

An Android App For Food Allergy Sufferers To Search For And Review Restaurants

DT265

Higher Diploma in Computing

**Name: Maeve Rooney**

**Supervisor: Michael Collins**

**<Second Reader>**

School of Computing

Dublin Institute of Technology

**<Date>**



Abstract

Declaration

I hereby declare that the work described in this dissertation is, except where otherwise stated, entirely my own work and has not been submitted as an exercise for a degree at this or any other university.

Signed:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

<Student Name>

<Date>

Acknowledgements

Body text

Table of Contents

1. Introduction 8

1.1.Background 8

1.2.Project Objectives 8

1.3.Project Challenges 8

2. Technologies Researched 8

2.1.Introduction 8

2.2.Eclipse and Java 9

2.3.RESTful Services andAndroid 10

2.4.Third party web server – x10hosting.com 11

2.5.FTP Program – CyberDuck 13

2.6.TextWrangler and php, mysql 14

2.7.Android platform 16

2.8.Google Maps API 18

3. Architecture & Design 20

3.1.Introduction 20

3.2.System Architecture 21

3.3.Use Case Design 24

3.4.Design Methodology 25

3.5.Development Environment 25

3.6.Web Server and MySQL Design 26

3.7.Local SQLite Design 29

3.8.User Interface Design 30

3.9.List of Features 33

4. Development & Implementation 33

4.1.Details of each component within the project, problems encountered and resolved, challenges overcome or worked around. 33

4.2.Identify key development components 33

5. System Validation 33

5.1.Testing: What testing was performed, why it was selected and what are the key use cases within the project. 33

6. Project Plan 33

6.1.Project Plan analysis and review of how it changed from the initial proposal including explanation of what changed and why, and suggestions on how to address this if the project was repeated. 33

7. Conclusion 33

7.1.Analysis of the projects key elements identify the key learning obtained from the project and recommendations and suggestions for how the work can be improved on continued into the future. 33

8. Bibliography 33

9. Appendix 34

­

*(Note: For bibliography use IEEE or Harvard referencing styles)*

**Problems**

* Starting activity from map overlay item- finding context
* Passing paramters to async task
* Using name pair values to post http request to php
* Getting context in itemized overlay class
* Pass restaurant id into new activity to change content dynamically
* Progress dialog while switching activities
* Phone and email
* Mergeadapter for listview with header in restaurantview
* Store passwords with salt encryption
* Use ajax to check if unique username or email already in use
* Favoutrite icons on map markers
* Check if user logged in on map and list view
* Show only restaurants with rating of more than 2 for each of users allergies
* Write review. Locate on map
* Add review to restauratn on map
* Place marker at users current location. Drag to change and make note of new geopoint
* Add heart to list view
* Blacklist
* Put rating bar indicator in alertbox

# Introduction

## Background

## Project Objectives

## Project Challenges

# Technologies Researched

## Introduction

## Eclipse and Java

Eclipse Development Environment

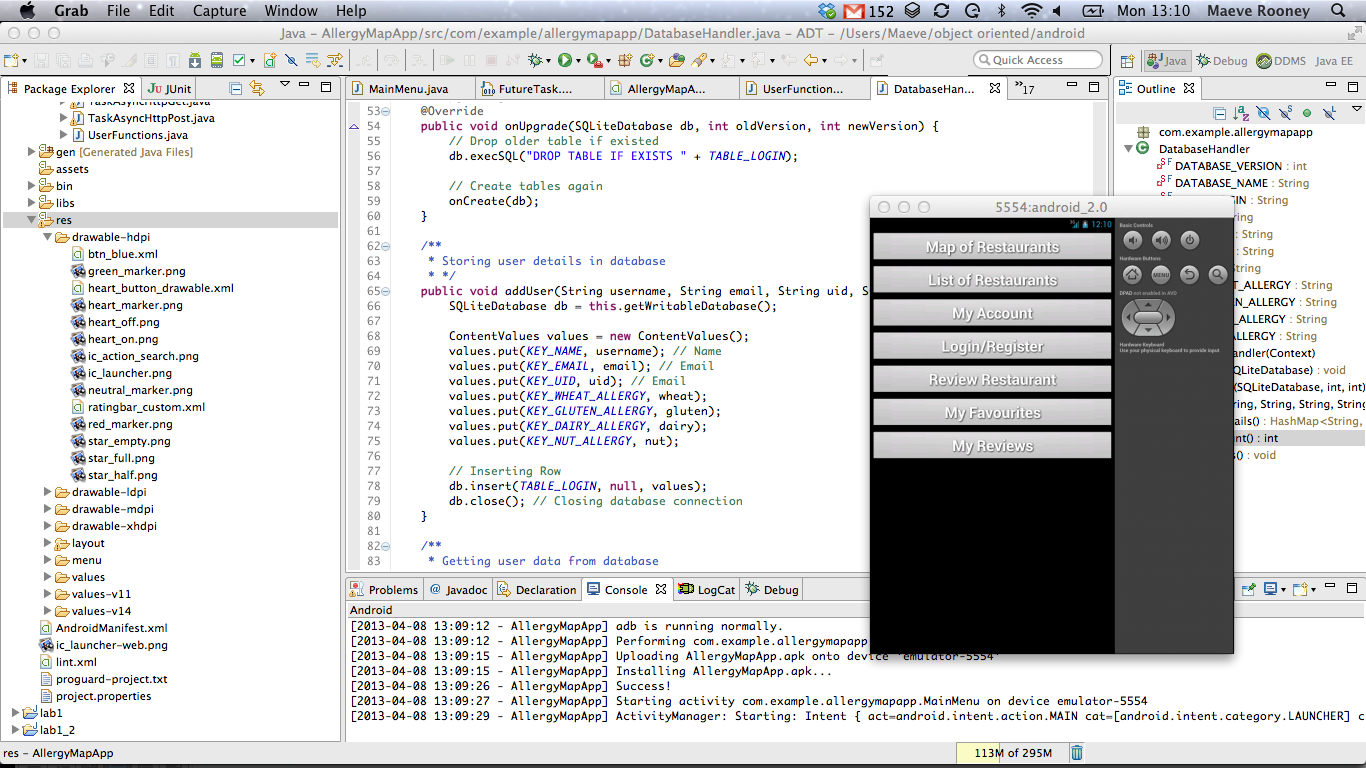


Figure 2‑1 - Eclipse for Mac with Android Emulator

**Development Device – Samsung Galaxy Ace**

Figure 2‑2 - Samsung Galaxy Ace

## RESTful Services andAndroid

The RESTful (Representational State Transfer) architecture is ideal for the purposes of creating an Android App with dynamic content. It allows loose coupling between different services, in this case an Android app and a website with a MySQL database. REST does not require XML parsing. Alternatively JSON (JavaScript Object Notation) can be used which is ideal as it is easier to parse in Java. JSON is more lightweight than XML. It uses brackets instead of tags and does not require a DTD (Document Type Definition) declaration.

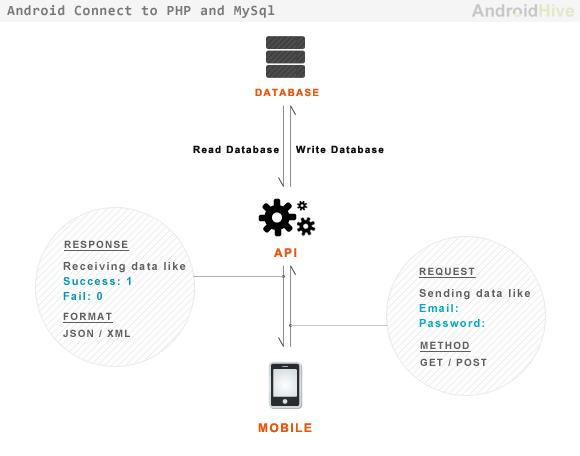
****

Figure 2‑3 - Android RESTful api

The Allergy Map App RESTful web API is implemented using the following aspects:

* The base URI for the API is http://maeverooney.x10.mx/\*
* The [Internet media type](https://en.wikipedia.org/wiki/Internet_media_type) used is JSON.
* The HTTP Methods used are GET and POST.
* The API is driven by Hypertext.
* It is Stateless. No client context stored on server between requests.
* The server and the android app are not dependant on each other. Separation of concerns.

## Third party web server – x10hosting.com

In order to run a RESTful service on the app it is necessary to setup a website with a SQL database. Many services offer free hosting. X10hosting.com offers such a service which includes MySQL databases and PHP hosting. Using these services it is easy to set up a simple website that can be used to store the content for my dynamic Android App. The content can then be retrieved and update using http calls to this website from the android device.

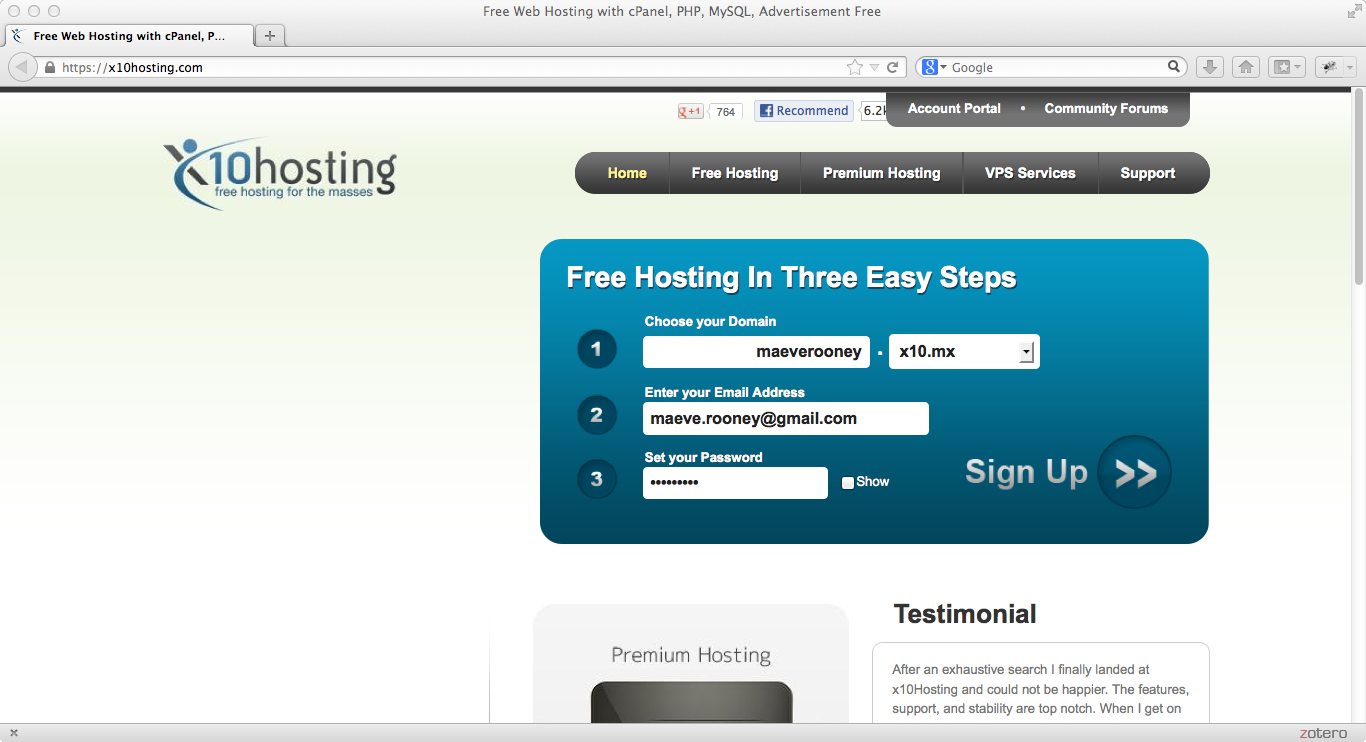


Figure 2‑4 - Signup page for hosting on x10hosting.com

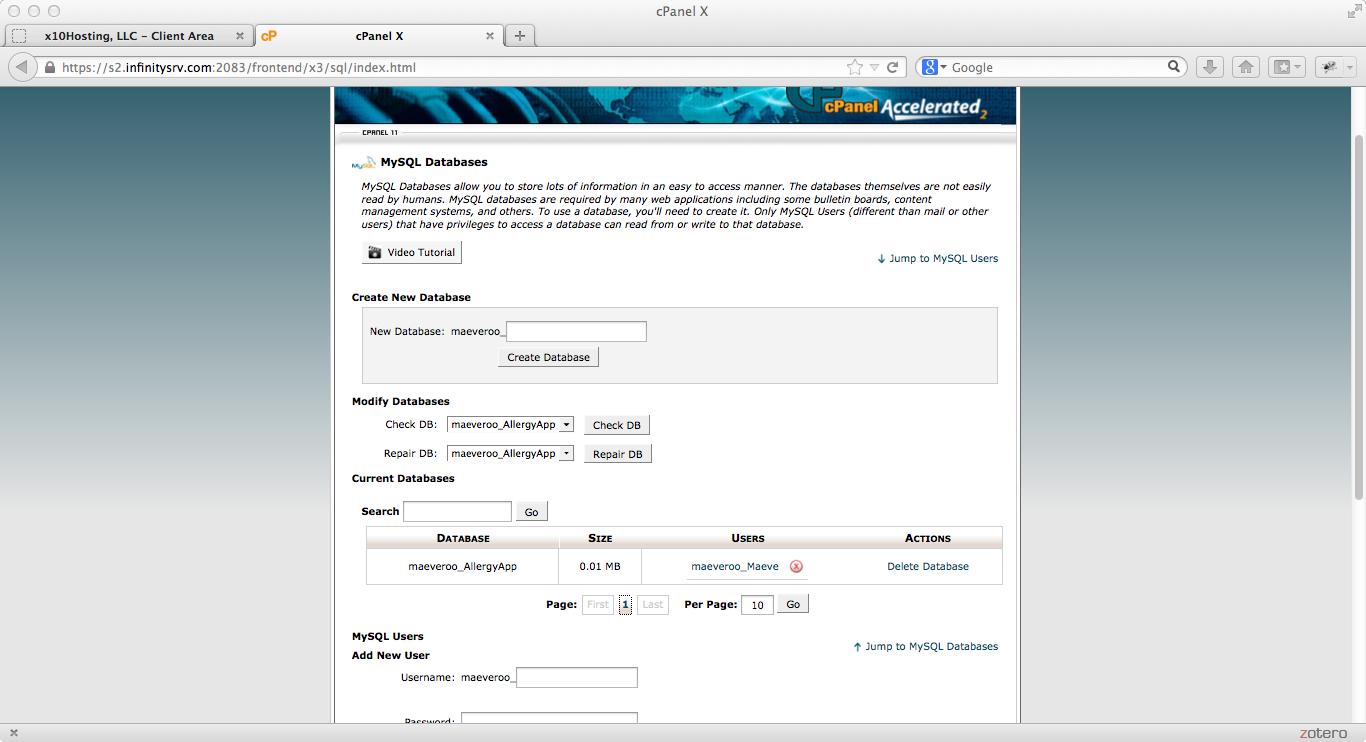
****

Figure 2‑5 - cPanel page to manage MySQL databases

## FTP Program – CyberDuck

Connect to web server to manage file transfer



Figure 2‑6 - CyberDuck manages file transfer

## TextWrangler and php, mysql

TextWrangler used to write php and html for web server

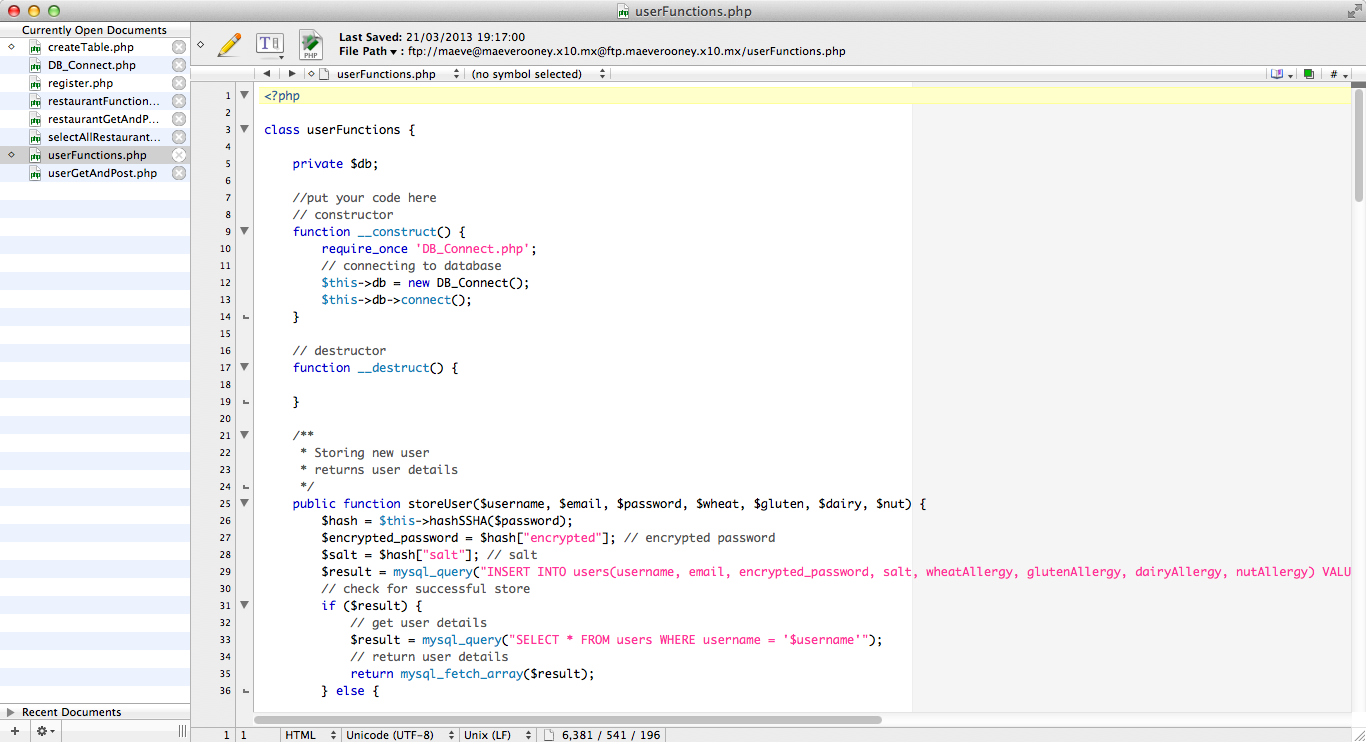


Figure 2‑7 - Textwrangler text editor

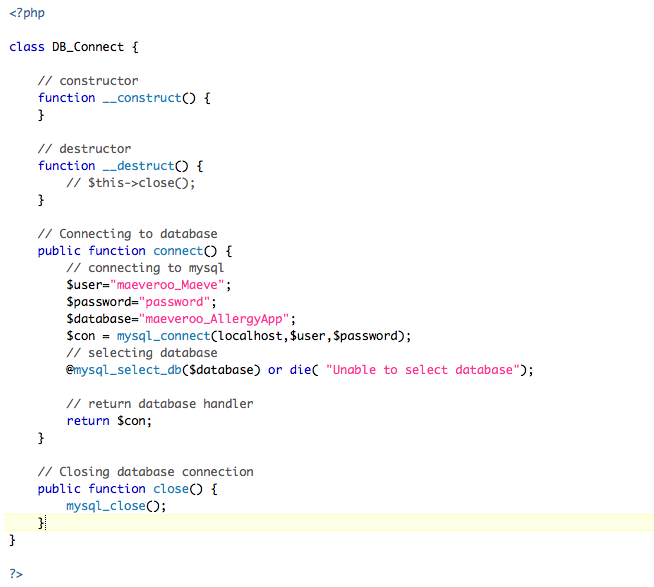
**PHP script to connect to MySQL database**

Figure 2‑8 - PHP script to connect to Database

## Android platform

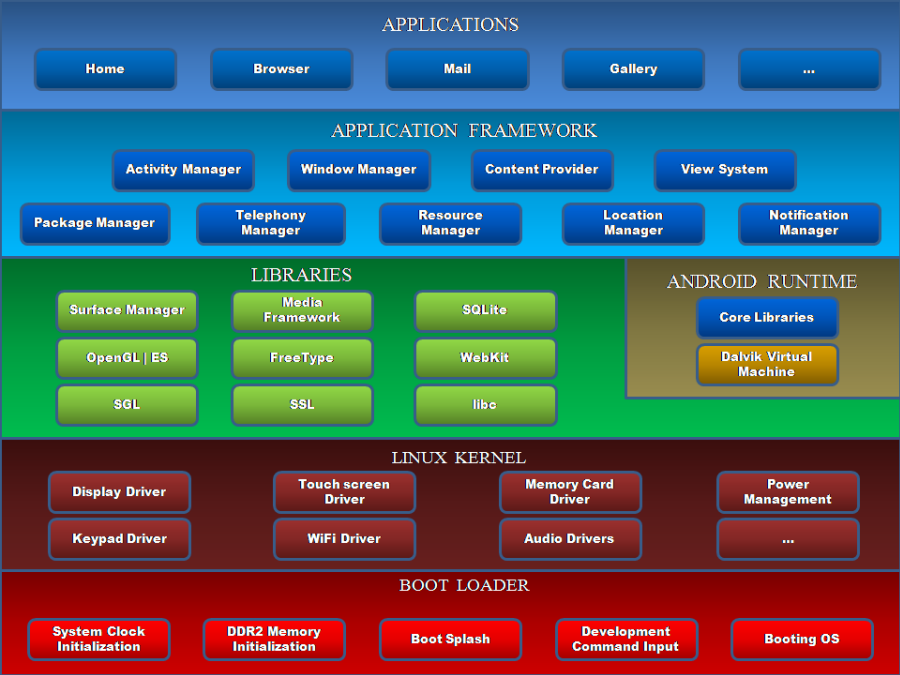


Figure 2‑9 - Android Architecture Diagram

**Versions of Android in Market 2013**

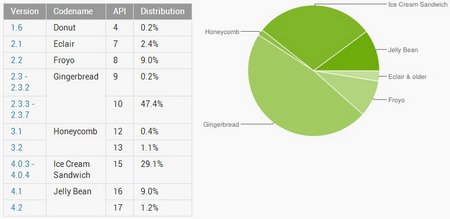
****

Figure 2‑10 - Market Share of Android Versions 2013

This app is optimised for the JellyBean (API 17) version but is working well on the Gingerbread version which the Samsung Galaxy Ace runs on.

## Google Maps API

To develop an Android app that will display Google Maps data using the API provided in the Maps external library, you must register with the service and get a Google Maps Android API v1 Key.

**Get a Google maps key for android**

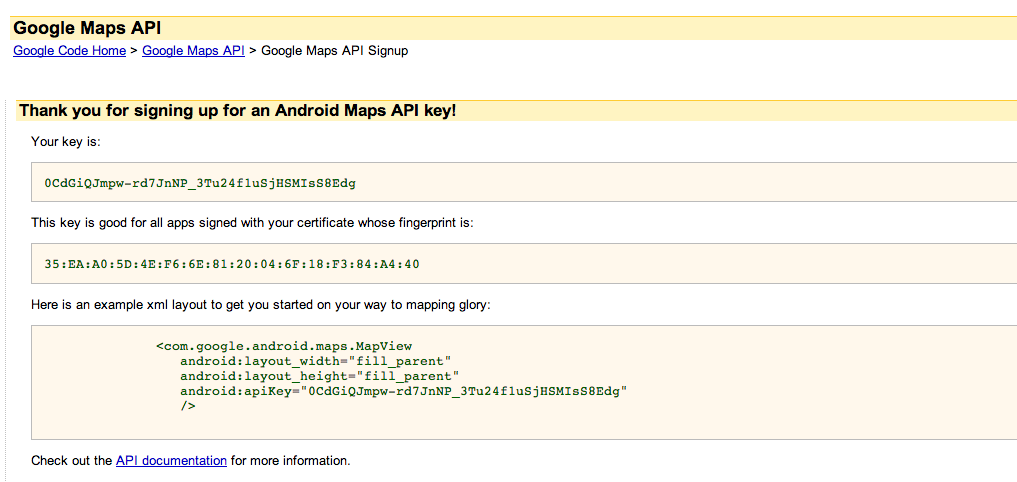
****

Figure 2‑11 - Getting Google Maps API key for Android

**Use key in xml layout of MapView activities**



Figure 2‑12 - Inserting Google API key into map layout xml

# Architecture & Design

## Introduction

## System Architecture

**Initial System Architecture Diagram**

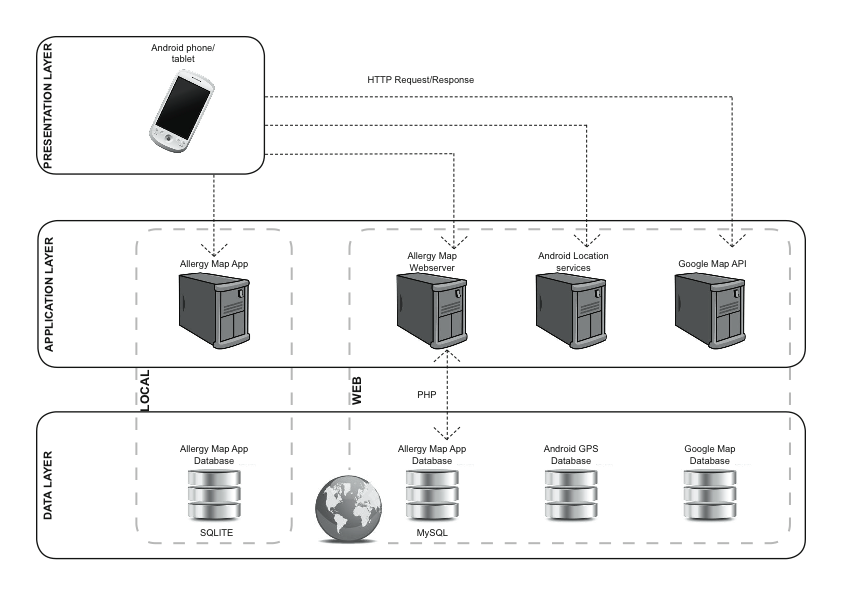
****

Figure 3‑1 - Intial System Architecture Design

**Final System Architecture Diagram**

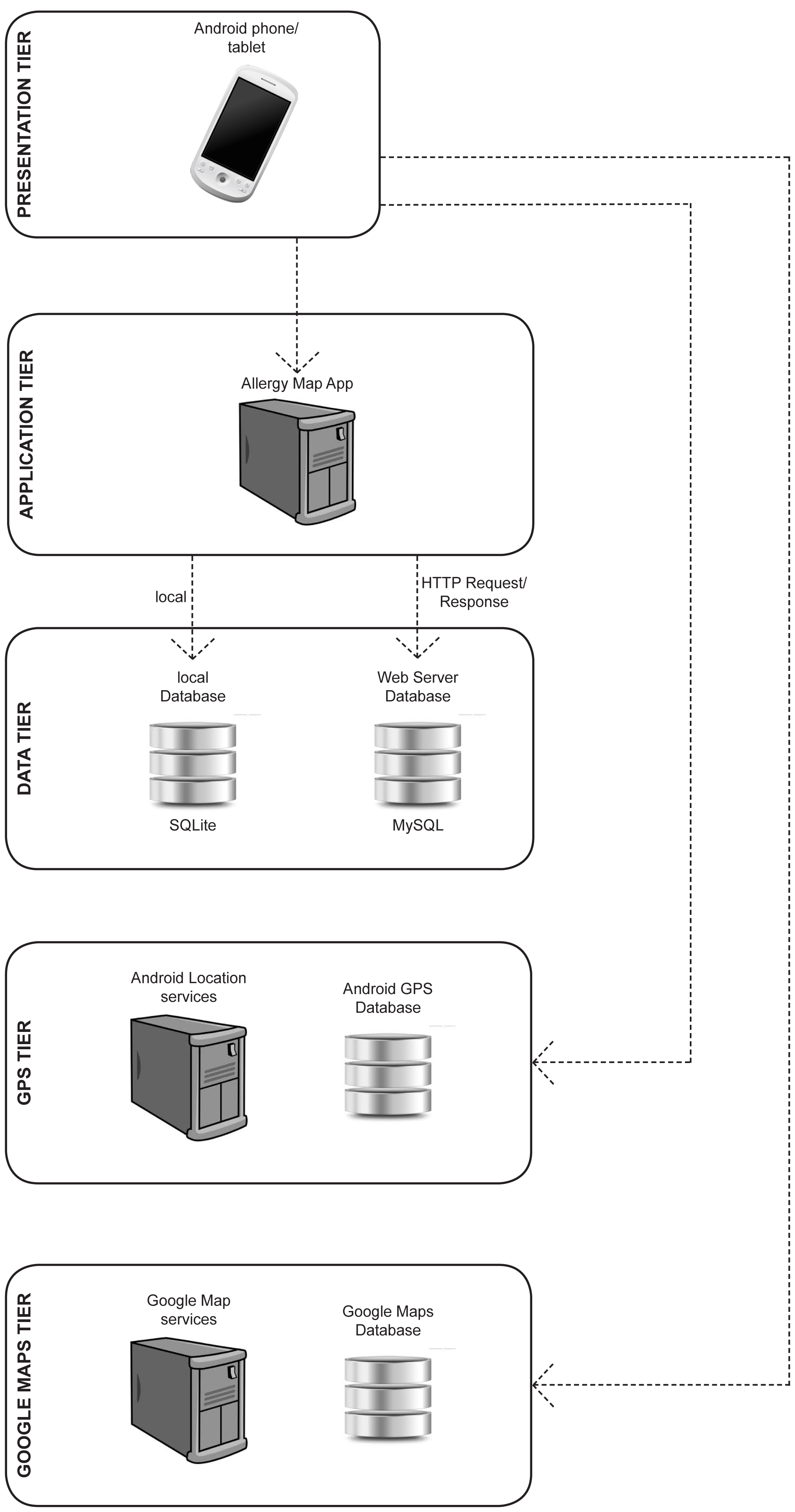


Figure 3‑2 - Final System Architecture Design

**5 Tiers**

1. Presentation Tier.
2. Application Tier.
3. Data Tier.
4. Android GPS Tier.
5. Google Maps Tier.

## Use Case Design

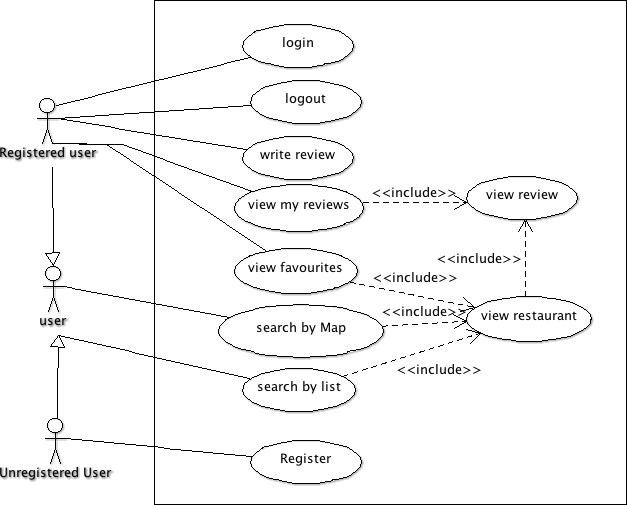
****

Figure 3‑3 - Use Case for user

## Design Methodology

## Development Environment

## Web Server and MySQL Design

**ERD Diagram for Web Server MySQL database**

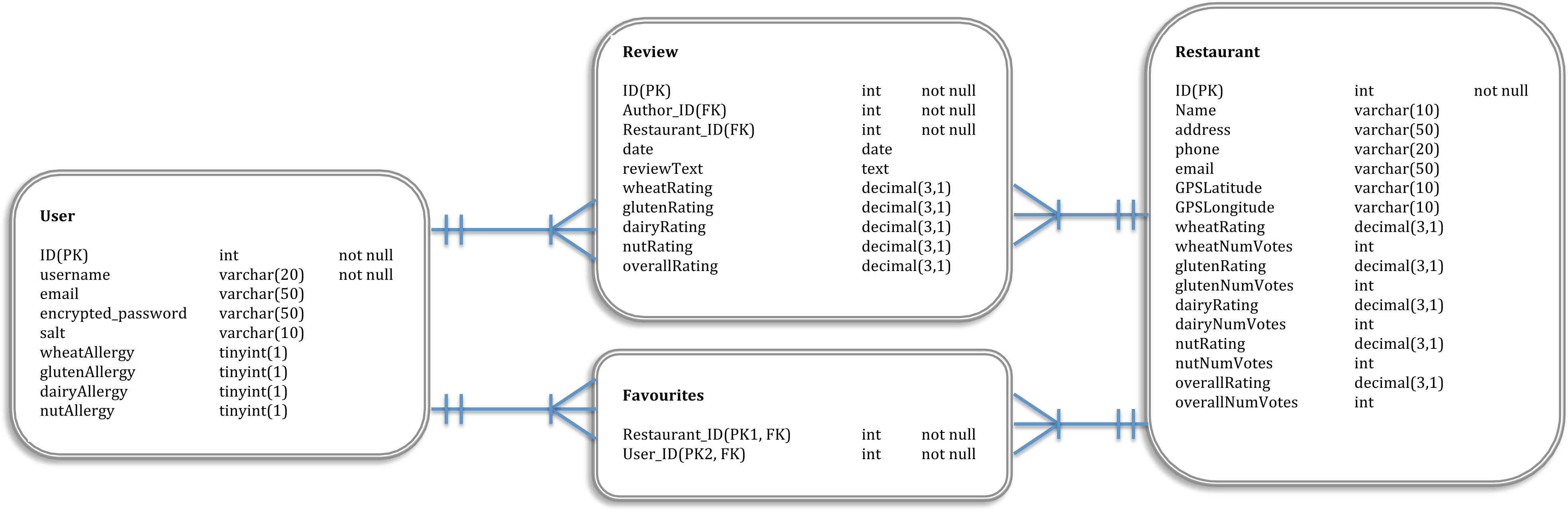
****

Figure 3‑4 - Entity Relationship Diagram for database

**User Table**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| FIELD | TYPE | SIZE(byte) | NULL | DEFAULT |
| ID (PK) | int | 4 | no | Auto-increment |
| username | varchar | 20 | No |  |
| email | varchar | 50 | no |  |
| Encrypted\_password | varchar | 50 | no |  |
| Salt | varchar | 10 | no |  |
| wheatAllergy | tinyint | 1 | yes |  |
| glutenAllergy | tinyint | 1 | yes |  |
| dairyAllergy | tinyint | 1 | yes |  |
| nutAllergy | tinyint | 1 | yes |  |

**Restaurant Table**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| FIELD | TYPE | SIZE(byte) | NULL | DEFAULT |
| ID (PK) | int | 4 | no | Auto-increment |
| name | varchar | 30 | No |  |
| address | text | <=65535 | yes |  |
| email | varchar | 50 | yes |  |
| GPSLatitude | varchar | 10 | no |  |
| GPSLongitude | varchar | 10 | no |  |
| wheatAllergy | decimal | 8 | yes |  |
| wheatNumVotes | int | 4 | yes |  |
| glutenAllergy | decimal | 8 | yes |  |
| glutenNumVotes | int | 4 | yes |  |
| dairyAllergy | decimal | 8 | yes |  |
| dairyNumVotes | int | 4 | yes |  |
| nutAllergy | decimal | 8 | yes |  |
| nutNumVotes | int | 4 | yes |  |
| overallRating | decimal | 8 | yes |  |
| overallNumVotes | int | 4 | yes |  |
| numFavourites | int | 4 | yes |  |

**Review Table**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| FIELD | TYPE | SIZE(byte) | NULL | DEFAULT |
| ID (PK) | int | 4 | no | Auto-increment |
| Author\_ID (FK) | int | 4 | no |  |
| Restaurant\_ID (FK) | int | 4 | no |  |
| date | date | 3 | no |  |
| reviewText | text | <= 65535 | yes |  |
| wheatRating | decimal | 8 | yes |  |
| glutenRating | decimal | 8 | yes |  |
| dairyRating | decimal | 8 | yes |  |
| nutRating | decimal | 8 | yes |  |
| overallRating | decimal | 8 | yes |  |

**Favourites Table**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| FIELD | TYPE | SIZE(byte) | NULL | DEFAULT |
| User\_ID (PK1, FK) | int | 4 | no |  |
| Restaurant\_ID (PK2, FK) | int | 4 | no |  |

## Local SQLite Design

**User Table**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| FIELD | TYPE | SIZE(byte) | NULL | DEFAULT |
| ID (PK) | int | 4 | no | Auto-increment |
| Unique\_ID | int | 4 | no |  |
| username | varchar | 20 | No |  |
| email | varchar | 50 | no |  |
| wheatAllergy | tinyint | 1 | yes |  |
| glutenAllergy | tinyint | 1 | yes |  |
| dairyAllergy | tinyint | 1 | yes |  |
| nutAllergy | tinyint | 1 | yes |  |

The local database stores one table that is used to check if a user is registered/logged in. It does not store the user’s password locally for security reasons. A connection to the web server database has to be made to validate the user against the password stored there. The primary key of the user in the web server database is stored in the local database in the field “Unique\_ID”. This way, any altering of the web server database (eg adding a review) can point to the correct user using the “Unique\_ID” stored locally. It minimizes the amount of calls to the web server needed, until the user logs out.

## User Interface Design

|  |  |
| --- | --- |
| Macintosh HD:Users:Maeve:Dropbox:Android Allergy App:infographics:screen grabs:main menu.tiff  Figure 3‑5 – Main Menu | Macintosh HD:Users:Maeve:Dropbox:Android Allergy App:infographics:screen grabs:register.tiff  Figure 3‑6 - Register Account |
| Macintosh HD:Users:Maeve:Dropbox:Android Allergy App:infographics:screen grabs:map view.tiff  Figure 3‑7 - Map View | Macintosh HD:Users:Maeve:Dropbox:Android Allergy App:infographics:screen grabs:list view.tiff  Figure 3‑8 - List View of restaurants |
| Macintosh HD:Users:Maeve:Dropbox:Android Allergy App:infographics:screen grabs:restaurant view.tiff  Figure 3‑9 - Restaurant Detail | Macintosh HD:Users:Maeve:Dropbox:Android Allergy App:infographics:screen grabs:review screen.tiff  Figure 3‑10 - Review Restaurant |

## List of Features

# Development & Implementation

## Details of each component within the project, problems encountered and resolved, challenges overcome or worked around.

## Identify key development components

# System Validation

## Testing: What testing was performed, why it was selected and what are the key use cases within the project.

# Project Plan

## Project Plan analysis and review of how it changed from the initial proposal including explanation of what changed and why, and suggestions on how to address this if the project was repeated.

# Conclusion

## Analysis of the projects key elements identify the key learning obtained from the project and recommendations and suggestions for how the work can be improved on continued into the future.

# Bibliography

## Images

Figure 2‑1 - Eclipse for Mac with Android Emulator 9

Figure 2‑2 - Samsung Galaxy Ace 9

Figure 2‑3 - Android RESTful api 10

Figure 2‑4 - Signup page for hosting on x10hosting.com 12

Figure 2‑5 - cPanel page to manage MySQL databases 12

Figure 2‑6 - CyberDuck manages file transfer 14

Figure 2‑7 - Textwrangler text editor 15

Figure 2‑8 - PHP script to connect to Database 16

Figure 2‑9 - Android Architecture Diagram 17

Figure 2‑10 - Market Share of Android Versions 2013 17

Figure 2‑11 - Getting Google Maps API key for Android 19

Figure 2‑12 - Inserting Google API key into map layout xml 19

Figure 3‑1 - Intial System Architecture Design 22

Figure 3‑2 - Final System Architecture Design 23

Figure 3‑3 - Use Case for user 25

Figure 3‑4 - Entity Relationship Diagram for database 27

Figure 3‑5 – Main Menu 32

Figure 3‑6 - Register Account 32

Figure 3‑7 - Map View 32

Figure 3‑8 - List View of restaurants 32

Figure 3‑9 - Restaurant Detail 33

Figure 3‑10 - Review Restaurant 33

# 

## Image References

Figure 2‑2 - Samsung Galaxy Ace

Retrieved March 19, 2013, from GSM Arena website: <http://www.gsmarena.com/samsung_galaxy_ace_s5830-pictures-3724.php>

Figure 2‑3 – Android RESTful api

Retrieved April 15, 2013, from Android Hive website: http://www.androidhive.info/2012/01/android-login-and-registration-with-php-mysql-and-sqlite/

Figure 2‑8 - Android Architecture Diagram

Retrieved March 19, 2013, from Android Developer website: <http://developer.android.com/about/versions/index.html>

Figure 2‑9 - Market Share of Android Versions 2013

Retrieved March 19, 2013, from GSM Arena website: <http://www.gsmarena.com/state_of_android_jan_2013_jb_market_share_still_under_15-news-5489.php>

Figure 2‑10 - Getting Google Maps API key for Android

Retrieved March 19, 2013, from Google Developer website:

https://developers.google.com/maps/documentation/android/index

# Appendix