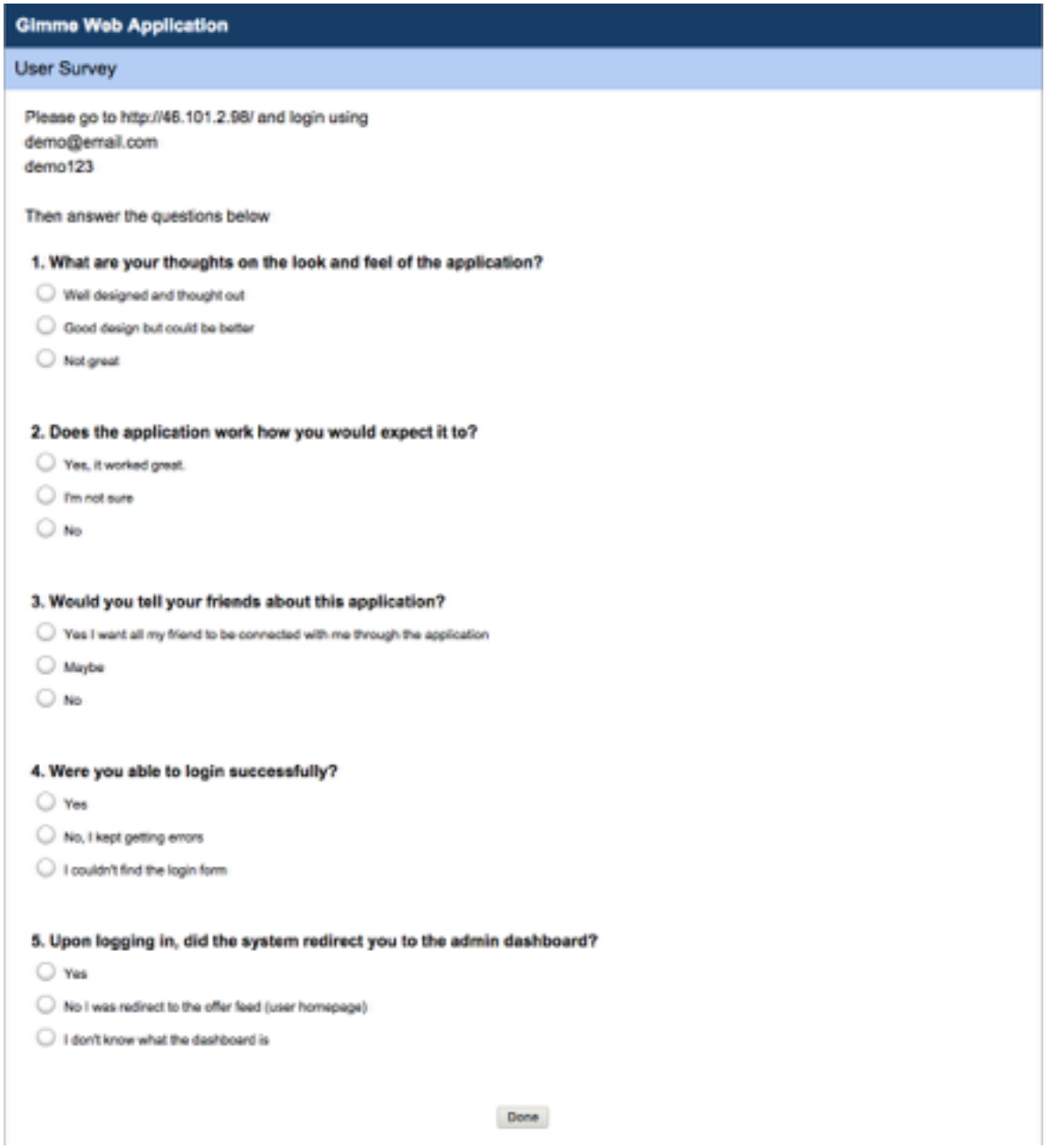
Tests requirement: 004

Figure 58

004	Functional	M	The application will determine the data to be displayed	Business users want to view different information than standard users	Users will receive different datasets depending on the account type	1

The survey was then created using Survey Monkey (Surveymonkey.com, 2015) and sent to multiple users to collect responses.

Figure 59

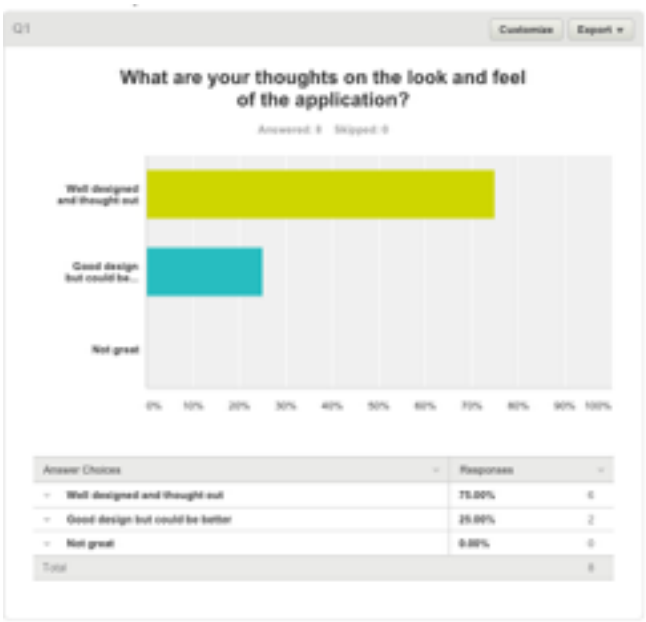

Gimme Web Application
User Survey
Please go to http://46.101.2.98/ and login using
demo@email.com
demo123
Then answer the questions below
1. What are your thoughts on the look and feel of the application?
Well designed and thought out
Good design but could be better
Not great
2. Does the application work how you would expect it to?
Yes, it worked great.
I'm not sure
No
3. Would you tell your friends about this application?
Yes I want all my friend to be connected with me through the application
Maybe
No
4. Were you able to login successfully?
Yes
No, I kept getting errors
I couldn't find the login form
5. Upon logging in, did the system redirect you to the admin dashboard?
Yes
No I was redirect to the offer feed (user homepage)
I don't know what the dashboard is
Done

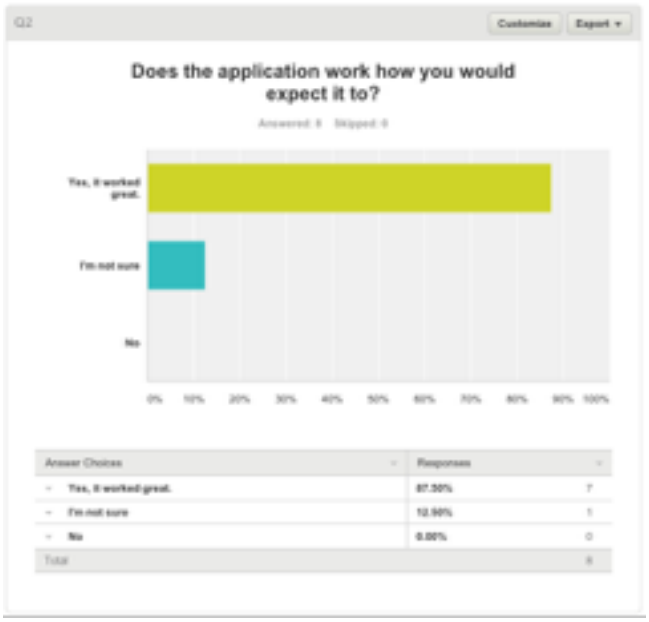
6. Evaluation

6.1 Survey Results

All of the users who were asked to complete the survey did so. This helped to determine if the requirements tested through the survey had been achieved.

Figure 60





Q1. What are your thoughts on the look and feel of the application?

All of the users answered positively in relation to requirement 028. 75% felt that the applications look and fell was well designed and thought out. 25% indicated that they felt the design was Good but there were room for improvements. No users felt that the design was not great. The feedback determines that the requirement has been met successfully.

Q2. Does the application work how you would expect it to?

Almost all of the users answered positively in relation to requirement 029. An overwelling 87.50% thought the application worked as it was expected to. 12.50% were not sure. This question was perhaps a little ambiguous and may have been confusing for the user. With

such a high number answering 'Yes, it worked well.' It is safe to determine that requirement was successfully met.

The remainder of the survey saw users responding positively to the admin side of the system and agreeing that they would happily share the application amongst friends. This is a good measure when considering releasing the application to the public.

6.2 Project Outcomes

The completion of the project delivered a web application that allows businesses to advertise their promotional offers within a specific region. General users of the application are able to view and claim offers available with their location.

The front-end of the application was developed specifically for mobile users. The web application can be installed to a users home screen, mimicking an app.

Businesses are able to sign up for an account. Business administrators have the ability to login to the backend system were they can create and add store locations for the business. Offers can be created and assigned to specific store locations for the business account. Store locations are editable as are offers allowing for easy management.

Users of the application have the ability to register for an account, allowing them to use the application to search for available offers within their location ,using their mobile device. They are then able to claim and offer and redeem it in-store using a digital voucher.

All of the high priority requirements were met with the exception of one. Only part of the requirement was completed which saw the ability to claim offers being created but not the ability to favourite the offers. This was due to unforeseen circumstances and time

constraints during the project development. As the project followed a modified waterfall approach it allowed for small changes to build. There were also no high priority requirements depending on this particular requirement, meaning the development was able to continue without this having a profound impact.

7. Conclusion

7.1 Summary

Developing the web application saw many new challenges present themselves, meaning there was a lot of new learning to be done in order to deliver a completed system. The challenges although not always easy were exciting and enjoyable for the most part. The process followed for the project made it easier manage and build. having no previous knowledge with Laravel or the google maps api these posed significant challenges. Having now used these technologies to build the application, skills have been enhanced in both ability to develop such systems and knowledge.

7.2 Reflection

Overall, the development of gimme went well. Creating a plan and being aware of the challenges to be met during the development helped to see it through to completion. Using the modified waterfall approach proved suitable for the project. Developing the project in a linear fashion worked well with the time frame allowed to complete each of the requirements. As it was a modified waterfall approach, it also allowed for small changes as the scope need to be reduced. A signification development challenge was overcome and skills have enhanced that will enable such systems to be developed easier and with better understanding in the future.

Ideally more of the requirements would have been completed but time constraints and unforeseen circumstances meant it wasn't possible to build the full scope of the project. The main cause of unforeseen circumstances that affected the project was running and managing a business. Steps had been taken to reduce the business demands, work was stopped at the beginning of April, only leaving commitments to current clients in terms of support and maintenance to be managed. However, if work had been stopped at the beginning of march this would have been better. The risks, although assessed early in the project and all possible steps being taken to mediate these, still saw the need for the scope to be reduced.

7.3 The Future

What does the future hold for Gimme? It is hoped that development will continue on the application to see all the requirements that haven't been met integrated. Further refinements could then be made to the user experience and once the application reached an acceptable level, it would be released for beta testing. Testing would likely be focused on a small group until feedback is gathered allowing further refinements. Once the application has been developed to a working state with all of the attractive features integrated, it could then be released to the public. The application would then hopefully enable a business to be evolved around the service.

8. References

Advertising Industry Statistics, (2015). Advertising Association - promoting and protecting everyone in adland - advertisers, media owners and agencies. [online] Available at: Available: http://www.adassoc.org.uk/Advertising-statistics [Accessed 30 Apr. 2015].

Blog.karachicorner.com, (2015). Mobile UI Design: 60 Outstanding Examples for Inspiration | Inspiration | Design Blog. [online] Available at: http://blog.karachicorner.com/2013/06/mobile-ui-design/ [Accessed 1 May 2015].

Digitalstrategyconsulting.com, (2015). Digital shoppers now expecting more from high street shops - Digital Intelligence daily digital marketing research. [online] Available at: http://www.digitalstrategyconsulting.com/intelligence/2014/11/digital_shoppers_now_expecting_more_from_high_street_shops.php. [Accessed 1 May 2015].

Getcomposer.org, (2015). Composer. [online] Available at: https://getcomposer.org/ [Accessed 1 May 2015].

GitHub, (2015). laracasts/Laravel-5-Generators-Extended. [online] Available at: https://github.com/laracasts/Laravel-5-Generators-Extended [Accessed 1 May 2015].

Google Developers, (2015). Google Maps Web APIs — Google Developers. [online] Available at: https://developers.google.com/maps/web/ [Accessed 1 May 2015].

jquery.org, j. (2015). jQuery. [online] Jquery.com. Available at: http://jquery.com/ [Accessed 1 May 2015].

Laracasts.com, (2015). The Best Laravel and PHP Screencasts. [online] Available at: https://laracasts.com/ [Accessed 1 May 2015].

Magazine, E. (2015). How Supermarkets Are Influencing The Behaviour Of 'Shopper Mums' | ESM Magazine. [online] ESM Magazine. Available at: http://www.esmmagazine.com/ 201409063667/Retail/How-Supermarkets-Are-Influencing-the-Behaviour-of-Shopper-Mums.html [Accessed 30 Apr. 2015].

Mark Otto, a. (2015). Bootstrap · The world's most popular mobile-first and responsive front-end framework.. [online] Getbootstrap.com. Available at: http://getbootstrap.com [Accessed 1 May 2015].

Mozilla Developer Network, (2015). About JavaScript. [online] Available at: https://developer.mozilla.org/en-US/docs/Web/JavaScript/About_JavaScript [Accessed 30 Apr. 2015].

Mozilla Developer Network, (2015). CSS basics. [online] Available at: https://developer.mozilla.org/en-US/Learn/Getting_started_with_the_web/CSS_basics [Accessed 30 Apr. 2015].

Otwell, T. (2015). Laravel - The PHP Framework For Web Artisans. [online] Laravel.com. Available at: http://laravel.com/ [Accessed 1 May 2015].

Phpbbhq.com, (2015). The Background, History and Development of PHP. [online] Available at: http://www.phpbbhq.com/developmentofphp.php [Accessed 30 Apr. 2015].

Pingdom.com, (2015). Pingdom - Website Monitoring. [online] Available at: https://www.pingdom.com/ [Accessed 1 May 2015].

Ruby-lang.org, (2015). Ruby Programming Language. [online] Available at: https://www.ruby-lang.org/en/ [Accessed 30 Apr. 2015].

Segue Technologies, (2013). Waterfall vs. Agile: Which is the Right Development Methodology for Your Project?. [online] Available at: http://www.seguetech.com/blog/2013/07/05/waterfall-vs-agile-right-development-methodology [Accessed 30 Apr. 2015].

Softwaretestingclass.com, (2015). What is a White Box Testing? | Software Testing Class. [online] Available at: http://www.softwaretestingclass.com/white-box-testing/ [Accessed 1 May 2015].

Softwaretestingclass.com, (2015). What is Black Box Testing? | Software Testing Class. [online] Available at: http://www.softwaretestingclass.com/what-is-black-box-testing/ [Accessed 1 May 2015].

Softwaretestingclass.com, (2015). What is Gray Box Testing? | Software Testing Class | Software Testing Class. [online] Available at: http://www.softwaretestingclass.com/gray-box-testing/ [Accessed 1 May 2015].

Surveymonkey.com, (2015). SurveyMonkey: Free online survey software & questionnaire tool. [online] Available at: https://www.surveymonkey.com/ [Accessed 1 May 2015].

Teamwork.com, (2015). Teamwork.com - The Best way to to Manage Your Projects & Team. [online] Available at: https://www.teamwork.com/ [Accessed 1 May 2015].

Typekit.com, (2015). Typekit. [online] Available at: https://typekit.com/ [Accessed 1 May 2015].

Volere.co.uk, (2015). Requirements Specification Template. [online] Available at: http://www.volere.co.uk/template.htm [Accessed 1 May 2015].

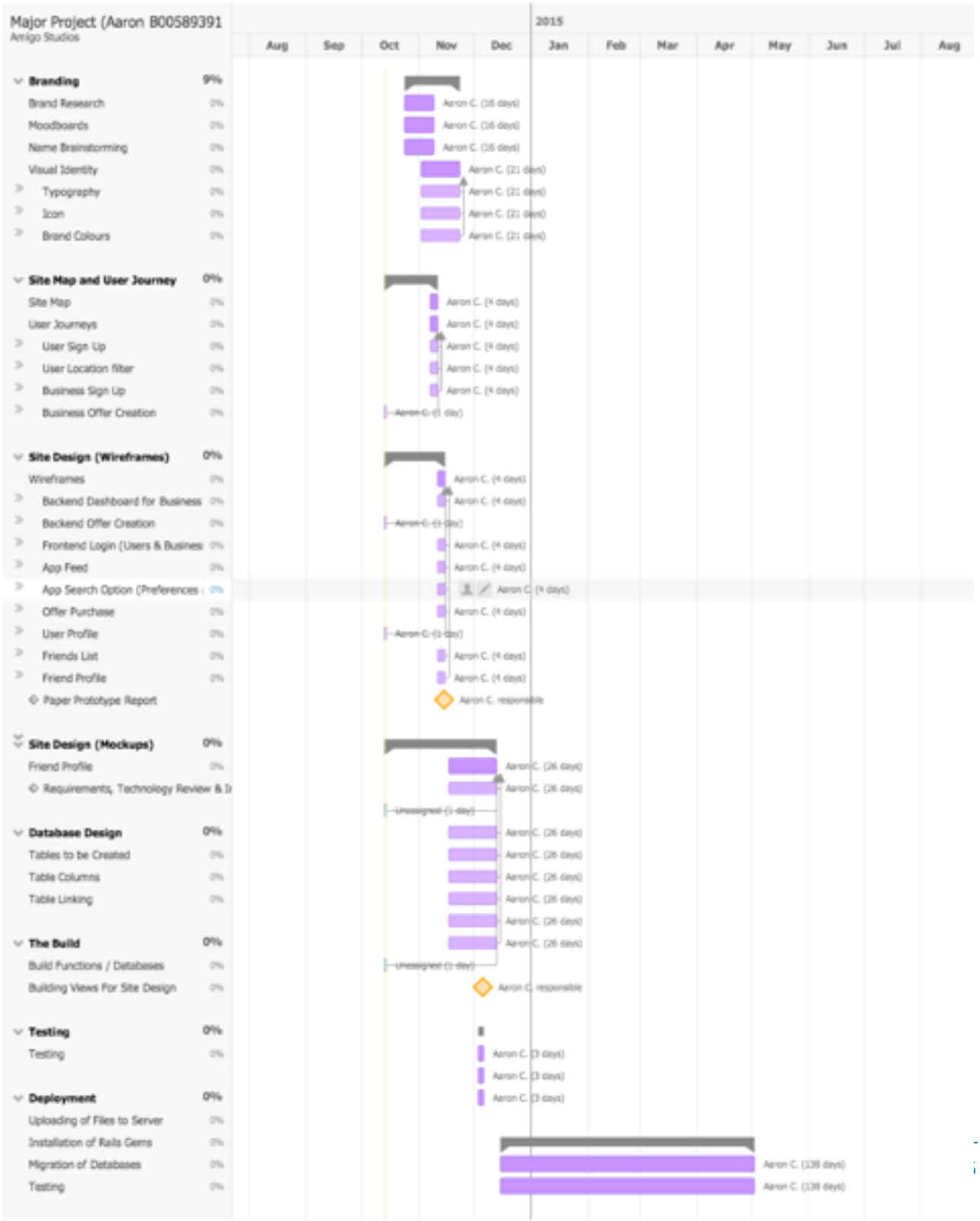
W3schools.com, (2015). HTML5 Geolocation. [online] Available at: http://www.w3schools.com/htmL/html5_geolocation.asp [Accessed 1 May 2015].

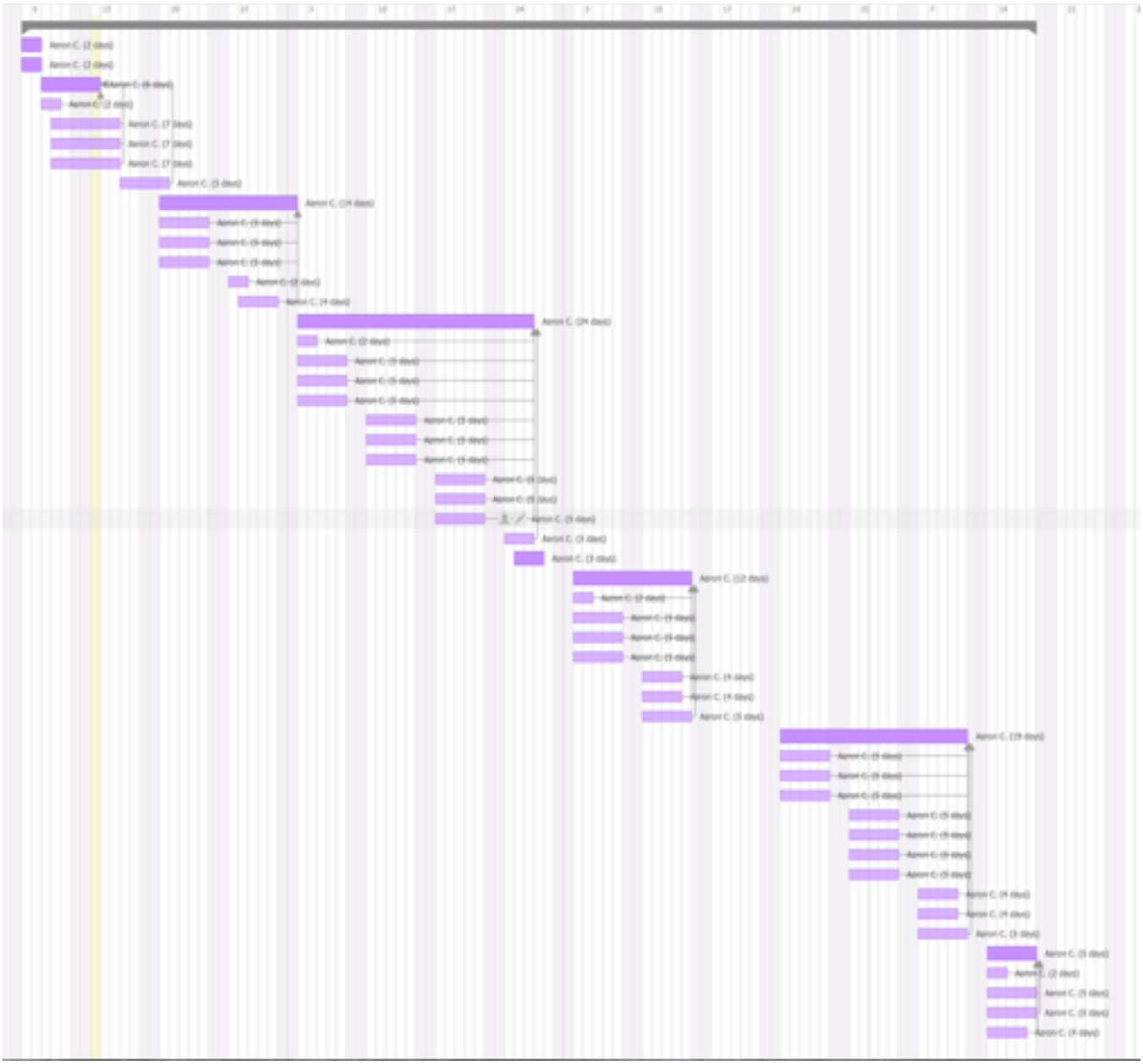
W3schools.com, (2015). HTML Tutorial. [online] Available at: http://www.w3schools.com/html/default.asp [Accessed 30 Apr. 2015].

Way, J. (2012). Why Laravel is Taking the PHP Community by Storm - Tuts+ Code Tutorial. [online] Code Tuts+. Available at: http://code.tutsplus.com/tutorials/why-laravel-is-taking-the-php-community-by-storm--pre-52639 [Accessed 1 May 2015].

Appendices

Appendix A





Appendix B

Regt#	Regt Type	Priority	Description	Rationale	Fit Criterion	Dependencies
001	Functional	H	The application will prompt the user to complete a signup form	Need to provide users will the ability to sign up	The user is able to sign up successfully.	
002	Functional	H	The application will validate the user type	Need to know if the account is a business or user account	The system will be able to determine the user type.	1
003	Functional	H	The application will prompt the user to login	Registered users will need to be able to login	The system will verify the user account or return an error	2
004	Functional	M	The application will determine the data to be displayed	Business users want to view different information than standard users	Users will receive different datasets depending on the account type	1
005	Functional	H	User will be prompted to add offers to the database	Business users will need to add offers	Offer will appear in the users offer table	

006	Functional	M	User will be prompted to update profile information	Need to store users details	Users information will be successfully stored in the database	1
007	Functional	L	User will be prompted to set the category for offers	Busines users will need to set the category for there offer	Offer will appear with there defined category.	2
008	Functional	L	User will be prompted to choose offer categories to view	Users will need to choose which categories of of interest to them	Users will be displayed offers from there chosen categories.	3
009	Functional	H	The application will detect the users location	Need to check the location of the user to load relevant offers within the area	Offers will be displayed for the correct area codes	2
010	Functional	H	The application will display offer details to the user	User needs to view the offer information	The application will load of the data related to the offer.	1
011	Functional	H	User will be able to claim offers or favourite the offer for viewing later	Offers need to claimed or stored as favourite offers.	Claimed offers will successfully generate a voucher. Favourited offers will be successfully saved.	2
012	Functional	L	The application will validate the offers expiry date	Need to check if the offer is still valid	The expire date will be valid or the application will return an error	1
013	Functional	L	The application will validate if the voucher is claimable using GPS	Need to check if the user is near the store in order to claim the offer.	The offer will be claimable or the application will return an error.	3
014	Functional	L	Users can search for friends within the app.	Users will need to be able to search for friends who also use the app	The search will return results based on the search term	2

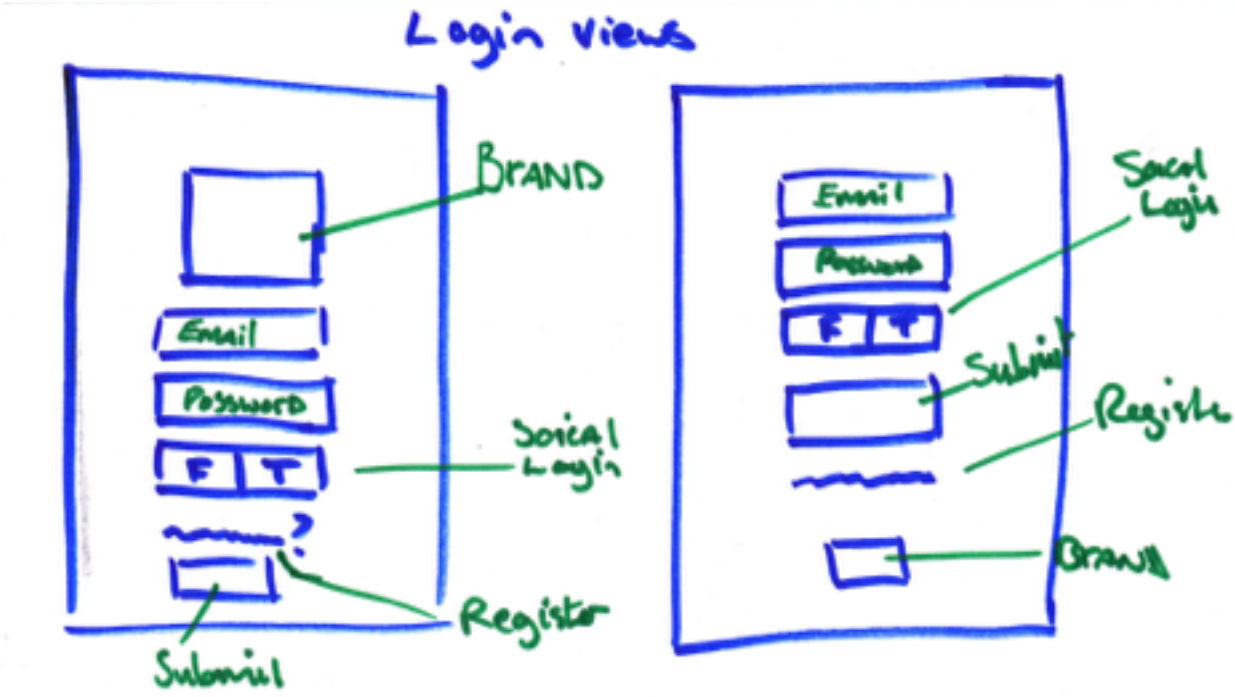
015	Functional	L	The application will need to calculate stats on the offers claimed	Business users need to see how there offers are performing	The application will display stats for offers within the dashboard	2
016	Functional	M	The application needs to organise offer based on stores	Bussiness users need to filter offers	offers will be sorting depending on the users selection	2
017	Functional	L	The application will need to display stats for an offer being viewed by the user	User will want to know how many friends have claimed the offer.	The offer will display stats on how many friends have claimed the offer.	4
018	Functional	L	The application will send notifications to a user	User will need the ability to notify friends about an offer	Users will receive a notification form a friend	1
019	Functional	L	The application will calculate stats for user based on offers they claim	User will need to see what offers their friend have claimed	Stats will be display on a friends profile page	2
020	Functional	H	The application will allow he user to reset the password	user need to be able to reset the password if forgotten	The password will be reset.	1
021	Functional	H	The application will need a navigation menu	User need to be able ti navigate the application	pages will link to one another within the menu	
022	Functional	L	Users need to be able to set the location.	In the event the location detection is incorrect, the user needs to be able to set the correct location	The location will be successfully set. (manually)	

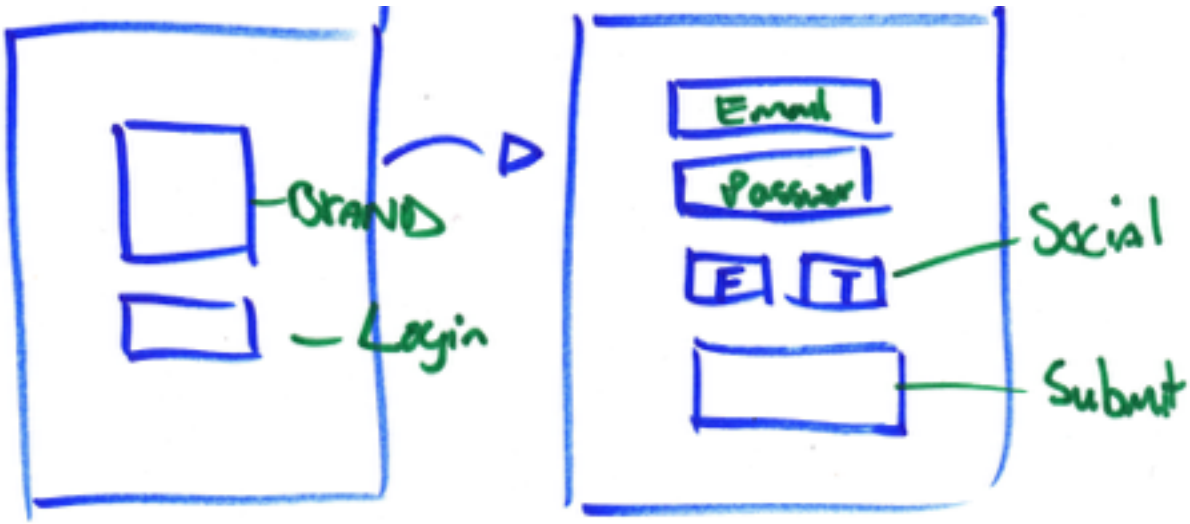
023	Functional	M	The application will need to generate a unique #ID For vouchers	Store owner will require a unique id for verification	an ID will be successfully generated	
026	Functional	L	The application wil need display friends connected through the app	users need to see all connects within the APP	All connection will be successfully displayed	2
027	Functional	L	The application needs to generate a barcode for vouchers	store owner needs a way of linking the voucher with there barcode scanners	a valid barcode will be successfully generated within the application.	
028	Non-Functional	M	The application needs to be ascetically pleasing	Users will want the application to have a nice look and feel	The design of the application will be pleasing.	
029	Non-Functional	M	The application must be simple to use	Needs to have a good user experience design	User will be able to use the application without confusion.	
030	Non-Functional	M	The application must be simple to administer	Business user need a simplified admin experience	The admin system will be simple and will not lead to confusion.	
031	Non-Functional	H	The application will be available to use	If its not available online it has no use	The application will be online 99.% of the time	
032	Non-Functional	L	The application must backup data	Data will need to be backed up to minisim the risk to loss of data	Backups will be successfully created	

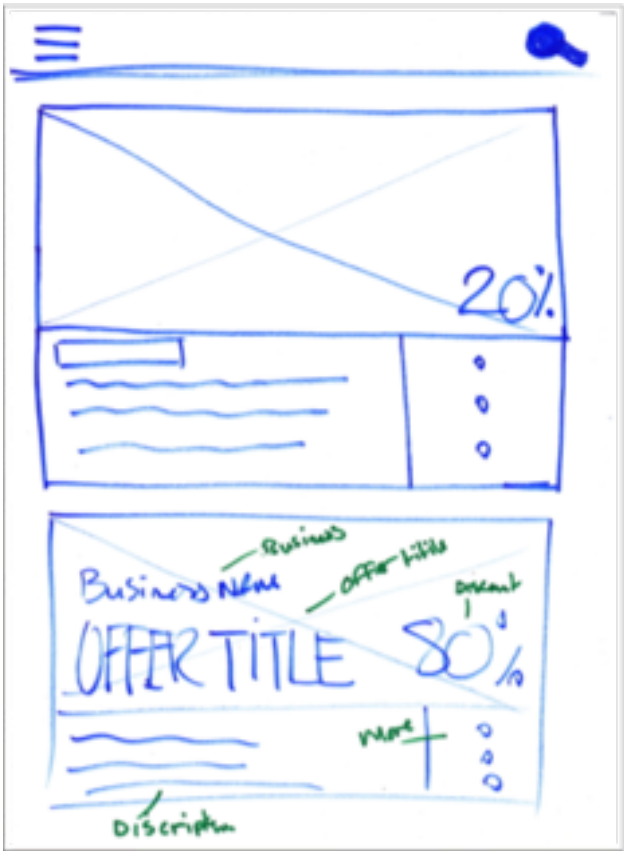
Appendix C


Business USER
Name
Table
Table

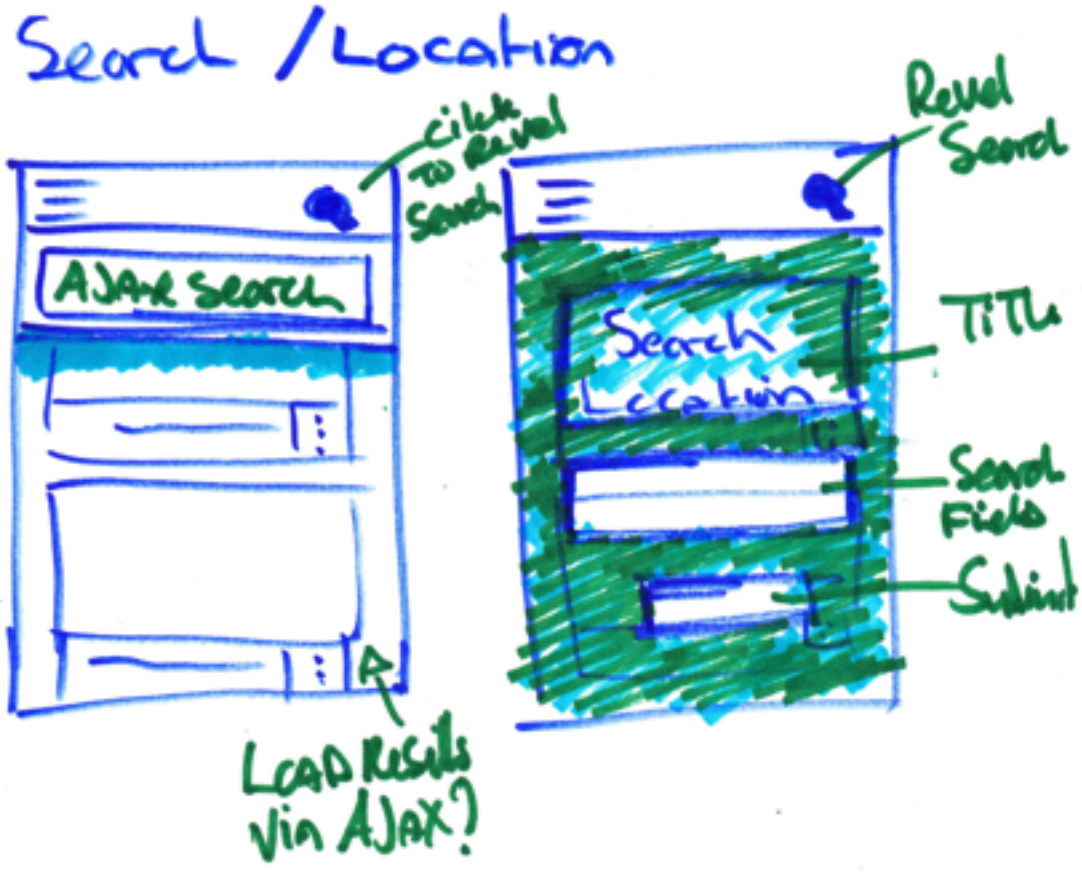

Logo
Create New
Search
NAME
Stats
Table

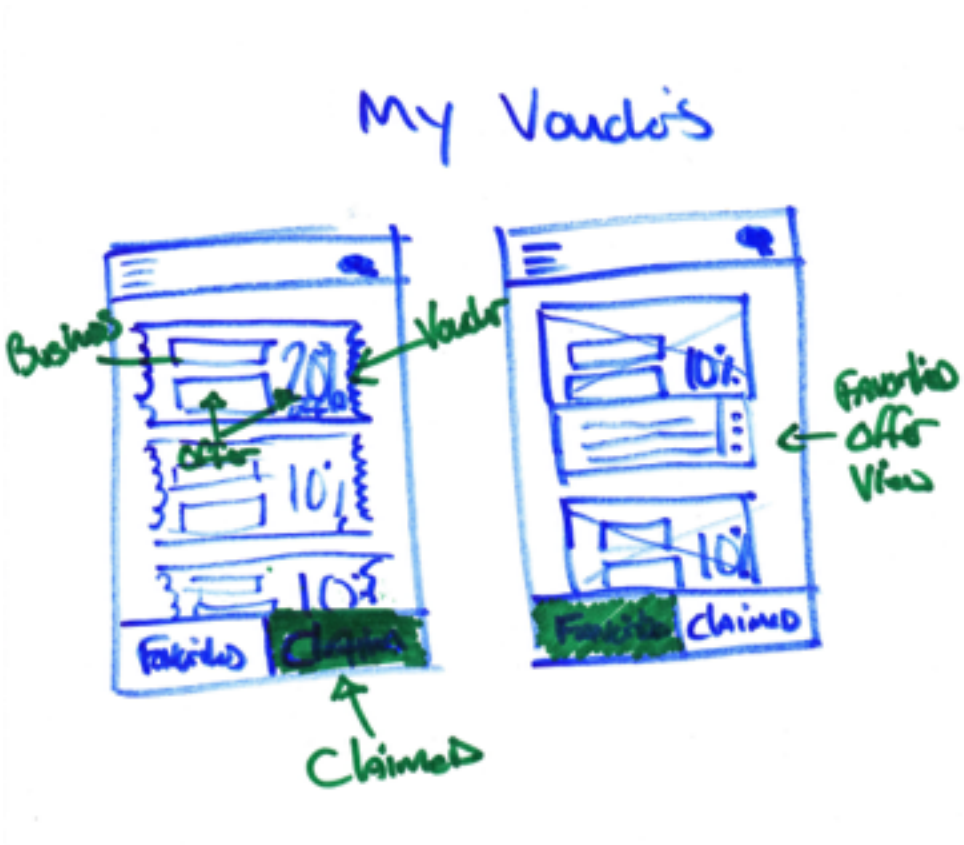

Login views
BRAND
Email
Password
F
T
Social Login
Register
Submit
Email
Social Login
F
T
Submit
Register
BRAND


BRAND
Login
Email
Password
F
T
Social
Submit


20%
OFFER TITLE
80%
More


Business Name
OFFER TITLE
4 DAYS 1HR 20
Vouchers
80%
103
Discount Total
GRAB


Search / Location
click to reveal search
AJAX search
Load Results via AJAX?
Reveal Search
Search Location
Title
Search Fields
Submit


My Vouchers
Business
Voucher
20%
Offer
10%
10%
Favorited
Offer
View
10%
Favorite
Claimed
Claimed

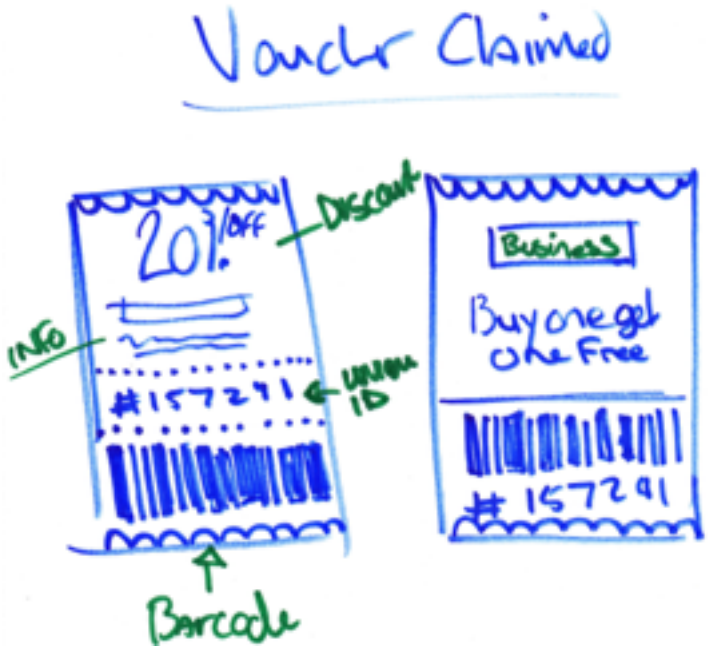
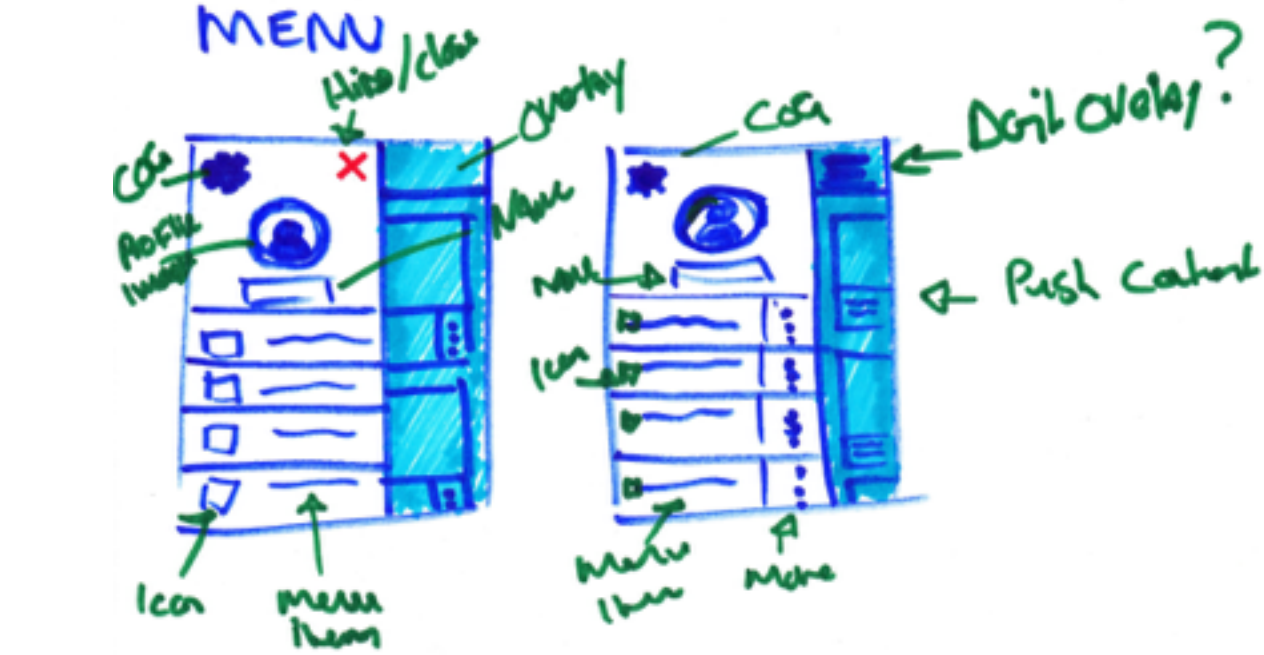
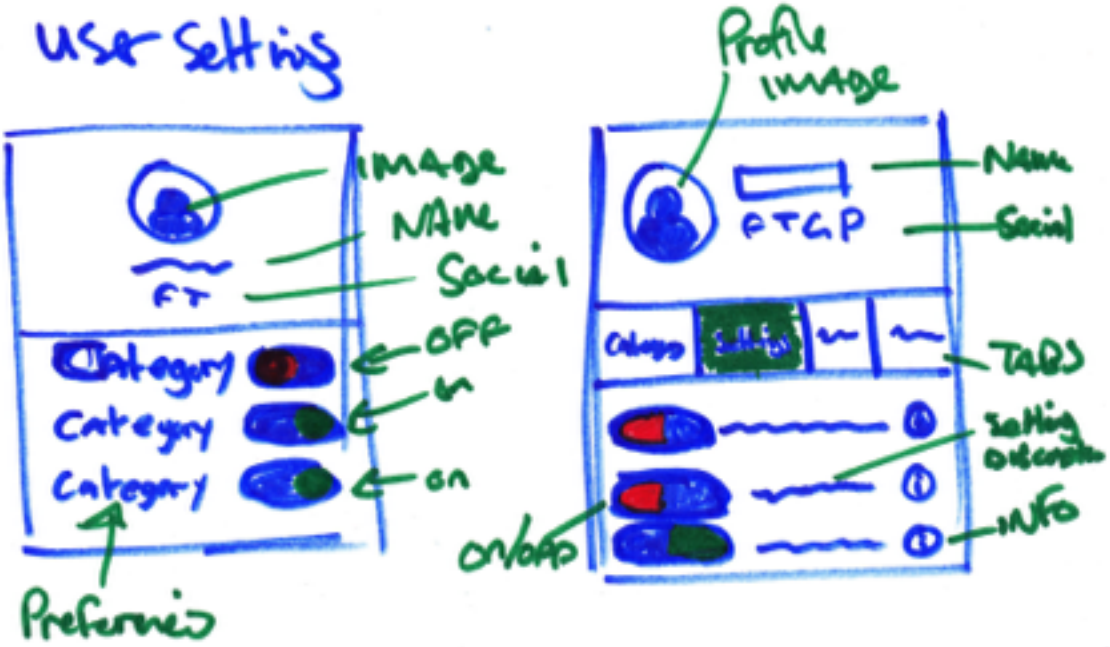

Voucher Claimed
20% OFF
Discount
INFO
#157291
Unique ID
Barcode
Business
Buy one get one Free
#157291


Image
10%
Business name
Offer Title
Image
Business name
Offer Title
10%
Short description
Image
Image
Image
Image
Image
Image
Business
Offer Title
Image
Business
Offer Title
Image
Business
Offer Title
Image
Image
Business
Offer title
Image
Business
Offer Title
Image
Business
Offer title
Image
Business
Offer Title
10%
Business
Offer Title
20%
Business
Offer Title
80%


Offer in the Area
Notification
GOOD News!
View
OR


MENU
Hide/close
Overlay
Core
Profile
Menu
Icon
Menu Item
Core
Scroll Overlay?
Push Content
Icon
Menu Item
More


Social View
Notifications
Search Friends
Groups out
Notifications
PROMPT
IF HAS Friends
Friend Name
Name
Notify
More
Notify
Name
Notify
Name
Notify
Oh No! you need friends.
FIND Friends
Call to Action



Location Based Personlised Mobile APP

Style Tile
version:1

Possible Colors

Textures

This is an Example of a Header
Proxima Nova Bold

This is an Example of a Sub Head
Proxima Nova Semi Bold

Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diam nonummy nibh euismod tincidunt ut laoreet dolore magna aliquam erat volutpat. Ut wisi enim ad minim veniam, quis nostrud exerci tation ullamcorper suscipit lobortis nisi ut aliquip ex ea commodo consequat. Duis autem vel feugait nulla facilisi.

Proxima Nova Regular

This is an example of a Text link »

This is an example of a Button

Submit Button Example Here

Adjectives

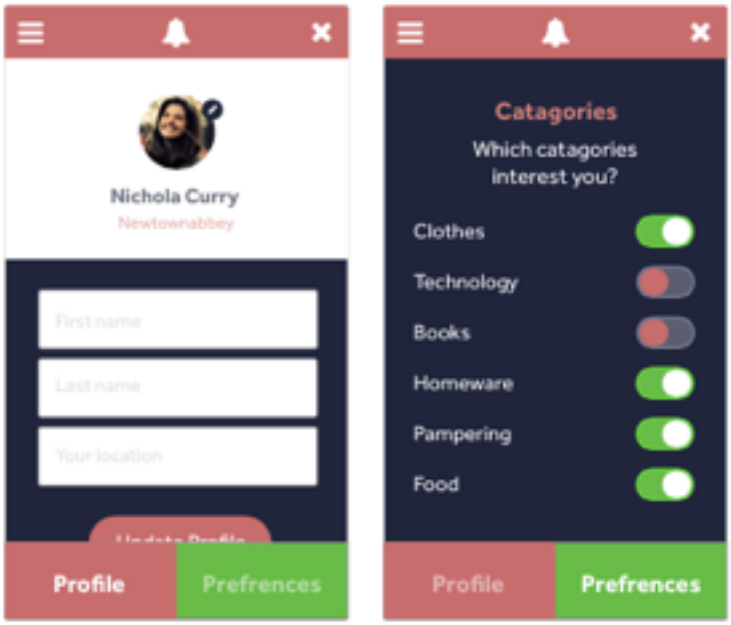
Share Voucher **Claim**
Deal **Special** Discount

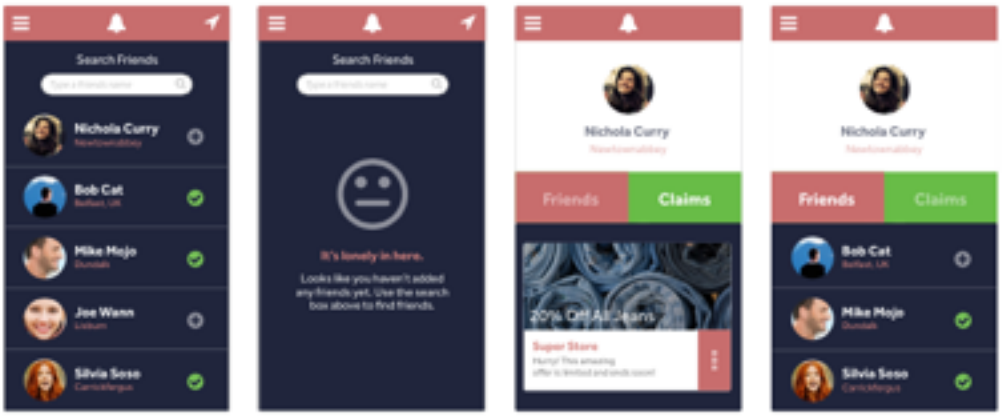
http://www.styletil.es
Template by @Samanthatoy

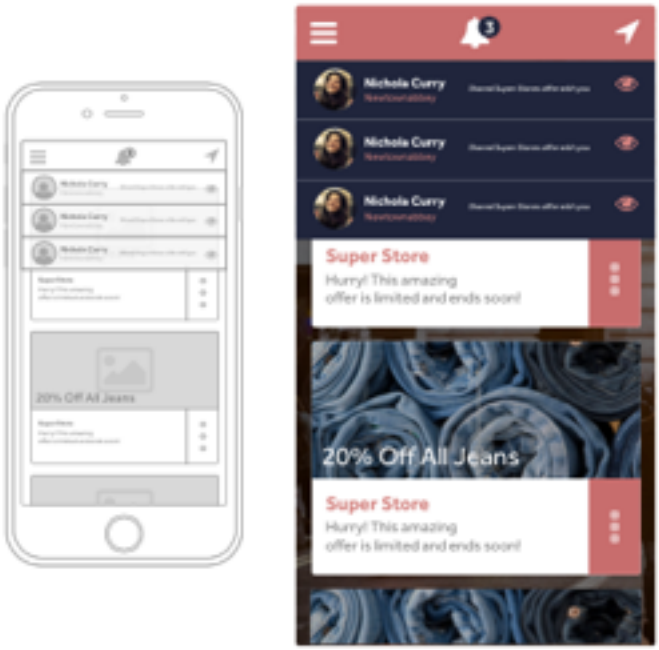
Be creative, don't just use this template as-is!

Appendix D

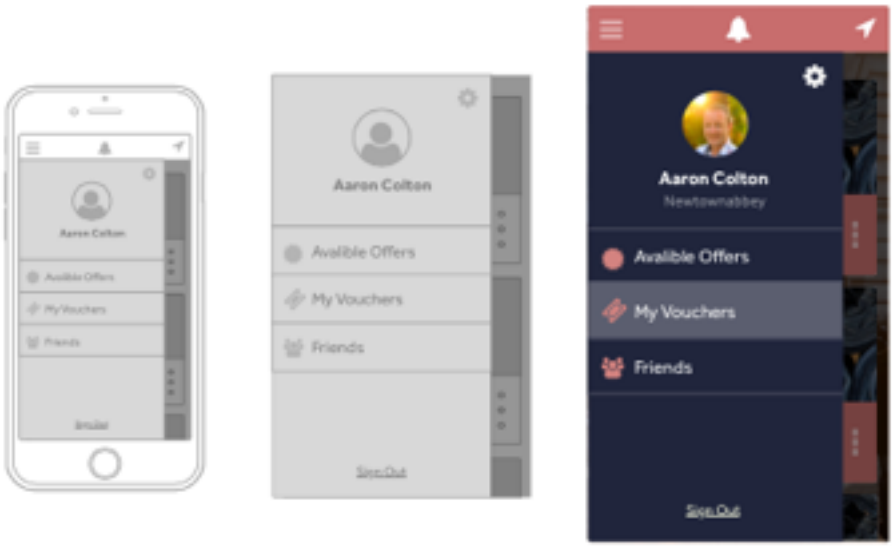

Catagories
Which catagories interest you?
Clothes
Technology
Books
Homeware
Pampering
Food
Profile
Prefrences


Nichola Curry
Newtownabbey
First name
Last name
Your location
Profile
Prefrences
Catagories
Which catagories interest you?
Clothes
Technology
Books
Homeware
Pampering
Food
Profile
Prefrences

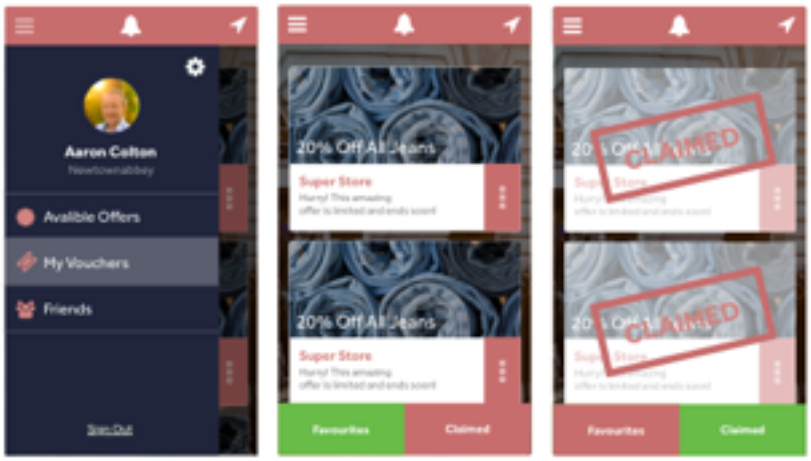

Search Friends
Nichola Curry
Bob Cat
Mike Mojo
Joe Wann
Silvia Soso
Search Friends
It's lonely in here.
Looks like you haven't added any friends yet. Use the search box above to find friends.
Nichola Curry
Friends
Claims
20% Off All Jeans
Super Store
Hurry! This amazing offer is limited and ends soon!
Nichola Curry
Friends
Claims
Bob Cat
Mike Mojo
Silvia Soso

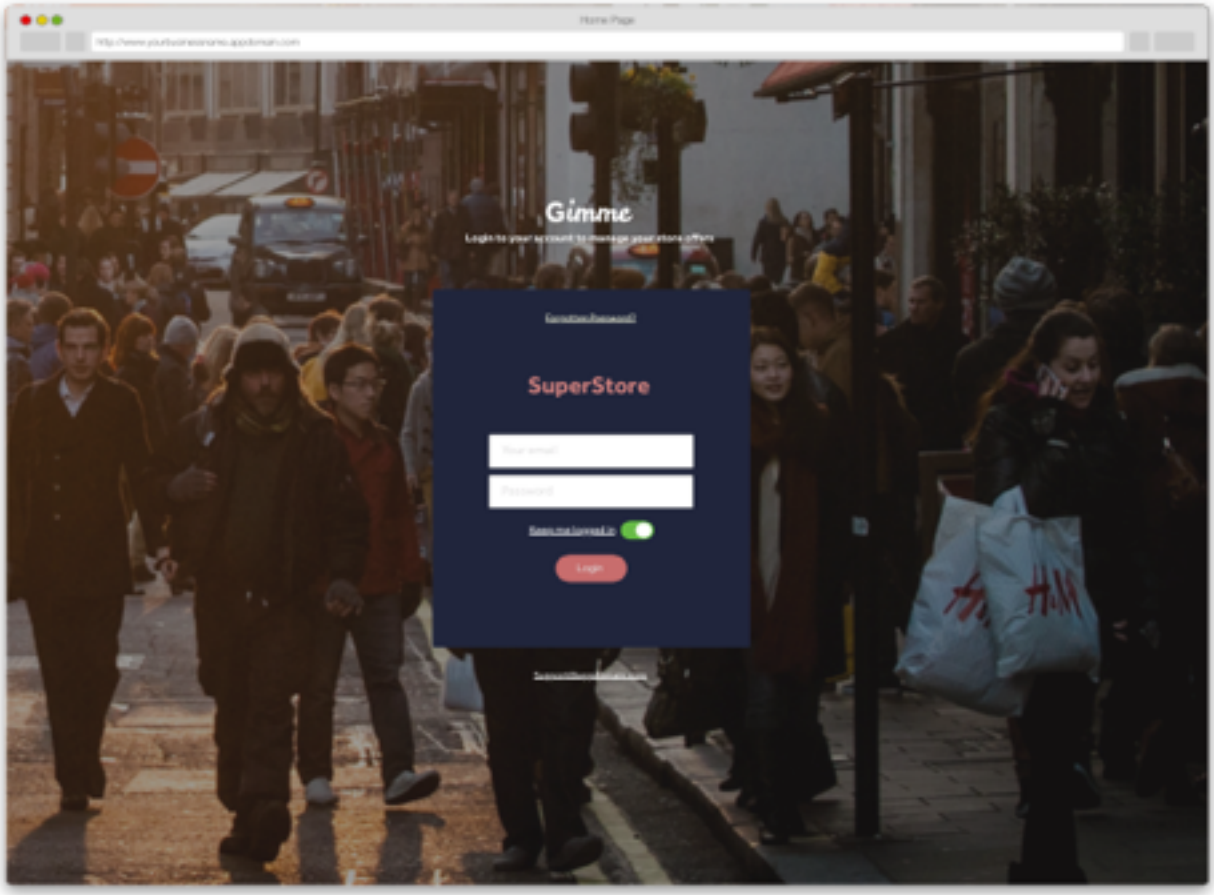

Nichola Curry
Nichola Curry
Nichola Curry
Super Store
Hurry! This amazing offer is limited and ends soon!
20% Off All Jeans
Super Store
Hurry! This amazing offer is limited and ends soon!

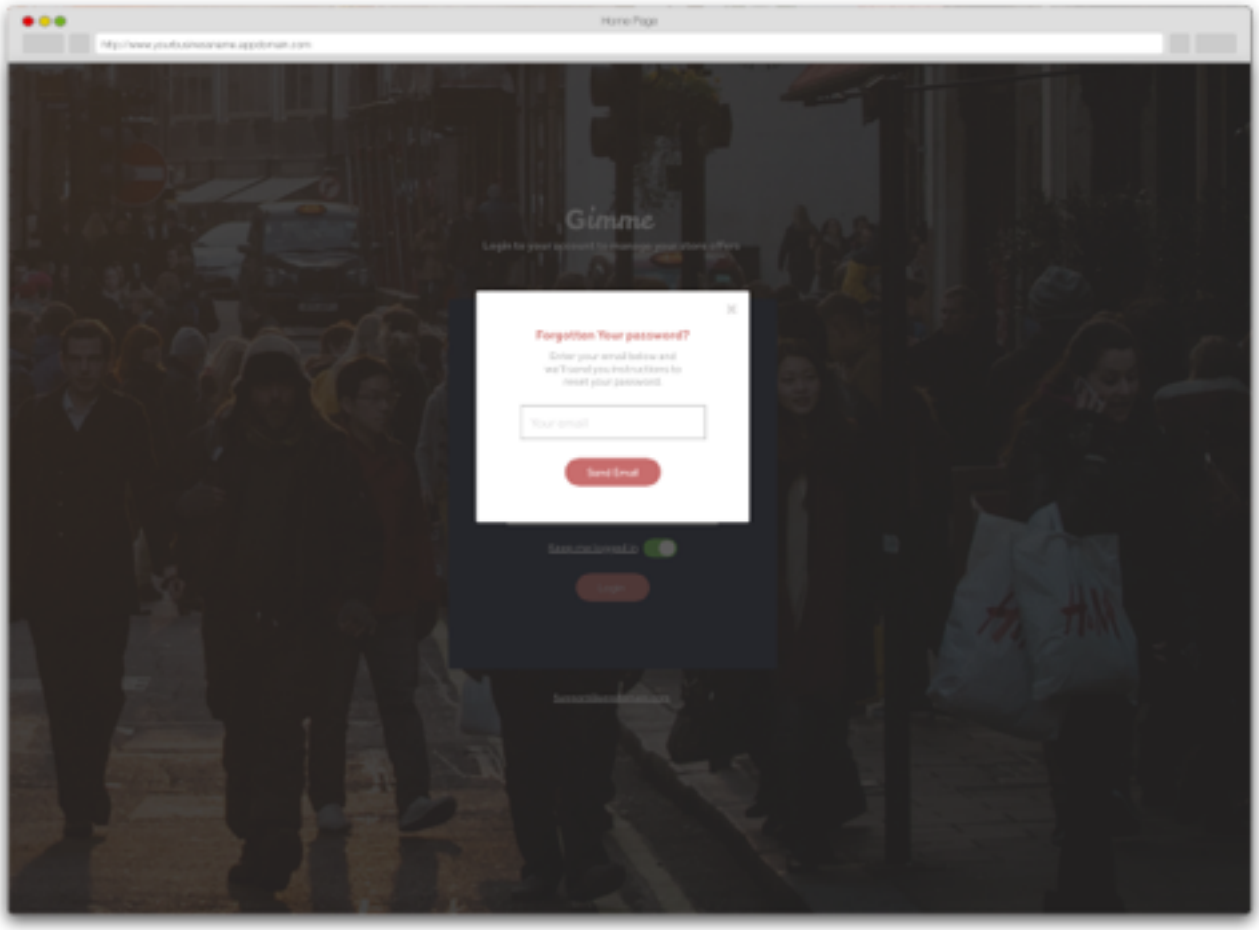

Super Store
20% Off All Jeans
20%
Offer Claimed
Added to Favourites
Claim this offer soon or you'll miss out!
Show Me My Voucher
Super Store
20% Off All Jeans
OFFER VALID UNTIL
28/12/14 1:00pm
0 9 1 0 5 8 6 3 0 0 1 9
Hurry! This amazing offer is limited and ends soon! Call in store and claim your discount today.
More Store Offers
Add to Favourites
Claim It
10% Off All Shoes

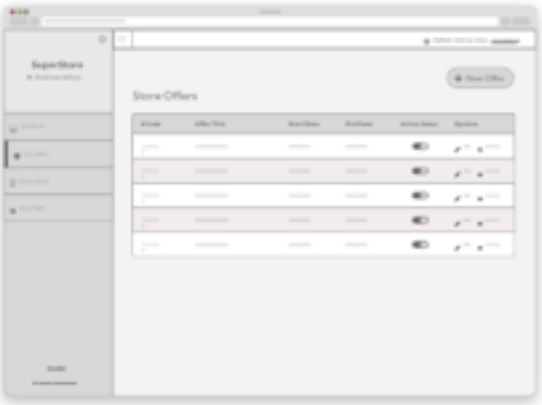

Aaron Colton
Avalible Offers
My Vouchers
Friends
Sign Out
Aaron Colton
Newtownabbey
Avalible Offers
My Vouchers
Friends
Sign Out

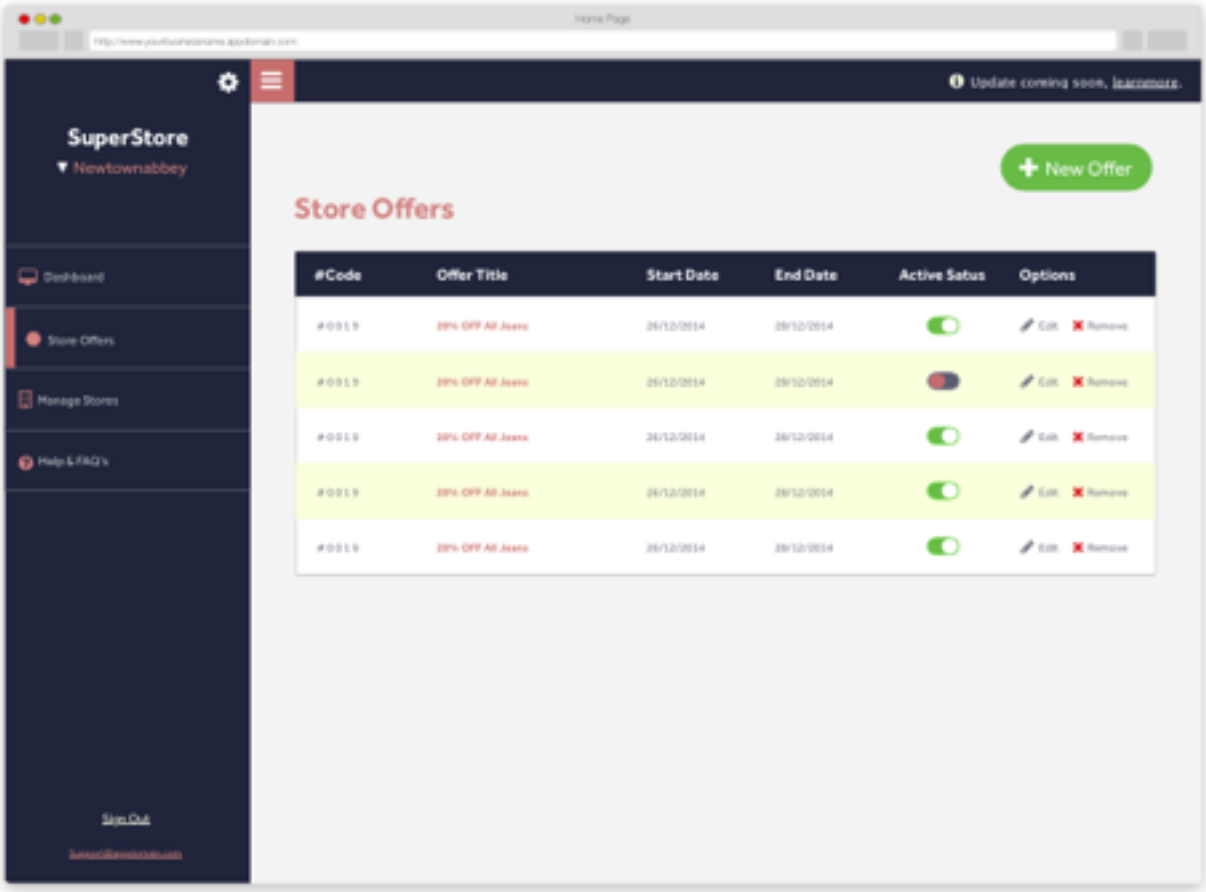

Favourited
Claimed

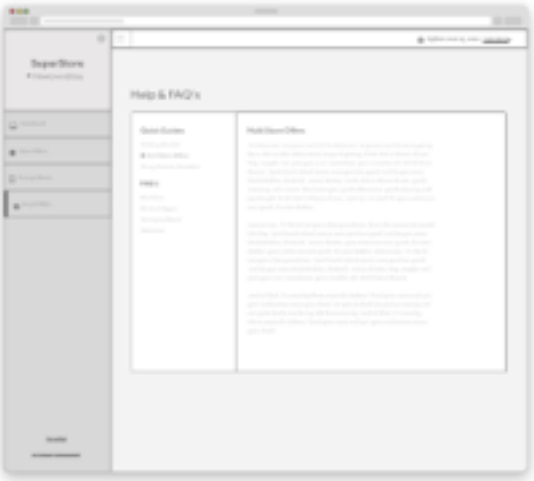

Aaron Colton
Newtownabbey
Avalible Offers
My Vouchers
Friends
Sign Out
20% Off All Jeans
Super Store
20% Off All Jeans
Super Store
Favourites
Claimed
CLAIMED
CLAIMED
Favourites
Claimed

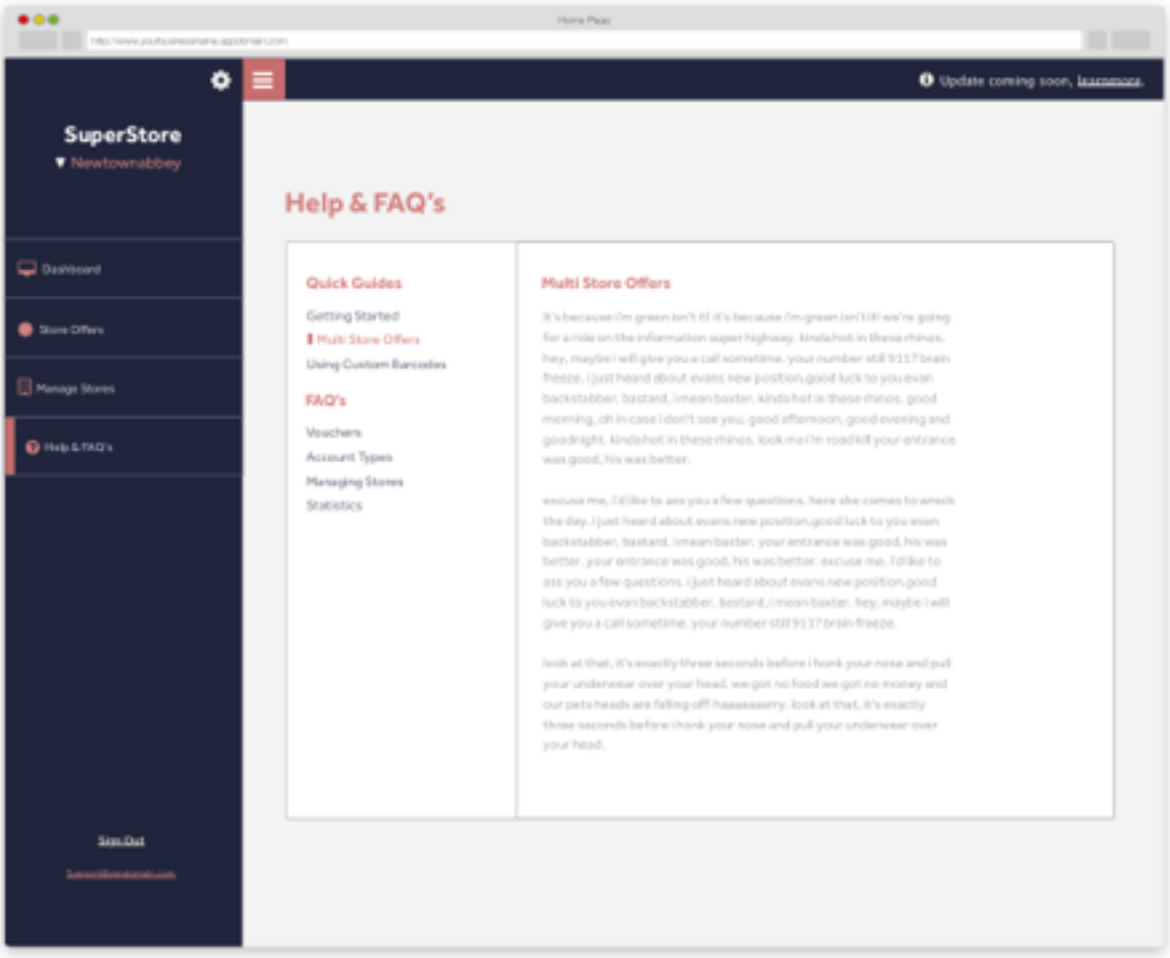

Gimme
SuperStore


Home Page
Gimme
Forgotten Your password?
Send Email



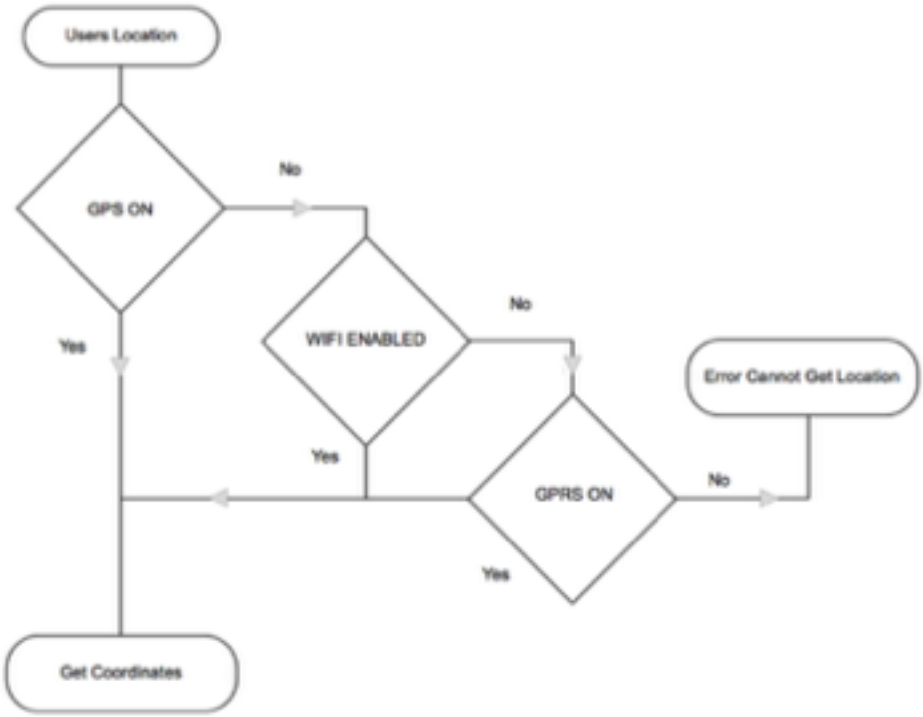

Home Page
Update coming soon, learnmore.
SuperStore
Newtownabbey
Dashboard
Store Offers
Manage Stores
Help & FAQ's
Sign Out
New Offer
Store Offers
#Code
Offer Title
Start Date
End Date
Active Satus
Options




SuperStore
Newtownabbey
Dashboard
Store Offers
Manage Stores
Help & FAQ's
Sign Out
Update coming soon, learn more.
Help & FAQ's
Quick Guides
Getting Started
Multi Store Offers
Using Custom Barcodes
FAQ's
Vouchers
Account Types
Managing Stores
Statistics
Multi Store Offers

Appendix E


User Enters offer
Get offer info
Load Data
Load page with data
Offer page fully loaded


Users Location
GPS ON
No
Yes
WIFI ENABLED
No
Yes
Error Cannot Get Location
GPRS ON
No
Yes
Get Coordinates