



National College of Ireland  
BSc in Computing  
2015/2016

Darragh Blake  
x12467038  
x12467038@student.ncirl.ie

OfferApp



Technical Report

## Declaration Cover Sheet for Project Submission

### SECTION 1 *Student to complete*

<b>Name:</b> <b>Darragh Blake</b>
<b>Student ID:</b> <b>X12467038</b>
<b>Supervisor:</b> <b>Dr. Dominic Carr</b>

### SECTION 2 Confirmation of Authorship

*The acceptance of your work is subject to your signature on the following declaration:*

I confirm that I have read the College statement on plagiarism (summarised overleaf and printed in full in the Student Handbook) and that the work I have submitted for assessment is entirely my own work.

Signature: \_\_\_\_\_

Date: \_\_\_\_ 11/05/16 \_\_\_\_\_

NB. If it is suspected that your assignment contains the work of others falsely represented as your own, it will be referred to the College's Disciplinary Committee. Should the Committee be satisfied that plagiarism has occurred this is likely to lead to your failing the module and possibly to your being suspended or expelled from college.

**Complete the sections above and attach it to the front of one of the copies of your assignment,**

## **What constitutes plagiarism or cheating?**

The following is extracted from the college's formal statement on plagiarism as quoted in the Student Handbooks. References to "assignments" should be taken to include any piece of work submitted for assessment.

Paraphrasing refers to taking the ideas, words or work of another, putting it into your own words and crediting the source. This is acceptable academic practice provided you ensure that credit is given to the author. Plagiarism refers to copying the ideas and work of another and misrepresenting it as your own. This is completely unacceptable and is prohibited in all academic institutions. It is a serious offence and may result in a fail grade and/or disciplinary action. All sources that you use in your writing must be acknowledged and included in the reference or bibliography section. If a particular piece of writing proves difficult to paraphrase, or you want to include it in its original form, it must be enclosed in quotation marks

and credit given to the author.

When referring to the work of another author within the text of your project you must give the author's surname and the date the work was published. Full details for each source must then be given in the bibliography at the end of the project

## **Penalties for Plagiarism**

If it is suspected that your assignment contains the work of others falsely represented as your own, it will be referred to the college's Disciplinary Committee. Where the Disciplinary Committee makes a finding that there has been plagiarism, the Disciplinary Committee may recommend

- that a student's marks shall be reduced

- that the student be deemed not to have passed the assignment
- that other forms of assessment undertaken in that academic year by the same student be declared void
- that other examinations sat by the same student at the same sitting be declared void

Further penalties are also possible including

- suspending a student college for a specified time,
- expelling a student from college,
- prohibiting a student from sitting any examination or assessment.,
- the imposition of a fine and
- the requirement that a student to attend additional or other lectures or courses or undertake additional academic work.

# Table of Contents

Executive Summary .....	8
1 Introduction.....	<b>Error! Bookmark not defined.</b>
1.1 Background .....	<b>Error! Bookmark not defined.</b>
1.2 Aims .....	<b>Error! Bookmark not defined.</b>
1.3 Technologies .....	8
2 System .....	10
2.1 Requirements .....	10
2.1.1 Functional requirements .....	10
2.1.2 Use Case Diagram.....	10
2.1.3 Requirement 1: Login Use Case Diagram .....	11
Description & Priority .....	11
2.1.4 Requirement 2 Geolocation .....	13
2.1.5 Requirement 3: Adding Offers Use Case Diagram .....	16
2.1.6 Requirement 4: Map Case Diagram.....	18
2.1.7 Requirement 5: Chat System Case Diagram .....	20
2.1.8 Data requirements .....	22
2.1.9 User requirements .....	23
2.1.10 Environmental requirements .....	23
2.2 Implementation .....	24
2.3 Design and Architecture .....	25
2.4 Hardware Architecture.....	26
2.5 Software Architecture .....	26
2.6 Testing.....	34
2.7 Graphical User Interface (GUI) Layout .....	39
3 Conclusions.....	40
4 Further development or research .....	41
5 References .....	42
5.1 References .....	42
6 Appendix .....	43

6.1	Project Proposal .....	43
	Objectives.....	43
7	Background .....	44
8	Technical Approach.....	45
9	Special resources required.....	45
10	Technical Details .....	47
11	Evaluation.....	47

## **Executive Summary**

OfferApp is a Geolocation based Web Application designed for mobile devices. It is intended to allow business to easier promote their offers which can be picked up automatically by near-by devices. OfferApp aims to show users the offers which are within the users vicinity or search parameters so they can avail of them at their leisure. OfferApp saves users money and time as well as being an effective marketing and promotional tool for Businesses. The application is built in Web 2.0 Languages such as PHP, JavaScript, CSS, HTML5, XML and JSON. OfferApp uses MySQL which is a Relational Database Management System. The application uses many networking practices such as Web services & API's.

### **1.1 s**

I am to create a mutual ground where all companies can add in their offers and deals. The user will be able to easily search for the area they are in and also using device Geolocation.

### **1.2 Technologies**

I used modern web languages such as HTML5, PHP5, JavaScript, XML, CSS3 and JQuery(W3C). I used the modern CSS framework Bootstrap for the clean User Interface. For the main functionality I used PHP which is my main Object Orientated Language. I choose PHP as is great for back ends as it works very well with databases. I used MySQL as my relational database for OfferApp. To manage my MySQL database I used PHPMyAdmin(PHP)MyAdmin). I used JavaScript for other functionality and in particular for the Google Maps API. XML is used to output a response from my own API which I wrote in PHP which passes the information from the database to the JavaScript Google Maps API. I used D3.JS to display



dynamic data. For the mobile aspect I used Android Webview to build an apk written in Java for OfferApp.

## 2 System

### 2.1 Requirements

OfferApp is a Web Application which means it will run on any device connected to the internet which has a browser. Some functionality such as the Geolocation is written in HTML5 which means older devices will be limited to usage. OfferApp also uses the device GPS to find location. The Application has an Android Web view version which can be ran only on Android Devices and downloaded from the Play store as an APK which makes it easier for Android users to use OfferApp. An internet connection is required to use the Application as it uses Heavy networking functionality such as The Google Maps API, Web hosted Database and OfferApp's API.

#### 2.1.1 Functional requirements

Offer App has many functions. Here I have devised a list of the functional requirements in order of importance.

- **Login System** - The login is used to allow users to sign up and login to the system.
- **Geolocation** - This requirement is an essential requirement as it is how the user will be able to view what deals are around them.
- **Adding Items** - The business users will be presented with a different dashboard which will allow them to create and add different offers and deals.
- **Map** - Used to implement the users location and view the deals.
- **Chat feature** - Used to direct message the business owners for information regarding offers.

#### 2.1.2 Use Case Diagram

Below is a diagram of my full use case.

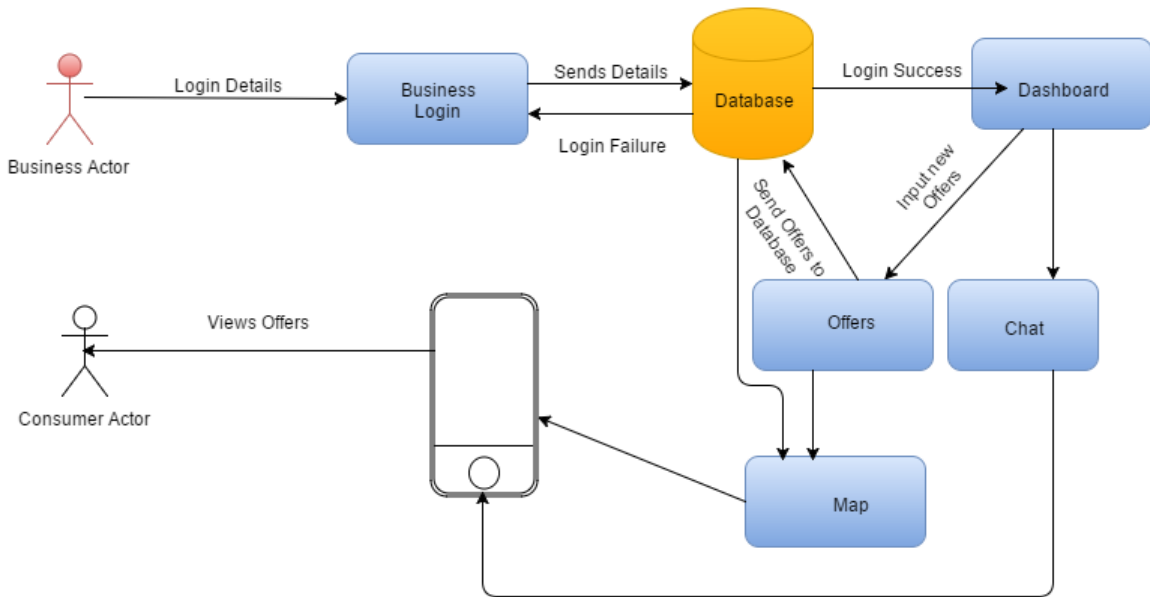


Fig 1.0

### 2.1.3 Requirement 1: Login Use Case Diagram

#### Description & Priority

The Login allows users to access the application. This is essential as business users need to be able to login to add details of offers and communicate with the customers. This is also essential because the application is populated by user generated content, which makes the number of users very important. This is my second most important requirement.

#### 2.1.3.1 Use Case

##### Scope

The scope of this use case is to log a user into the system.

##### Description

This use case describes the logging in of a user into the system. It shows the steps the system will take to validate the users details.

##### Use Case Diagram

The actor is either a business user or a consumer.

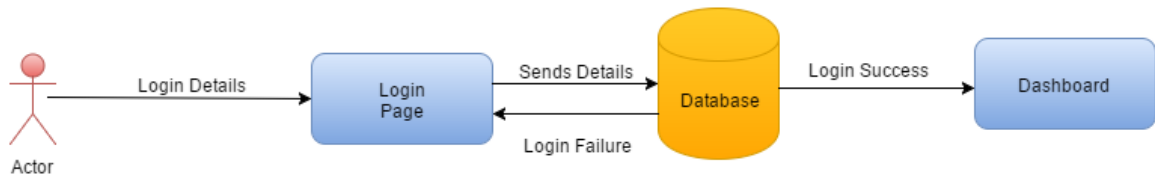


Fig 1.1

## Flow Description

### Precondition

The system is in initialisation mode when a user goes to the login page.

### Activation

This use case starts when an the actor enters his/her information into the login system.

### Main flow

1. The system allows the user to input there login credentials to the login page. Then the data is send to an external SQL database which validates the user. Once the data has been validated with the correct username and password as that found in the database it allows the user login, and brings them to the dashboard page.

### Alternate flow

A1 : User is not signed up

1. The system will alert the user that they have not signed up.
2. The user will click the sign up button and create an account.
3. Once they have created an account the system will then log the user in.

### Exceptional flow

A1 : Wrong Username or Password.

1. The system will alert the user of an incorrect username or password.
2. The user will correct the login information.

3. The system will then log the user in.

### **Termination**

This is when the user successfully logs into the system and a session is started.

### **Post condition**

The system goes into a wait state when the user is not logging out.

## **2.1.4 Requirement 2 Geolocation**

### *2.1.4.1 Description & Priority*

The Geolocation is essential to the application. It finds the users location and therefore can give you offers and deals in your vicinity.

### *2.1.4.2 Use Case*

#### **Scope**

The scope of this use case is to find the users realtime location.

#### **Description**

This use case describes the process of gathering the users location from Googles API's and assisted GPS.

### Use Case Diagram

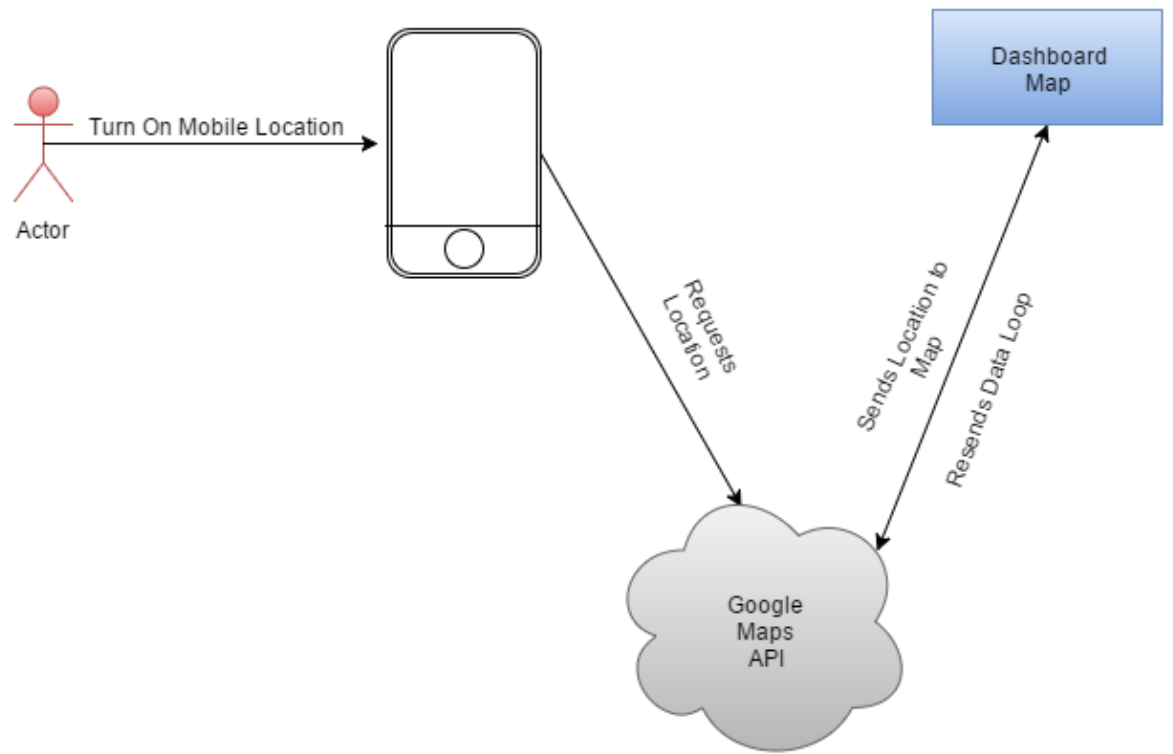


Fig 1.2

### Flow Description

### Precondition

The system is in initialisation mode when the users Login to the system and are on the dashboard map page.

### **Activation**

This use case starts when the users turn on their mobile devices location.

### **Main flow**

1. The system detects the users location via the mobile devices location.
2. The actor turns on this feature.
3. The system finds the location and presents it on a map.
4. The actor can view the map for offers near their location.

### **A1 : Location Turned Off**

1. The system will throw an error is the users mobile device has the location turned off.
  2. The user will enable device location.
  3. The system will pull the users correct device location.

### **Exceptional flow**

#### **E1 : Device does not support HTML5**

1. The system will alert the user that HTML5 is not supported.
2. The user will have to switch to a new device or upgrade to a newer version of Android.
3. The use case terminates.

### **Termination**

The system terminates when the user turns off the mobile location.

### **Post condition**

The system goes into a wait state after a user has not moved location for a number of seconds.

## 2.1.5 Requirement 3: Adding Offers Use Case Diagram

### 2.1.5.1 Description & Priority

This is the key feature to my application. It is where the database is populated by the user generated content. The business user adds data into the offer entry page. This data is then sent to a database where it is saved. The data is then pulled from the database into a Google map. It is displayed in balloons in the map.

### 2.1.5.2 Use Case

#### Scope

The scope of this use case is to show how the application populates the map with offers added by business users.

#### Description

This use case describes how a business user can login and add different offers to the offer page. Then it shows how the offers are displayed on the map.

#### Use Case Diagram

The actor is a business.

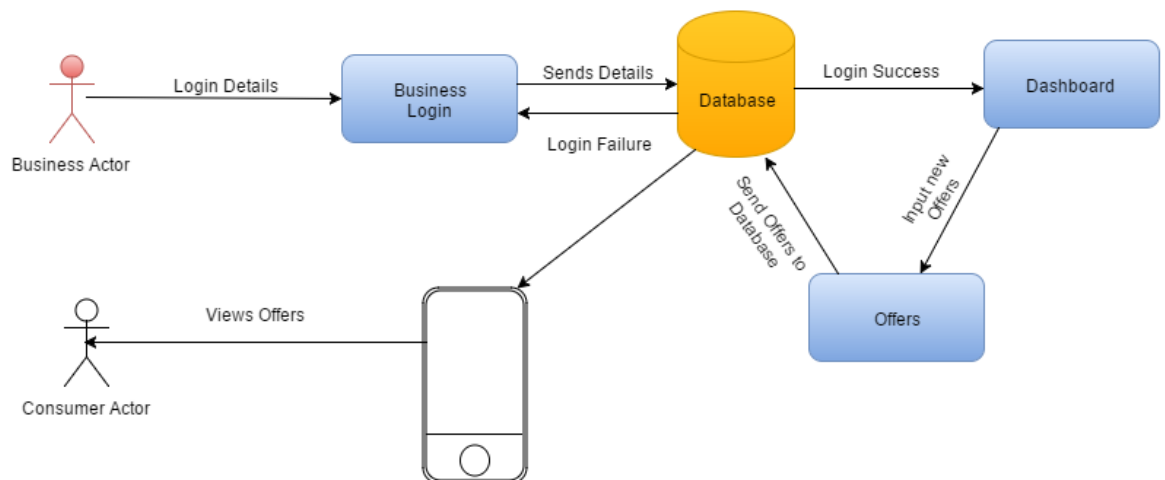


Fig 1.3



## **Flow Description**

### **Precondition**

The system is in initialisation mode when business user goes into the offers page.

### **Activation**

This use case starts when an the business user enters his/her information into the offers page.

### **Main flow**

1. The system allows the business users to login to the application. Then they can go to the offers page and add an offer. The system pulls the business location in the form of latitude and longitude on the map. It then allows the business users to enter their offers which will appear under the business name on the map. They are displayed as balloons on the map.

### **Alternate flow**

A1 : Too Many Offers

1. The system will only allow three offers at a time.
2. The will delete one offer from the database.
3. The use case allows the user to enter a new offer once one has been deleted.

### **Exceptional flow**

E1 : Not a business owner.

1. The system alert the user that they are not a business owner and therefore will not allow them to create offers.
2. The actor will sign out of the system and sign back in as a business owner if they have one.
3. The use case continues at position 4 of the main flow once the user successfully logs in as a business owner.

### **Termination**

This is when the user enters the data successfully.

### **Post condition**

The system goes into a wait state when there is no data being uploaded to the server.

## **2.1.6 Requirement 4: Map Case Diagram**

### ***2.1.6.1 Description & Priority***

The Map is a very important functional requirement. It manages all of the data sent to the map and it is also where the users real-time location is displayed.

### ***2.1.6.2 Use Case***

#### **Scope**

The scope of this use case is to explain the use of the Google maps API in Offer App

#### **Description**

This use case describes how the Maps works within the system.

#### **Use Case Diagram**

The actor is a consumer.

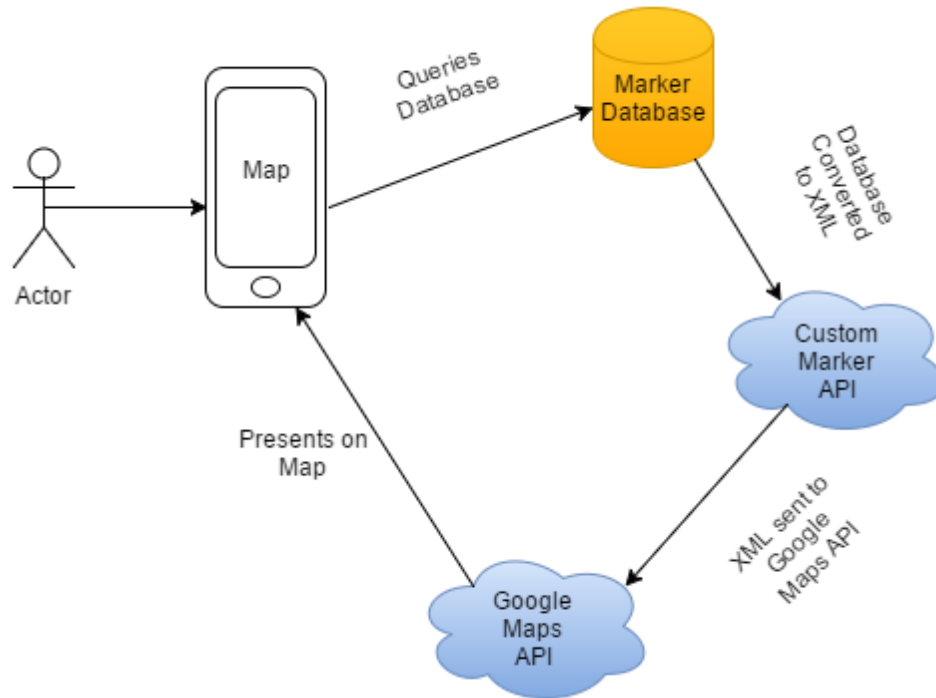


Fig 1.4

**Flow Description**

**Precondition**

The system is in initialisation mode when a user goes to the maps page.

**Activation**

This use case starts when a user the queries the system for offers in the area.

**Main flow**

1. The system allows the user to query the Google maps API which connects to the database and returns data from offers in the area. The database uses an API which returns data to the Google Maps API using XML.

**Alternate flow**

A1 : Location Turned Off

1. The system will throw an error is the users mobile device has the location turned off.
2. The user will enable device location.
3. The system will now continue at position two and send the location to the API

### **Exceptional flow**

E1 : No Data in area

1. The system will return no data in a specific area.
2. The will expand the radius of search.
3. The use case continues at position 3 and returns data in the wider area.

### **Termination**

This is when the user gets the offers in the area they are in.

### **Post condition**

The system goes into a wait state when the user does not request more offers.

## **2.1.7 Requirement 5: Chat System Case Diagram**

### ***2.1.7.1 Description & Priority***

The Chat system is the lowest priority functional requirement. It is to allow user and business owners to communicate about offers and deals.

### ***2.1.7.2 Use Case***

#### **Scope**

The scope of this is to allow users to communicate via the app. This feature will help with clarification.

## Description

This use case describes how the chat system will work within the application.

## Use Case Diagram

The actor is a consumer.

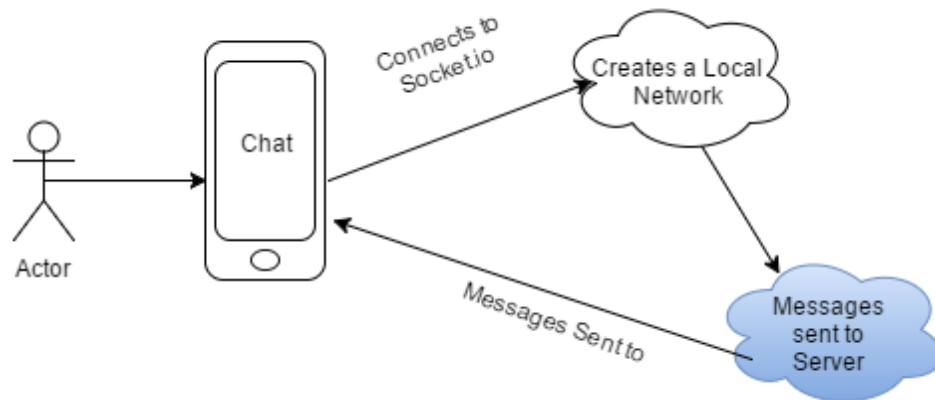


Fig 1.5

## Flow Description

### Precondition

The system is in initialisation mode when a user logs in to the system.

### Activation

This use case starts when a user goes to the messages page and sends a message.

### Main flow

1. The system logs the user in. They will go to the messages page where they will be able to message business owners.

### **Alternate flow**

A1 : No Online users.

1. The system will tell the user if there is no business owners available to message.
2. The user will wait until one becomes available.
3. The system will now allow the user to send a message.

### **Exceptional flow**

E1 : User is sending too many messages

1. The system will prevent the user from spamming the system with messages.
2. The user will be warned by the system and will be limited to a certain number of messages in a specified timeframe.
3. The use case will allow the user to send more messages once the time has passed.

### **Termination**

This is when the user sends a message.

### **Post condition**

The system goes into a wait state when the user logs out of the system.

## **2.1.8 Data requirements**

The application uses a MySQL database. The reason behind MySQL is because it is easily integrated into PHP. I am also using MySQL because it is a relational database. This makes it easier when linking the data in ways such as using foreign keys. This makes it easier when running PHP sessions and connecting the sessions to the unique users. I return data from the database and which populates the markers in the map.

### **2.1.9 User requirements**

**Smartphone/Android:** The application will run on any Smartphone with HTML5 capabilities. It also uses the phones GPS to get there location. There is an Android version written in Java which can only run on Android devices.

**Browser:** To run OfferApp you will need an internet browser unless you have an Android device. Desktop computers can access OfferApp by going to the website. JavaScript must be enabled for the map to work.

**Internet Access:** Every device capable of using OfferApp will also need internet access. This is because it uses a hosted database and also an API.

### **2.1.10 Environmental requirements**

I used SublimeText2(Sublime) A the text editor whilst programming OfferApp. I used WAMP as my server to test the application. WAMP also includes PHPMyAdmin and PHP5.5(PHP) which is what I needed to develop PHP. WAMP also includes MySQL which is what type of database I was using. I used Android Studio to build the Web View Application in Java. I used Composer and NPM to install dependencies.

## 2.2 Implementation

Below is a diagram of my system architecture. I aim to add in directions to the markers using dijkstra's algorithm.

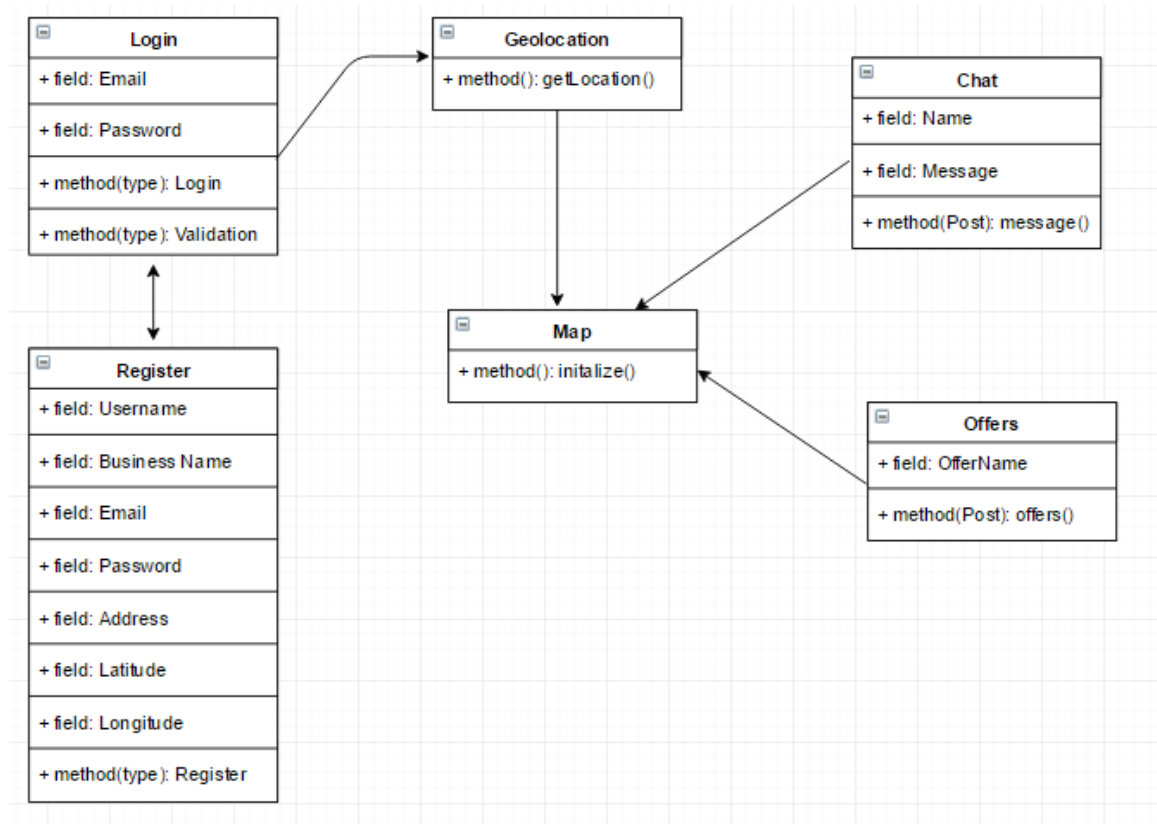


Fig 1.6

## Data Flow Diagram



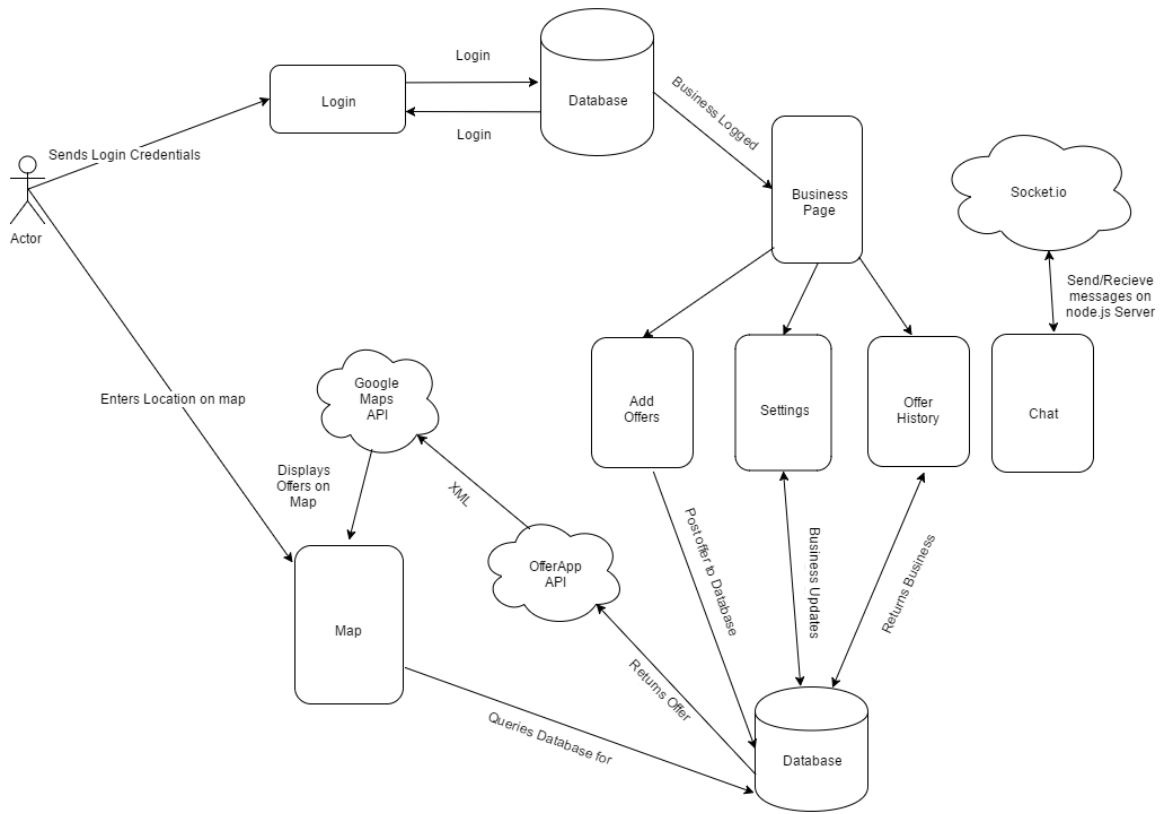


Fig 1.7

### 2.3 Design and Architecture

The Application is programmed in HTML5, CSS3, PHP5, JavaScript and uses a hosted MySQL database. The Application also uses an API which queries database and sends the formatted XML output to the Google maps API. Google Maps API then presents the information on the interactive map.

## **2.4 Hardware Architecture**

OfferAPP runs primarily on the web. This means it uses web servers to run. It uses a light footprint on the server as it is not data heavy. The mobile version uses Android OS as a dependency. This uses 12MB RAM and only one Android Activity as its using Webview. It works by mimicking a native Java app by running OfferApp as a web app running on the Android devices browser. It loads as fast as the phones internet connection. The server for the chat system runs Node.js (Node.js) and socket.io(Socket.io).

## **2.5 Software Architecture**

OfferApp uses PHP which is a scripting language that runs on web servers. It is a very fast and effective and powerful scripting language. It is also very easy to use with MySQL. I built the web service in PHP and it converts information in the database into XML which is then read by the Google Maps API. I used JavaScript for the interactive map. I also interacted with the Google Maps API using JavaScript. The database is a MySQL database which stores all information connected to the offers and businesses. I used D3.js which is a JavaScript library for creating dynamic diagrams tailored to the data they receive. Socket.io uses Node.js to run the chat system on OfferApp. It creates a network where the users can connect and chat about deals. This feature is hosted on cloud9(c9). For security measures I used form validation and MD5 hashing on the passwords.

## API Code

This code snippet sets the attributes of the API from the database and converts them into XML so they can be read by the Google Maps API.

```
header("Content-type: text/xml");

//appending attributes to nodes for xml output.

while ($row = @mysql_fetch_assoc($result)){

    $node = $dom->createElement("marker");

    $newnode = $parnode->appendChild($node);

    $newnode->setAttribute("name", $row['name']);

    $newnode->setAttribute("address", $row['address']);

    $newnode->setAttribute("offer", $row['offer']);

    $newnode->setAttribute("lat", $row['lat']);

    $newnode->setAttribute("lng", $row['lng']);

    $newnode->setAttribute("distance", $row['distance']);

}

echo $dom->saveXML();
```

This code Snippet generates a list of offers.

```
$query = sprintf("SELECT address, name, lat, lng, offer, ( 3959 * acos( cos(
radians('%s') ) * cos( radians( lat ) ) * cos( radians( lng ) - radians('%s') ) + sin(
radians('%s') ) * sin( radians( lat ) ) ) ) AS distance FROM markers HAVING
distance < '%s' ORDER BY distance LIMIT 0 , 20000",

mysql_real_escape_string($center_lat),

mysql_real_escape_string($center_lng),

mysql_real_escape_string($center_lat),
```

```
mysql_real_escape_string($radius));  
$result = mysql_query($query);  
if (!$result) {  
    die("Invalid query: " . mysql_error());  
}
```

## Markers

To add markers to the map I used JavaScript and the Google Maps API. The marker pulls Business name, Address and Offer Name. A variable is created which stores the HTML information in relation to the offers. When the user clicks the marker is displays a text box with the HTML offer information. The code snippet below shows how it is implemented.

```
function createMarker(latlng, name, address, offer) {  
    var offerInfo = "<b>" + name + "</b> <br/>" + address + "</b> <br/> Offer: " +  
offer;  
    var marker = new google.maps.Marker({  
        map: map,  
        position: latlng  
    });  
    google.maps.event.addListener(marker, 'click', function() {  
        infoWindow.setContent(offerInfo);  
        infoWindow.open(map, marker);  
    });  
    markers.push(marker);  
}
```

}

## Web Services

OfferApp has its own API written in PHP which outputs data in XML. The API takes the markers added to the database by the business users and sets the values of each attribute relative to the Marker information. Then it converts it to XML which is readable by the Google Maps API. By entering the query "http://offerapp.netne.net/xml.php?lat=53.386900&lng=-6.204027&radius=1" It shows the Offer Name, Address, Business Name, Coordinates and Distance from the specified Parameters. For the searching of nearby business I followed the Google Maps Store Locator tutorial(Store Locator) which has an algorithm for calculating radius as distance from latitude and longitude.

This XML file does not appear to have any style information associated with it. The document tree is shown below.

```
<?xml version="1.0" encoding="UTF-8" ?>
<markers>
  <marker name="Blake LTD " address="16 Chanel Road" offer="Two for one Oranges" lat="53.386700" lng="-6.203740" distance="0.0182036426056986"/>
  <marker name="Blake LTD " address="16 Chanel Road" offer="Buy one get one free on Coffee" lat="53.386700" lng="-6.203740" distance="0.0182036426056986"/>
  <marker name="Blake LTD " address="16 Chanel Road" offer="20% off Websites" lat="53.386700" lng="-6.203740" distance="0.0182036426056986"/>
  <marker name="Blake LTD " address="16 Chanel Road" offer="geolocation" lat="53.386700" lng="-6.203740" distance="0.0182036426056986"/>
  <marker name="Blake LTD " address="16 Chanel Road" offer="Half Price Websites" lat="53.386700" lng="-6.203740" distance="0.0182036426056986"/>
  <marker name="Blake LTD " address="16 Chanel Road" offer="rob lenehans photoshop for free!" lat="53.386700" lng="-6.203740" distance="0.0182036426056986"/>
  <marker name="Allybat Ltd." address="1 Chanel Road" offer="50% off steak when you pay with Card" lat="53.386398" lng="-6.203150" distance="0.0500850235158581"/>
  <marker name="Allybat ltd." address="1 Chanel Road" offer="fat stacks Vol" cash" lat="53.386398" lng="-6.203150" distance="0.0500850235158581"/>
  <marker name="Goo Design" address="35 Saint Brendan's Avenue" offer="Design Consultation/ Imaging €20 session" lat="53.386398" lng="-6.201160" distance="0.123133060209573"/>
  <marker name="A'n'T autos" address="46 Glin Grove Coolock" offer="Car servicing from €59" lat="53.400299" lng="-6.203320" distance="0.926301688411647"/>
</markers>
```

Fig 1.8

## Chat Feature

OfferApp uses Socket.io which is a library for Node.js. It allows users to connect over a network and communicate. It prompts the user to enter their name and then adds them to the chat network. It uses a Node.js server which searches the network and listens for users to connect and send messages. This module of OfferApp runs on Cloud9 which is a web server.

## User Control and Management

I used sessions in PHP which uniquely identify each user. I used this function to sign up and register each user and also while posting offers. First a session is started as the user and identified. Then a variable is created which identifies the user so it can be used to return and post information related to the signed in user.

```
<?php
include 'db_connect.php';
session_start();
include_once 'dbconnect.php';
if(!isset($_SESSION['user']))
{
    header("Location: login.php");
}
$res=mysql_query("SELECT * FROM b_users WHERE
user_id=".$_SESSION['user']);
$userRow=mysql_fetch_array($res);
?>
```

When adding offers into the database I created a SQL query which took the user id of the person signed in. It done this from the users table. Next the query posted the user ID into the markers table as a foreign key. Finally It sent the rest of the offer information. Below is the code snippet.

```
$user_id= $_SESSION['user'];
```

```

// attempt insert query execution

$sql = "INSERT INTO markers (name, address, offer, lat, lng, user_id) VALUES
('$business_name', '$address', '$offer', '$lat', '$lng', '$user_id)";

if(mysqli_query($link, $sql)){

    echo "Records updated successfully.";

    header("Location: settings.php");

} else{

    echo "ERROR: Could not able to execute $sql. " . mysqli_error($link);

}

```

I used similar code to implement the settings page and update the database.

### **Dynamic Data**

I used D3.js(D3.js) which is a library for javascript to display visual graphs based on the inputted data. I used a library called LiquidFillGauge(D3.js) to display the number of offers each user had in the database. It executes an SQL count and returns the number of offers. This number is inputted into the system and it displays it visually. Below is the code snippet for this.

```

var config5 = liquidFillGaugeDefaultSettings();

    config5.circleThickness = 0.2;

    config5.circleColor = "#6DA398";

    config5.textColor = "#0E5144";

    config5.waveTextColor = "#6DA398";

    config5.waveColor = "#246D5F";

    config5.textVertPosition = 0.52;

    config5.waveAnimateTime = 3000;

    config5.waveHeight = .3;

```

```
config5.waveAnimate = true;
config5.waveCount = 2;
config5.waveOffset = 1;
config5.textSize = 1.2;
config5.minValue = 7;
config5.maxValue = 10
config5.displayPercent = false;
var gauge6 = loadLiquidFillGauge("fillgauge6",
```

//This piece of code is written in PHP and it returns the number of offers created by the signed in user.

```
<?php $result = mysql_query("SELECT COUNT(*) FROM markers WHERE
user_id=".$_SESSION['user']) or die(mysql_error());
if (!$result) echo mysql_error();
$row = mysql_fetch_row($result);
echo $row[0];
?> , config5);
```

### **Database Structure:**

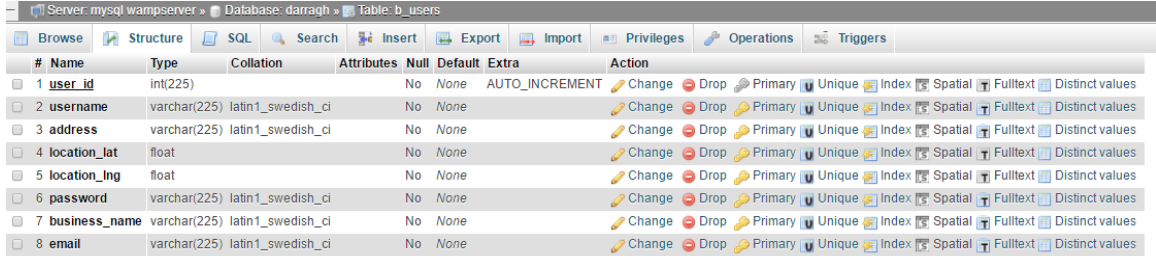
OfferApp uses a MySQL database. It is managed using PHPMyAdmin. The schema of the database is very complex. I used two tables. One table is for business users and the other is for the markers. The tables relate to each other and are both needed for the application to work.

### **Business Table**

The business users table stores all information related to the businesses. Information such as Business ID, Business Name, username, email and address



of the business as well as latitude and longitude coordinates. The business table sends some of its data to the Offer table once an offer is created.



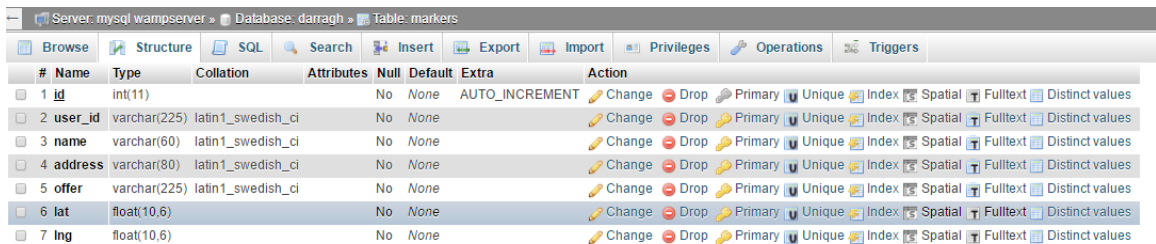
The screenshot shows the MySQL database structure for the table 'b\_users'. The table has 8 columns: user\_id, username, address, location\_lat, location\_lng, password, business\_name, and email. Each column has specific attributes like type, collation, nullability, and default values. The 'Action' column provides options for Change, Drop, Primary, Unique, Index, Spatial, Fulltext, and Distinct values.

#	Name	Type	Collation	Attributes	Null	Default	Extra	Action
1	user_id	int(225)			No	None	AUTO_INCREMENT	Change Drop Primary Unique Index Spatial Fulltext Distinct values
2	username	varchar(225)	latin1_swedish_ci		No	None		Change Drop Primary Unique Index Spatial Fulltext Distinct values
3	address	varchar(225)	latin1_swedish_ci		No	None		Change Drop Primary Unique Index Spatial Fulltext Distinct values
4	location_lat	float			No	None		Change Drop Primary Unique Index Spatial Fulltext Distinct values
5	location_lng	float			No	None		Change Drop Primary Unique Index Spatial Fulltext Distinct values
6	password	varchar(225)	latin1_swedish_ci		No	None		Change Drop Primary Unique Index Spatial Fulltext Distinct values
7	business_name	varchar(225)	latin1_swedish_ci		No	None		Change Drop Primary Unique Index Spatial Fulltext Distinct values
8	email	varchar(225)	latin1_swedish_ci		No	None		Change Drop Primary Unique Index Spatial Fulltext Distinct values

Fig 1.9

## Offer Table

The offer table stores all information related to the offers. Information such as Offer ID, Offer Name, address, latitude and longitude coordinates. It also stores information related to the business who posted the offer. The user id from the Business table is sent to the markers table once a business creates a marker. This makes it possible to connect a specific marker to a business, thus allowing them to view their previous offers.



The screenshot shows the MySQL database structure for the table 'markers'. The table has 7 columns: id, user\_id, name, address, offer, lat, and lng. Each column has specific attributes like type, collation, nullability, and default values. The 'Action' column provides options for Change, Drop, Primary, Unique, Index, Spatial, Fulltext, and Distinct values.

#	Name	Type	Collation	Attributes	Null	Default	Extra	Action
1	id	int(11)			No	None	AUTO_INCREMENT	Change Drop Primary Unique Index Spatial Fulltext Distinct values
2	user_id	varchar(225)	latin1_swedish_ci		No	None		Change Drop Primary Unique Index Spatial Fulltext Distinct values
3	name	varchar(60)	latin1_swedish_ci		No	None		Change Drop Primary Unique Index Spatial Fulltext Distinct values
4	address	varchar(80)	latin1_swedish_ci		No	None		Change Drop Primary Unique Index Spatial Fulltext Distinct values
5	offer	varchar(225)	latin1_swedish_ci		No	None		Change Drop Primary Unique Index Spatial Fulltext Distinct values
6	lat	float(10,6)			No	None		Change Drop Primary Unique Index Spatial Fulltext Distinct values
7	lng	float(10,6)			No	None		Change Drop Primary Unique Index Spatial Fulltext Distinct values

Fig 2.0

## **2.6 Testing**

To test OfferApp I used many testing techniques including unit testing, Consumer testing(Wammi) and Pingdom(Pingdom). The purpose of the testing process was to find errors, bugs, undocumented features and performance evaluation.

### **Unit Testing**

Unit testing is the process of testing each application feature extensively. I asked some Software Engineers to use OfferApp and recorded I the results. I used unit testing on the login and register system, finding offers on the map, on android with the Webview APK, adding new and reviewing previous offers, user settings and on Geolocation.

### **Login/Register**

They tested the login/register by creating accounts while entering incorrect credentials. They left fields blank while registering and left out @ symbols from the email field. The system handled these issues correctly by not allowing the user to register when leaving fields blank and also recognising the missing @ symbol from the email.

### **Map**

To test the map they entered in locations of various places. The map connected to the Google Maps API and located the positions. If a position was not found it would prompt a JavaScript Alert Box which would give a message displaying that the location was not found. The map performed well and notified the user when a location was not found. The distance parameter also performed correctly as it allowed the user to filter using distance from offers near to their searched location. It correctly populated a list of offers ranging from closest to furthest away.

### **Offer Management**

To test the system for inputting offers and retrieving them from the system they each added a list of offers to the map. There was no errors with posting the offers to the map.

### **Geolocation**

OfferApp uses Geolocation to pick up offers automatically. There was some errors with this system as Google have depreciated the use of Geolocation in Chrome version 50 without a valid SSL (https). This issue only came apparent in May. It started doing my testing using Mozilla Firefox as Geolocation is allowed using that browser.

### **Consumer Testing**

I performed two tests when doing consumer testing. The first test was a User acceptance test. I asked three individuals to perform eight tasks on OfferApp. To get a broader spectrum of results I tested with three different skill levels. The first was an IT project manager. The second was my dad who is a moderate user and the final was my aunt who owns a business but doesn't have much knowledge when it comes to technology. I timed each user as they performed the tasks and I also did not help them in any way. When they completed the test I then asked them to rate the usability on a scale of 0-10. 10 being the most user friendly and 0 being virtually unusable. Below is a table showing the results of the consumer testing. I defined an error as clicking the wrong page.

Questions	Advanced	Intermediate	Weak
Create an account and sign in	No errors	1 Error	1 Error
Change the name of the business	No errors	No errors	No Errors
Add an offer	No errors	No errors	No Errors
Add a second offer	No errors	No errors	No Errors
Find offers on the map (Which you made)	No errors	No errors	No Errors
View your created offers	No errors	No errors	1 Error

Delete all offers and sign out	No errors	No errors	No Errors
Time taken and score.	3mins (9/10)	7mins (9/10)	11mins (8/10)

## Wammi

Wammi(Wammi) is used to measure website user experience. It is similar to the Software Usability Scale (SUS). Wammi contains a list of questions about a particular website. I asked some independent adjudicators who have never seen OfferApp to use the website for a day. Then I asked them to fill out the Wammi Survey and recorded the results.

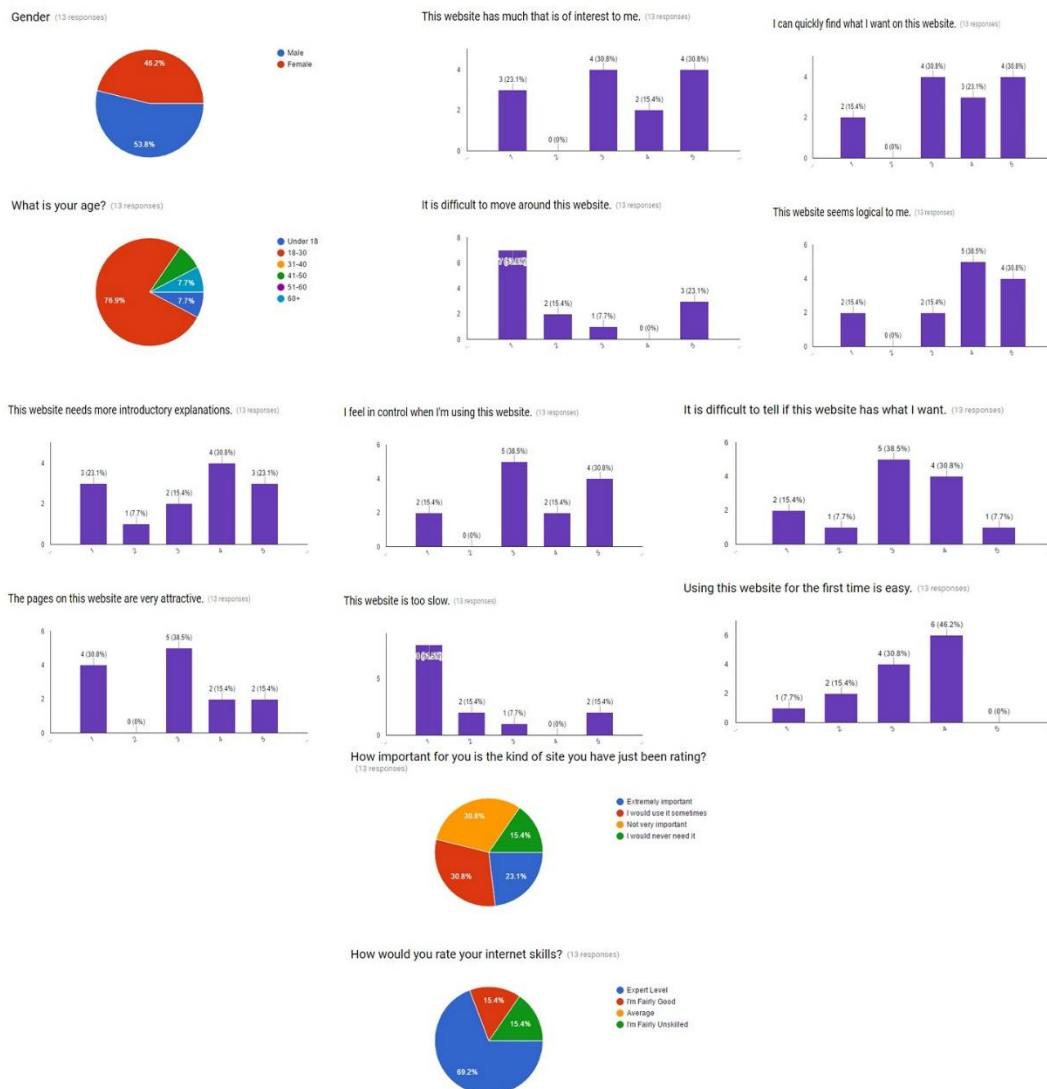


Fig 2.1

## Blackbox Testing

Since OfferApp runs primarily on the web I used Pingdom to test it. Pingdom runs your website and gives it a score out of 100 based on performance. OfferApp obtained a score of 85/100. The test looks at the footprint of the site. It measures cache, load time, website size, file requests and external file requests such as API's.

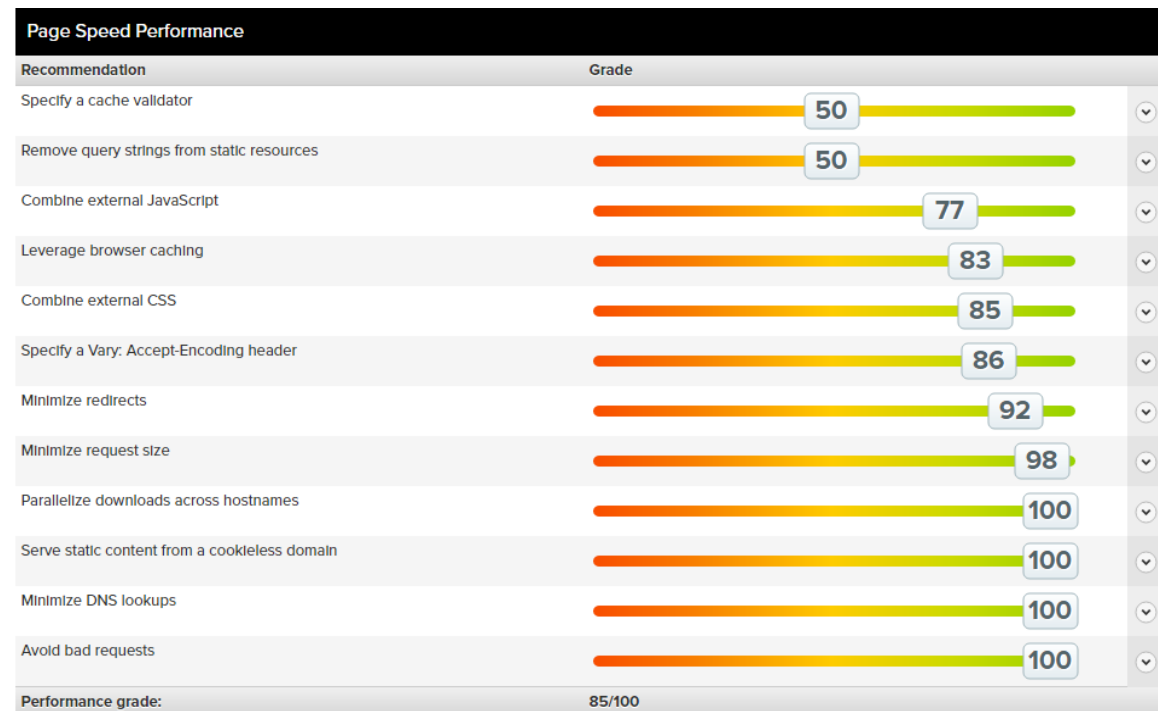
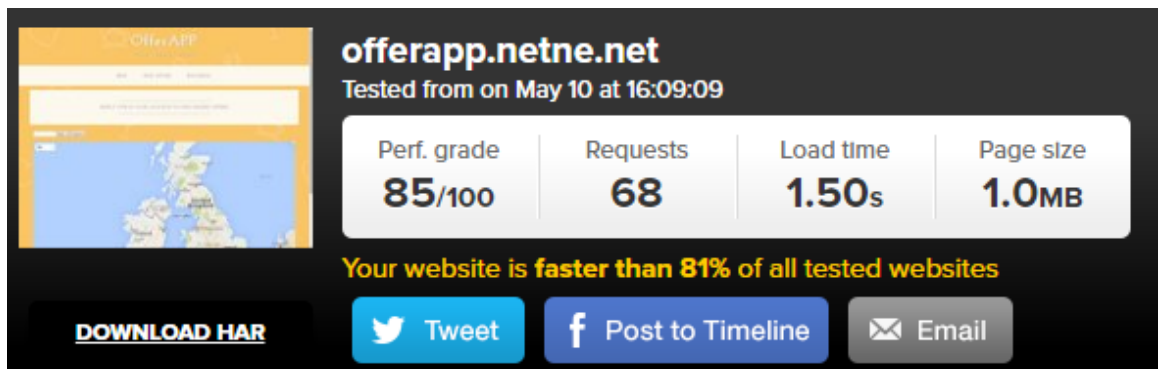
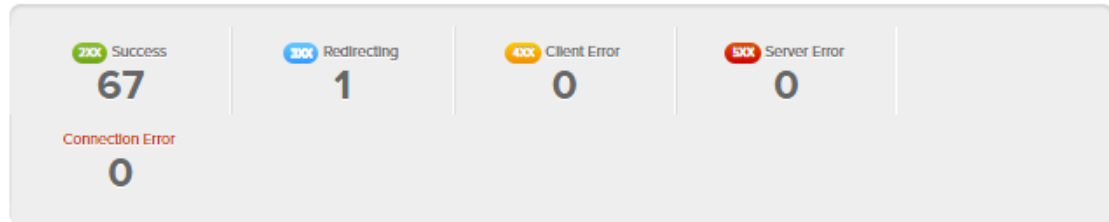
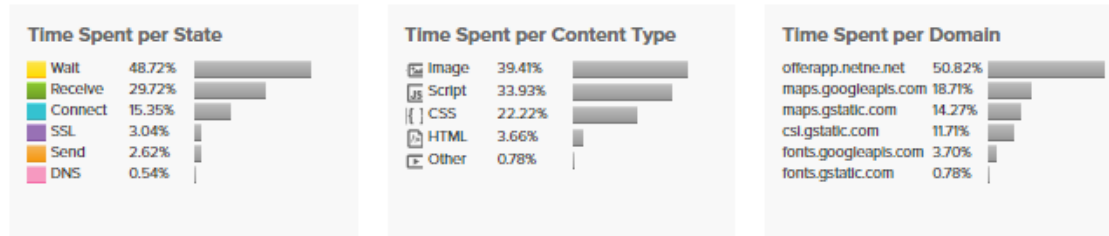


Fig 2.2

Server Response Code



Load Time Analysis



Size Analysis



Request Analysis

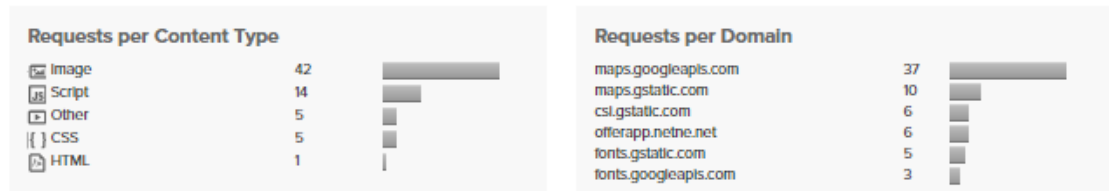


Fig 2.3

Evaluation

Overall OfferApp performed very well in the tests. I got some valuable feedback requesting to change some UI layouts. I updated the UI to rectify the issues. Going by the Pingdom results OfferApp is performing very well in relation to other websites tested by Pingdom.

## 2.7 Graphical User Interface (GUI) Layout

For the User Interface I used Bootstrap(Bootstrap). This is a CSS3 framework developed by Twitter. It is a responsive framework which uses jQuery(jQuery) to create mobile optimized websites.

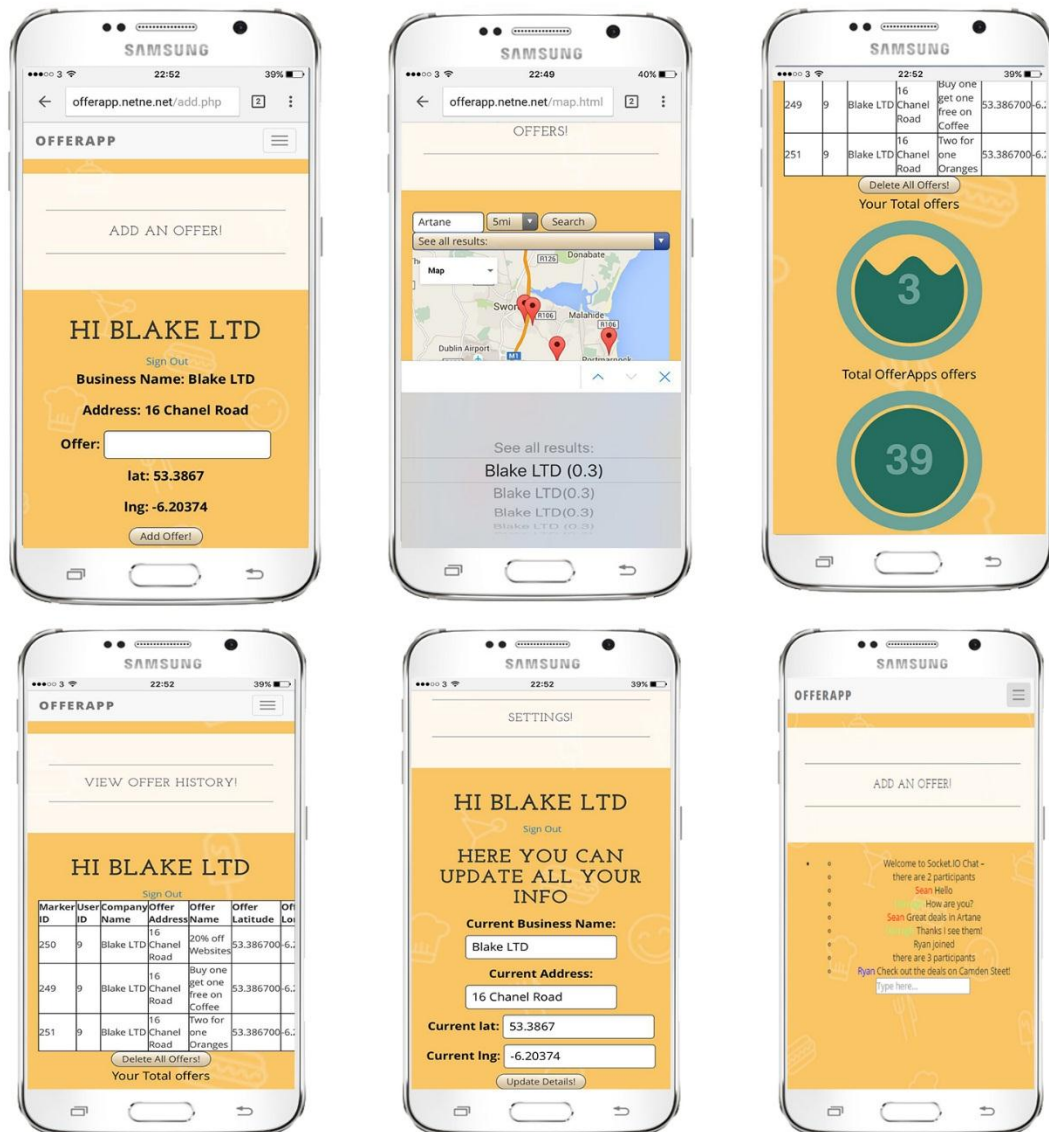


Fig 2.4

### **3 Conclusions**

Overall I am very happy with the outcome of OfferApp.

- I succeeded in meeting all of the applications functional requirements.
- OfferApp got positive feedback from all users.
- I aim to add more features further down the line.
- I am very happy with my API



## **4 Further development or research**

The application could become fully native with the correct time and development costs. OfferApp could eventually become a product used like a marketing tool for businesses to promote their latest offers and deals. OfferApp could include special offers which require QR codes to avail of. An example would be the first 50 people with the QR code gets a free drink in a particular nightclub.

- I aim to further develop OfferApp and tailor it more for commercial usage.
- I would like to expand it on to all three major platforms, iOS, Android and windows phone.
- I would like to use OfferApps API as a separate module which could be used for business use.
- Eventually licensing OfferApp for company is my main goal.

## 5 References

### 5.1 References

Bostock, Mike. "D3.js - Data-Driven Documents". *D3js.org*. N.p., 2016. Web. 11 May 2016.

Claridge, Jurek. "WAMMI - Home". *Wammi.com*. N.p., 2016. Web. 11 May 2016.

contributors, phpMyAdmin. "Phpmyadmin". *phpMyAdmin*. N.p., 2016. Web. 11 May 2016.

Creating a Store Locator with PHP, MySQL & Google Maps. "Creating A Store Locator With PHP, Mysql & Google Maps". *Google Developers*. N.p., 2016. Web. 11 May 2016.

Foundation, Node.js. "Node.js". *Nodejs.org*. N.p., 2016. Web. 11 May 2016.

"Google Maps Apis | Google Developers". *Google Developers*. N.p., 2016. Web. 11 May 2016.

Mark Otto, and Bootstrap contributors. "Bootstrap · The World's Most Popular Mobile-First And Responsive Front-End Framework.". *Getbootstrap.com*. N.p., 2016. Web. 11 May 2016.

"PHP: Hypertext Preprocessor". *Php.net*. N.p., 2016. Web. 11 May 2016.

"Pingdom - Website Monitoring Made Easy". *Pingdom*. N.p., 2016. Web. 11 May 2016.

"Socket.IO". *Socket.io*. N.p., 2016. Web. 11 May 2016.

"Wammi Testing Questions". *Wammi*. N.p., 2016. Web. 11 May 2016.

"World Wide Web Consortium (W3C)". *W3c.org*. N.p., 2016. Web. 11 May 2016.

Blake, Darragh.

Technical Report Midpoint Complete. Print.

Blake, Darragh.

Project Analysis And Design. 2015. Print.

## 6 Appendix

### 6.1 *Project Proposal*

#### Objectives

Summary:

Offer App is a application which gives you offers and deals based on your real time location. It will have features for users to connect to each other and do different activities listed in the app together. You will be able to see where each user is on your map.

**Real-time Location:** The application will aim to have real-time location updating to know exactly where the user is to configure deals in their area.

**Deal List:** The application will have a list of current deals and discounts in various stores around the area to which the user is currently located. It will also have a list of the most popular deals in each area. This will be on the users dashboard. The user will be able to filter by "Food & Drink" and "Leisure". It will get this information from data scraping other websites and from a system which allows business owners to input there offers into the backend of the application.

**Deal Map:** I aim to create a map with the users location on it. It will have the users location as a red balloon and the other deals and discounts will be yellow. You will also be able to see other users in your area on the map as yellow balloons.

**User Experience/Interaction:** The application will have a clean UI and login system. This will include uploading a photo of themselves as there profile picture. The app will also include a chat system for users in the area to communicate on different deals and to meet up and do an activity listed in the app.

## 7 Background

I noticed a niche in the market for an application which gave you deals based on your real time location. This application will be popular especially for students as they are always looking for deals and discounts.

I did some research on applications similar and found none of them will give you offers as you're in the location. They allow you to search and filter by location but none offer what offer app can deliver. I feel there is a certain novelty of an app that follows you around in order to find you deals and discounts.

This would also work very well for tourists. As they come over they may not know what to do in the area and offer app will provide a list of activities near you, at a discounted rate.

I also found that there is no app that has a feature to allow you to message others via a map location. This adds a nice amount of user to user activity as you can connect with other users of offer app beside you. This could be used for general discussion of an event nearby or even to go to an event together.

I have a good knowledge and interest in location based services. This is one of the reasons I chose this project. I also have a keen interest in API's and web services. I wanted to include some sort of API Mash-Up. Which is what this application will include.

I would like to learn more about data scraping so I am planning on creating a service which scrapes certain deals from websites. This feature would be best implemented once I could collect more data.

I have written some basic location based applications, including an active app on the Android store. I have also done some work with the Google maps API.

## **8 Technical Approach**

I created the web application first, I then began to create an android application using android Webview.

## **9 Special resources required**

OfferApp does not require much special resources. The database must be hosted so people can avail of the business offers. The website is hosted on 000ewebhost.com. I used Cloud9 to host the socket.io chat system.

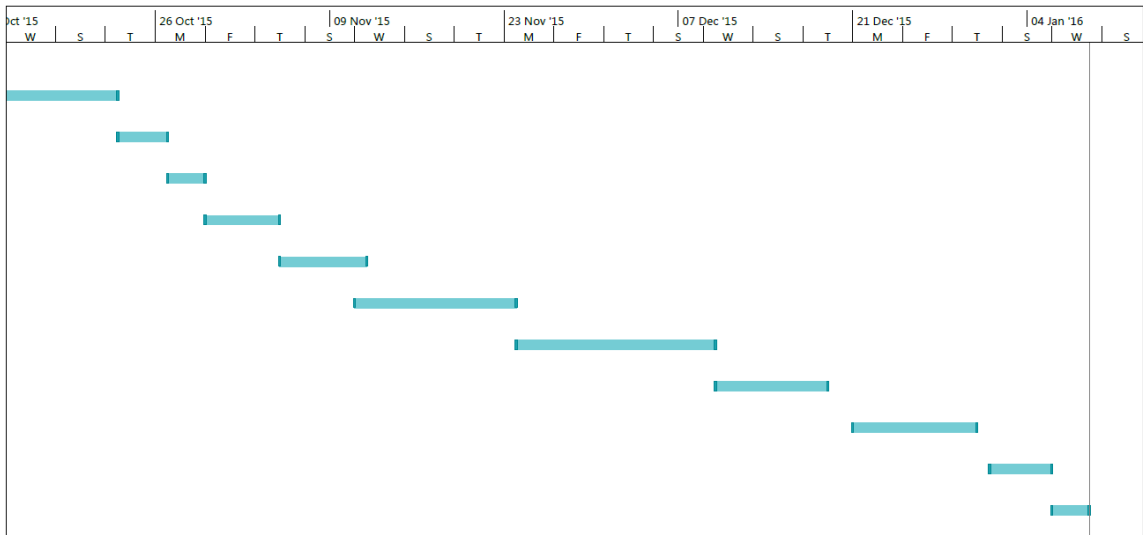
## **11 Project Plan**

Gantt chart using Microsoft Project with details on implementation steps and timelines

ID	Task Mode	Task Name	Duration	Start	Finish	Predecessors	28 Sep '15	12 Oct
1		Obtain Hosting and Domain	2 days	Wed 07/10/15	Thu 08/10/15			
2		Research + Prototype Location Programming	10 days	Fri 09/10/15	Thu 22/10/15			
3		Get backbone Template	2 days	Fri 23/10/15	Mon 26/10/15			
4		Set Up database	3 days	Tue 27/10/15	Thu 29/10/15			
5		Create User Login System	4 days	Fri 30/10/15	Wed 04/11/15			
6		Set Up Google Map + API	5 days	Thu 05/11/15	Wed 11/11/15			
7		Map User to Location	9 days	Wed 11/11/15	Mon 23/11/15			
8		Add Offers to Google maps via DB	12 days	Tue 24/11/15	Wed 09/12/15			
9		Create User chat system	7 days	Thu 10/12/15	Fri 18/12/15			
10		Add in Weather API	8 days	Mon 21/12/15	Wed 30/12/15			
11		Infrastructure Testing	3 days	Fri 01/01/16	Tue 05/01/16			
12		User Testing	3 days	Wed 06/01/16	Fri 08/01/16			

Project: Offer\_App  
Date: Fri 02/10/15

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



Project: Offer\_App  
Date: Fri 02/10/15

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

## **10 Technical Details**

PHP – I used PHP as the main language for my application. This was used for writing to the database, pulling information from the database and also for user management using sessions.

HTML5 – I used HTML5 as the backbone as this is a web application. This also will include the real time location tracking of the user.

JAVASCRIPT – I used this for creating the map using the Google Maps API.

MySQL – This was used for the database.

Google Maps API - This is what I used for the map itself.

D3.JS– This was used to represent data taken from users and offers.

Bootstrap – Bootstrap was used to keep the application mobile friendly and responsive.

XML - I used XML for creating the OfferApp API.

Socket.io/Node.js - These were used to create the chat system.

## **11 Evaluation**

I will perform many tests on my application, both during and upon completion. I will use black box testing for testing the functionality of the Application. I will use white box testing for testing the internal structure of the application. Finally I will have users use the application to test for problems from a UI point of view. I will use bugzilla for finding bugs in the code which could lead to other bigger problems.

### **1.1 Monthly Journals**

## **Reflective Journal September**

Student name: Darragh Blake x12467038

Programme (BSc (Hons) in Computing):

Month: September

### **2 My Achievements**

I started mocking up prototypes for different systems as soon as we were given time to work on our projects. I worked on building a login system and did some testing on different API's. I came up with the idea for my application.

The first step was to buy web hosting. I used 000webhost as they are a free hosting service. I also set up PHP my Admin and a WAMP server on my local machine. I got a domain to deploy and test my web application. Finally I set up a MySQL database.

The next step was to set up Angular.js as my front end to my application. I also downloaded bootstrap and Codeignitor the PHP framework. I decided to use sublime text as my editor for PHP and JavaScript. I downloaded and installed android studio for developing the push notifications and web view of my application.

I decided to use Google Maps API for my application so I set that up locally on my machine. After I was done setting up I watched some videos and tutorials on live tracking in the Google Maps API. I also looked at login/registration in Angular.js.

The final stage of setting up was creating a github and bitbucket account. I used these tools for version control of my project.

### **3 My Reflection**

I was procrastinating about my project ideas as I was hesitant about which type of application to build. I finally decided on my Geolocation based offer app. I felt this was the perfect mix of data basing, algorithms, API's and mobile development I could apply to an application.



I found that setting up version control was very important as its very easy to lose or break your application even through updates in frameworks etc

#### **4 Intended Changes**

Next month I intend to get a full working Geolocation based map running in real time alongside Google Maps API. I will look at creating a full login system with angular and a chat system using socket.io

I intend on writing up my technical specs for the project and lock down exactly how I am going to develop the application.

#### **5 Supervisor Meetings**

Date of Meeting: 14/10/2015

Items discussed: Version Control, Frameworks

Action Items:

My supervisor suggested we download and set up github for version control. I had already this created but I gave him access to my repository so he could review my progress. He suggested we use a framework for developing the application. He suggested slim for API's which I will look into over the next month.

#### **Reflective Journal October**

Student name: Darragh Blake x12467038

Programme (BSc (Hons) in Computing):

Month: October

#### **6 My Achievements**

I started looking into angular js as my front end for my application. I watched some tutorials on developing Angular for a login and connecting PHP to it. I wanted to

keep php as my back end and use it to authenticate users and for creating sessions for the API.

There was a problem with the hosting service I was using because they were hacked. Therefore I couldn't make any changes to my database until this was resolved.

I done some testing on my realtime location tracker. I went for a short drive to see if it was accurately tracking my position. It was but it was a little jumpy so I decided to change the update time to every seven seconds.

I pushed my code to github. My next step was to create a module that allowed business owners to sign up and add their location to the database. Then this would allow them to add the deals to the map and view them.

## **7 My Reflection**

I am happy with the amount of work and testing that was done this month. I do intend to getting more work done the following month and have a very basic version working to do some more testing.

Writing the technical specifications helped me get a much better understanding of how I was going to develop the application.

## **8 Intended Changes**

I intend on changing the speed of updating on the map or come up with a better way of tracking as it can be very buggy. I am going to use an API for my login as I do not want a local login system. This will help for portability and scalability.

## **9 Supervisor Meetings**

Date of Meeting: 21/10/2015

Items discussed: Android, API's

Action Items:

My supervisor suggested that I add some API functionality to the app such as the login. This will allow me to log in any users on any platform instead of a local login which can be found in angular js.

## **Reflective Journal November**

Student name: Darragh Blake x12467038

Programme (BSc (Hons) in Computing):

Month: November

### **10 My Achievements**

I began to develop the application user login using Angular.js. I successfully created it locally and then I used a Slim API in PHP and connected it to the database.

I done some testing on the database to see if it was hashing the passwords with MD5 which it was. I worked on researching some frameworks for the backend such as codeignitor.

I have done some basic research on the database design which I intend to complete next month. I done my Requirements Spec this month took a lot of time. I feel I completed it to a very good standard.

### **11 My Reflection**

I am happy with the amount of work and testing that was done this month. I am working on developing the database and the database design next month. I intend to get a basic version working by Christmas. This will be difficult with all of the assignments we have.

### **12 Intended Changes**

As of this month I do not have any intended changes as what I am doing is working out well.

### **13 Supervisor Meetings**

Date of Meeting: 11/11/2015

Items discussed: Requirements Spec

Action Items:

My supervisor reviewed my Requirements Specification and he gave me some feedback. The feedback was mostly positive and I applied some changes and submitted it.

### **Reflective Journal December**

Student name: Darragh Blake x12467038

Programme (BSc (Hons) in Computing):

Month: December

### **14 My Achievements**

I have built a beta version of my app using MySQL, PHP and JavaScript. The version allows users to search on the system for locations near to where they search. It shows a distance from the searched location to the relevant offer.

I have also built a front end in bootstrap and put the website live for people to try out.

### **15 My Reflection**

I am very pleased with the amount of work I got done this month. My aim is to keep moving forward at this pace and develop the best app possible.

Intended Changes

I intend on changing the front end from bootstrap to bootstrap and AngularJS.

### **Supervisor Meetings**

Date of Meeting: 11/12/2015

Items discussed: Project code

Action Items:

My supervisor reviewed my work and showed me some frameworks that I could possibly implement. He was happy with my progress.

### **Reflective Journal January**

Student name: Darragh Blake x12467038

Programme (BSc (Hons) in Computing):

Month: January

### **My Achievements**

I built a backend system to login and register users with user sessions. The session allows users to now add their own offers to the map rather than the DBA in the database. I have also added a settings page where users can update their information. The adding offers page populates the markers with user data before posting which allows them to only add in the offer.

### **My Reflection**

I am happy with the amount of work I got done in January. My aim is to keep moving forward to build a more advanced application than I originally planned.

### **Intended Changes**

I intend on doing more MySQL joins and foreign keys to allow users to update offers and also to create offers that delete after a certain time.

### **Supervisor Meetings**

Date of Meeting: 02/02/2016

Items discussed: Presentation and Progress.

Action Items:

I made some small changes to the presentation once my supervisor looked over it.

### **Reflective Journal February**

Student name: Darragh Blake x12467038

Programme (BSc (Hons) in Computing):

Month: February

#### **My Achievements**

I completed my presentation and worked on creating a system for OfferApp which allowed businesses to review using CRUD techniques their previous offers.

I began looking into testing the application and planning to implement my final feature which is a chat system.

#### **My Reflection**

I put a lot of effort into OfferApp in February and I am pleased with the overall performance. Although pressure is setting in I feel I will complete the Application to a high standard.

#### **Intended Changes**

I do not intend to change much, I want to be able to edit offers that are currently running, with CRUD functionality.

#### **Supervisor Meetings**

Date of Meeting: 26/02/2016

Items discussed: CRUD and chat.

Action Items:

I gave a brief rundown of my plans for February to my Supervisor who advised me on some methods to implement.

### **Reflective Journal March**

Student name: Darragh Blake x12467038

Programme (BSc (Hons) in Computing):

Month: March

### **My Achievements**

I completed most of my application in March. The application now has functionality to view previous offers and works very well.

### **My Reflection**

I exceeded myself in my plan to have a midpoint application built. The application now has more functionality than I initially anticipated. With now showing previous offers businesses created.

### **Intended Changes**

I do not intend to change much, I want to be able to edit offers that are currently running, with CRUD functionality.

### **Supervisor Meetings**

Date of Meeting: 20/03/2016

Items discussed: Exams and timeframe.

### **Action Items:**

I will put some of the project aside while I focus on my exams as I am ahead of my project schedule as planned for my exams.