Bachelor's thesis (TUAS)

Degree Programme in Information Technology

Specialization: Android App Development

2014

Raj kumar Singh

ANDROID ALERT APP (SHOP SALE)

TURUN AMMATTIKORKEAKOULU
TURKU UNIVERSITY OF APPLIED SCIENCES

BACHELOR´S THESIS | ABSTRACT
TURKU UNIVERSITY OF APPLIED SCIENCES

Degree Programme in Information Technology

Summer 2014| 27

Instructor: Almurani Balsam

RAJ KUMAR SINGH

ANDROID ALERT APP (SHOP SALE)

With evolving technologies, number of business organizations is conducting business via eCommerce. Mobile commerce is effective and efficient among one of them. Android is invading the enterprise and mobile commerce is moving to smartphones. It uses World Wide Web for exchanging data to facilitate the target customer with all its contents and the details.

The objective of this thesis was to build an Alert generating eCommerce app. As a result, Shop Sale was made. It is an Android application that communicates with the web services and use Google maps to obtain the specific location that run in the background and return the content of the sale page from the commercial website situated around/nearby the app user so that user can view the sales going on in that particular shop without entering the shop. Alerts appear as pop-up in the notification panel of display when the user is around the shop.

Shop Sale is able to accommodate wide number of enterprise. This app is productive for customers that helps stay in touch with eCommerce in real time. It is easy to access and is full of function and services.

KEYWORDS:

Android, Android SDK, AVD Manager, JAVA, MySQL, PHP, XML, SQLite

CONTENTS

LIST OF ABBREVIATIONS (OR) SYMBOLS 5

1 INTRODUCTION 6

2 DESIGN AND REQUIREMENTS 7

2.1 Application Target User 7

2.2 Application Requirement 7

2.3 Device Requirements 7

3 USED TOOLS AND TECHNIQUES 8

3.1 Development Environment 8

3.1.1 Eclipse 8

3.1.2 Eclipse ADT 8

3.1.3 Android SDK 8

3.1.4 AVD Manager 9

3.1.5 Google API key 10

3.2 Programming Environment 11

3.2.1 Android 11

3.2.2 JAVA 11

3.2.3 PHP 11

3.3 Database 12

3.3.1 MySQL 12

3.3.2 SQLite 13

3.4 UI design 13

3.4.1 Map Window 13

3.4.2 Point of Interest 14

3.4.3 Item List 14

3.5 Application Fundamentals 14

4 GRAPHICAL USER INTERFACE DESIGN 15

5 IMPLEMENTATION 16

5.1 Interface Components 16

5.1.1 Maps 16

5.1.2 Point of Interest (POI) 17

5.1.3 Map Database 18

5.1.4 Alert 18

5.2 Implementing the Interface 19

5.2.1 MainActivity.java 19
5.2.2 activity_main.xml 20
5.2.3 Androidmanifest.xml 21
5.3 JSON 22
5.4 Web Crawler 22
6 TEST AND RESULTS 24
6.1 JSON Functional Testing 24
6.2 Run-Time Problem 24
6.3 Security testing 24
7 FUTURE IMPROVEMENT 25
8 CONCLUSION 25
REFERENCES 26

FIGURES

Figure 1. Android SDK Manager 9
Figure 2. Alert Window..... 10
Figure 3. Google API key..... 11
Figure 4. MySQL Database 13
Figure 5: String example..... 15
Figure 6. Google Map 17
Figure 7. Point of Interest..... 18
Figure 8. Alert notification 19
Figure 9. MainActivity.java 20
Figure 10. activity_main.xml 21
Figure 11. AndroidManifest.xml 22
Figure 12. Web Crawler 23

APPENDICES

Appendix 1. Extract from MainActivity.java

LIST OF ABBREVIATIONS (OR) SYMBOLS

API	Application Programming Interface.
SDK	Software Development Kit
SQL	Structured Query Language
XML	Extensible Markup Language
OS	Operating System
Android OS	Developed by Google OS for mobile development
AVD	Android virtual device
ADT	Android developer tool
GPS	Global positioning system
JAVA	Software platform developed by Sun Microsystems, object oriented programming
IDE	Integrated Development Environment
XML	Extensible Markup Language
APP	Application
UI	User Interface
VM	Virtual machine
JSON	JavaScript Object Notation, a simple data format that can be used in many programming languages.
POI	Point of Interest

1 INTRODUCTION

The world is marching forward in the field of technology. There have been significant developments in the past few years which have made the life easy and convenient for people. The Internet and smartphones are few examples of such developments. People use smartphones as their personal digital assistants now days. With the enhancement of wide technology, a smartphone, when connected to internet, delivers a heap of data to the user. At the same time, electronic commerce has rapidly progressed and is considered as one of the most perspective field of business development. Its services are widely available to customers online. Data is an asset to any business in e-commerce. E-enterprises have a collection of processed information related to their subject. It is easy to retrieve and use the data from database using the Internet on Smartphone for any e-enterprise available online.

Considering the fact of evolving mobile technology, this thesis is mainly focused on an Android application that gathers almost all the product of sale pages mentioned in most of the shopping malls in Finland. The alert is activated according to the user choice where the sales of respective shopping mall is shown as an alert message whenever the user reaches near his/her point of interest shopping center. In order to accomplish our goal, we used two main platforms: firstly, Eclipse with Android SDK and Android AVD Manager for programming and secondly, MYSQL and SQLite for the database.

In order to achieve our goal, three main windows were created using android xml: Firstly, a Maps window where the user points the location of shops and the thus the latitude and longitude of the place is retrieved; Secondly, a Shops list window where the latitude and longitude retrieved from the Maps window is used for saving with its respective shops name; Finally, a Popup window that comes into an action only when the users passes by the shop location and thus display the content of the sale page of that shops. After the GUI design was

completed, a database for the shops and their product was created in MYSQL and for storing the list of shops we used mobile internal database, i.e., SQLite.

2 DESIGN AND REQUIREMENTS

2.1 Application Target User

The target users of this app are customer in particular malls. The app is designed so that the customer will easily, quickly and efficiently find the shop sale product or the needed item. The application uses maps, running in the background to locate the position of the user and displays the content. The app retrieves the content directly from the webpage or from its own database which is on the server.

2.2 Application Requirement

Our application is mostly dealing with a huge amount of data so it needs a database for it and we have used this database in it:

- MYSQL: This is an extra database needed for storing the shops' products in it.
- SQLite: This is an inbuilt database in Android needed to store the list of shops and their location.

2.3 Device Requirements

The device requirements for using the app are as follows:

- Android 2.2 Froyo operating system
- Built-in GPS
- Access to internet via a 3G/4G data interface

3 USED TOOLS AND TECHNIQUES

This section discusses the theoretical part used in the project. The purpose is to know and become familiar with these tools. The application creation was performed on a Windows machine where Vista was used as Operating system. Samsung Galaxy S4 was used as a testing device as well as Android Virtual Device. A real device is preferred since the compilation and running time consumed is less.

3.1 Development Environment

3.1.1 Eclipse

Eclipse is a Java-based open source platform that allows software developers to build an integrated development environment (IDE). Eclipse is used to develop applications in multi languages like C, C++, Cobol, Java, Python, Perl, PHP, etc. It provides features, such as code editor, syntax highlight, and debugging.

3.1.2 Eclipse ADT

ADT is a plug-in for Eclipse IDE which provides a powerful and integrated environment for building an Android app. It helps in quick setup of Android projects, creating user interfaces, adding packages, debugging app, exporting and importing .apk files. (ADT Plugin | Android Developers)

3.1.3 Android SDK

The Android SDK (Software Development Kit) consists of typically, software development tools, which is used to develop a particular software package, software platform, hardware platform, or system.

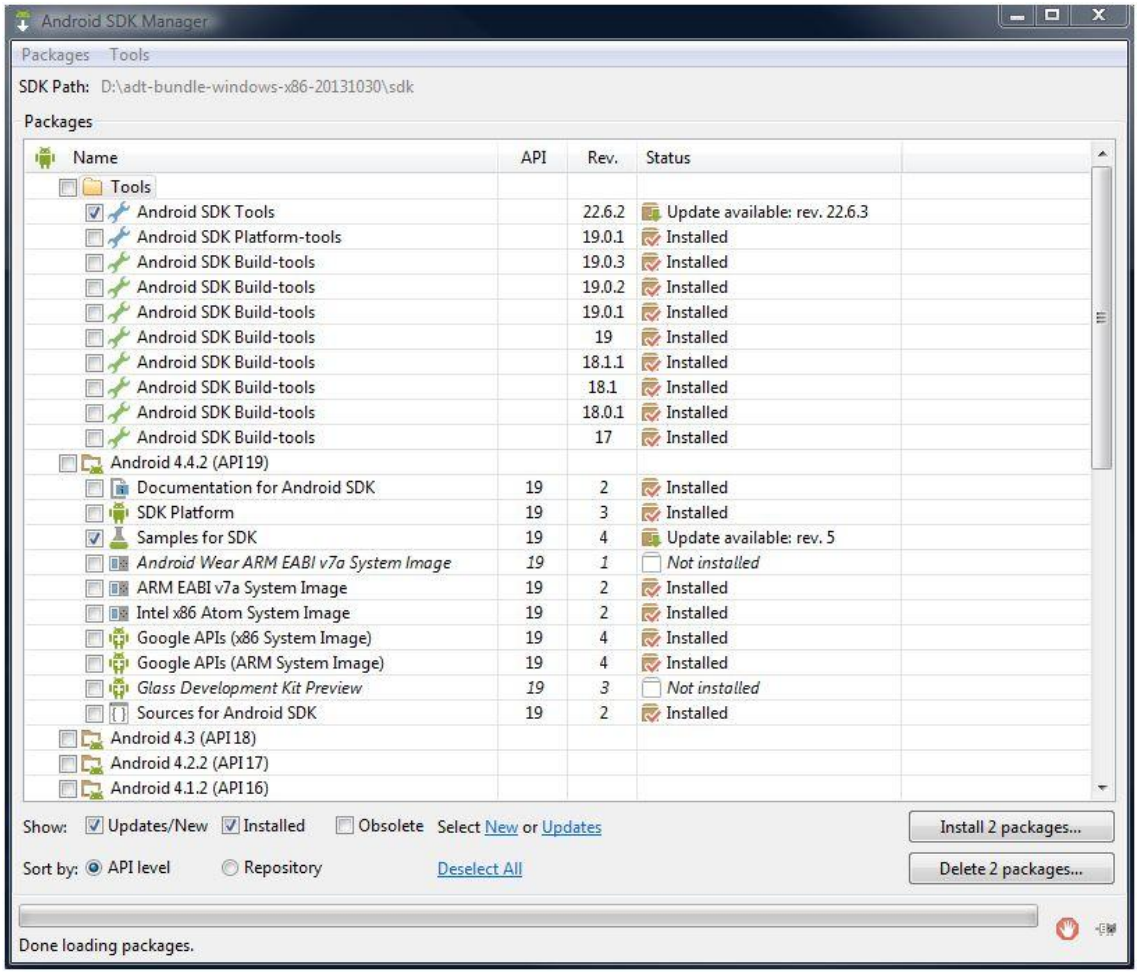

Figure 1. Android SDK Manager

3.1.4 AVD Manager

The Android Virtual Device Manager provides a graphical UI where Android Virtual devices can be created and configured. It provides virtual device emulator environment where Android application behavior is tested by installing and running it. The device model along with the specification is chosen and

launched.

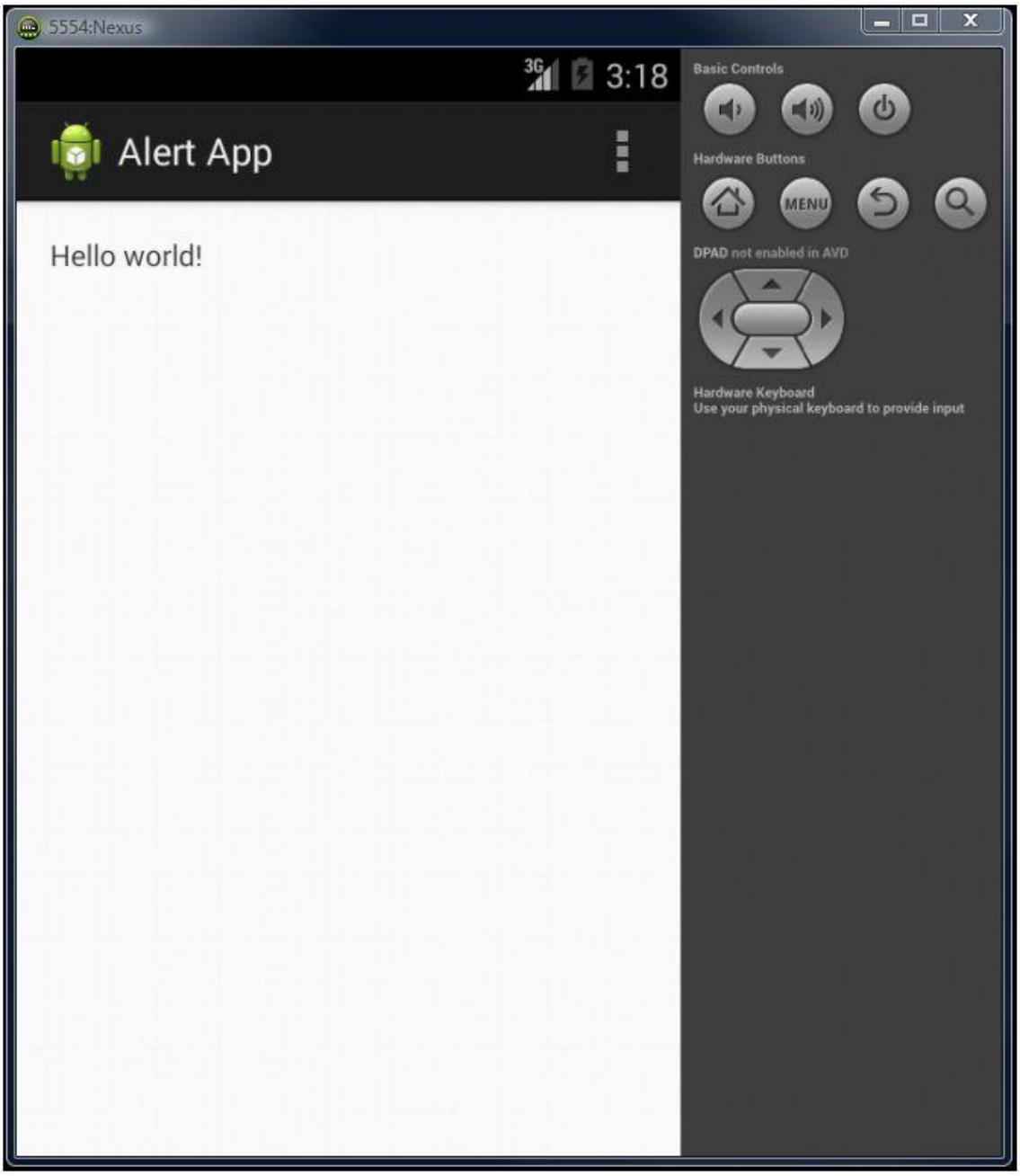

Figure 2. Alert Window

3.1.5 Google API key

Interactions among the software components are specified by the API (application programming interface) key. It helps the developer to integrate Google map to the app. It helps to uniquely identifying the user and the application to Google and also enables the term of services enforcement for

Google map. It is a free service. In order to obtain API key, the user has to have a Google API project. A Gmail account is also needed. The user can enter the console via https://console.developers.google.com . Google needs user credentials because Google needs to know what the map is being used for. Google has some limiting factor for its API keys. An application using an API key can request 1000 keys in 24 hours. Furthermore, the API key request can be increased to 100,000 requests in 24 hours free of charge but the user needs to verify them via credit card. A map without API keys displays blank grids on the empty screen. (Google Maps - Wikipedia, the free encyclopedia.)

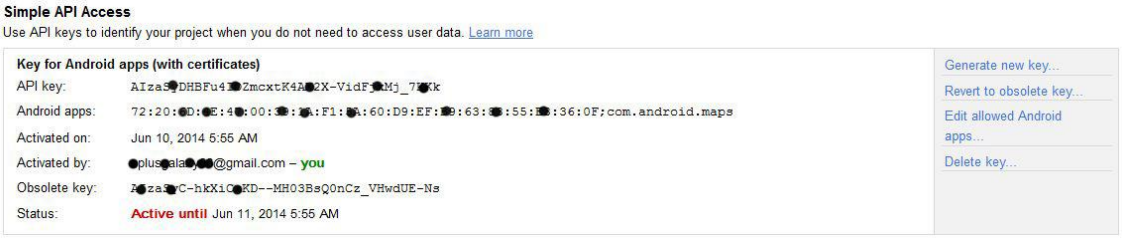

Figure 3. Google API key

3.2 Programming Environment

3.2.1 Android

Android is a Linux-based mobile operating system developed by Google. It is primarily designed for tablet computers and smartphones. Android, being an open source platform is used by a variety of several other electronic medias/devices like TVs, cameras, smart watches, headphones, etc.

3.2.2 JAVA

It is developed by Sun Microsystems and it is platform-independent and a fully object-oriented programming language.

3.2.3 PHP

PHP is open source scripting language. It is designed for web programming and development. It is executed on servers. A WAMP server is used in this project.

PHP is used to run the SQL script and make connections to the server. All major systems support it.

3.3 Database

A collection of data is called database. In this particular app, both the MySQL and SQLite database are used. MySQL is installed along with the WAMP package. WAMP is an acronym for Windows, Apache, MySQL and PHP that define Windows as OS, Apache as web server, MySQL as database and PHP, Perl or Python as scripting language. It is a small server web development environment that runs on Window OS. The files hosted on this server can be accessed by typing http://localhost (127.0.0.1) in the address bar of web browser.

3.3.1 MySQL

MYSQL is database management system. It is a program that lets users store and retrieve the data efficiently. It is one of the popular open source databases. It is a server-side data storage database used for developing web applications. (MySQL :: MySQL 3.23) In this app, it is used to store the data crawled from various shops. Table named “product_table” is created under the “db_hansa” database.

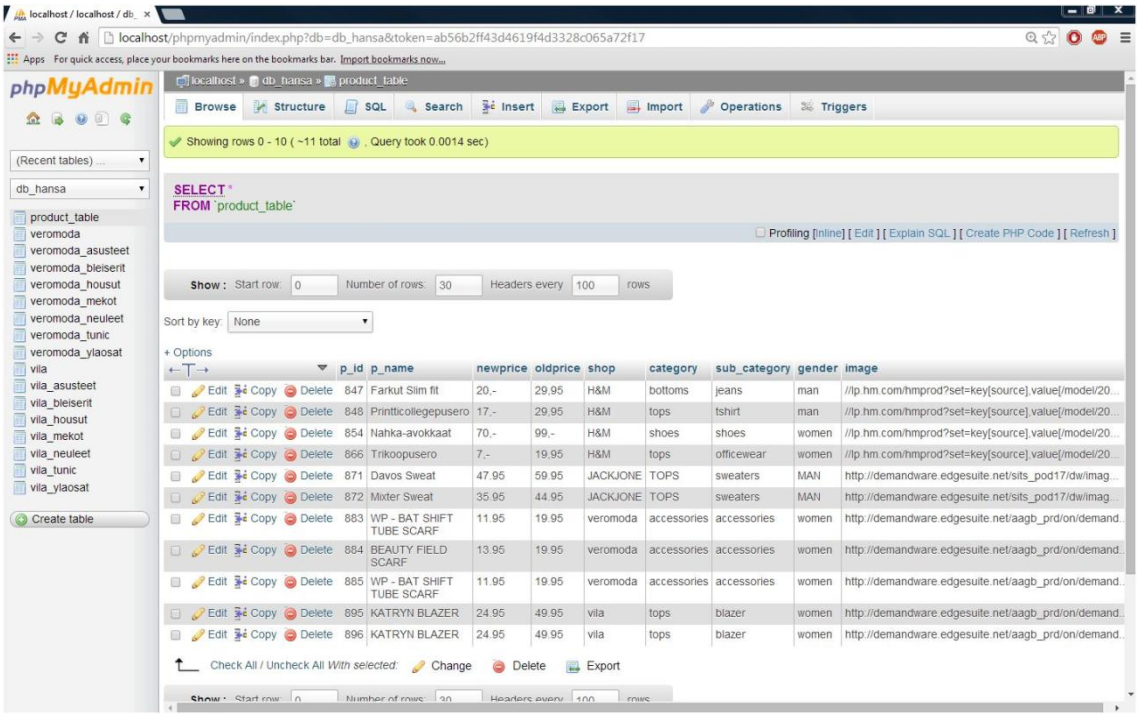

Figure 4. MySQL Database

3.3.2 SQLite

Client side data storage databases are mostly embedded in the SQL database engine. SQLite is a zero.configuration and serverless database engine. It does not have a separate server process. Reading and writing data is directly done on the disk file. It is a compact library ranging the library size from 300KiB – 500KiB. It is used in cell phones, MP3 players etc. In this app, SQLite stores the location information provided by the user.

3.4 UI design

3.4.1 Map Window

The Map window contains the Google map for the App. It is created by getting Google API keys from Google.

3.4.2 Point of Interest

It is the point (location, place) on the map that the user wishes to save with name and the coordinates. It is used for generating the popup when the user is beside/nearby that location. The popup shows if any shop saved on the database is available. The name of the places might be one or many.

3.4.3 Item List

Item list displays all the items (goods) present on the database. Items from the sale page are stored on the database and they are displayed as list item when accessed.

3.5 Application Fundamentals

The Eclipse platform provides hundreds of plug-in and different .jar files that add more functionality. Basically, a jar file contains a class file as well as supports a config file needed by the applications. Some of the jar files are Google play service.jar, Google play service lib.jar, Android support-v4.jar, android.jar.

Layout is basically a user interface for an activity that defines the visual structure. Located in the res/layout folder, a user interface is an .XML file. It can either be declared in .XML of the user interface or can be instantiated during layout. There are basically of two types of layout, i.e., linear layout and relative layout. Linear layout organizes its children into single horizontal or vertical rows whereas relative layout enables to specify children locations relative to each other. In addition, table layout and web view are also used for layout. The most common values within layout are FILL_PARENT and WRAP CONTENT. FILL_PARENT helps to take the full width whereas WRAP_CONTENT takes the full height.

MainActivity.java helps launch the application. It is a source file located in the src folder. The App user interface is defined by the activity_main.xml file located in the layout folder. AndroidManifest.xml, located in the res/layout directory

describes and defines the fundamental characteristic of the application and each of its components. Strings.xml contains all the texts, labels, buttons, spinner, etc. and is located in the res/values folder. The default string looks as follows:

```
1 <?xml version="1.0" encoding="utf-8"?>
2 <resources>
3
4     <string name="app_name">Alert App</string>
5     <string name="hello_world">Hello world!</string>
6     <string name="action_settings">Settings</string>
7
8 </resources>
9
```

Figure 5: String example

4 GRAPHICAL USER INTERFACE DESIGN

While designing the GUI, user ease for accessing was considered. It is made so that the users will save the point of interest when they visit different places. It is to be done on the map interface. They can click the get button for their current location and save the place and shop name. This is on POI.xml interface which is point of interest. Once the location is saved and the database on the server contains any shop nearby that place within a 20-meter radius, they will receive a pop-up in the notification bar. A single alert is created for every shop. When the popup is opened, it displays the entire list of sale products contained in that particular shop. This is an item list interface. In addition, the user can then categorize the product according to gender (M/F) and sub-category.

5 IMPLEMENTATION

5.1 Interface Components

5.1.1 Maps

Google Maps is a US-based Google produced a route planner service. It provides Viewing Street and satellite maps, as well as the opportunity to travel the route planning and search for local businesses. A Google map can be integrated to the activity with the help of Google map Android API v2. Moreover, it needs an API key from the Google console and adding some specific settings in the AndroidManifest.XML. Google offers an external Android library map that includes a com.google.android.maps package. The class includes features like downloading map title, rendering and buffering, zoom control, scale, rotate, etc. The map offers features, like adding different types of map, i.e., normal, satellite, hybrid, terrain. The map can be customized in various ways. Markers can be added and drawings can be done. This application uses Google maps running in the background. (Google Maps - Wikipedia, the free encyclopedia.)

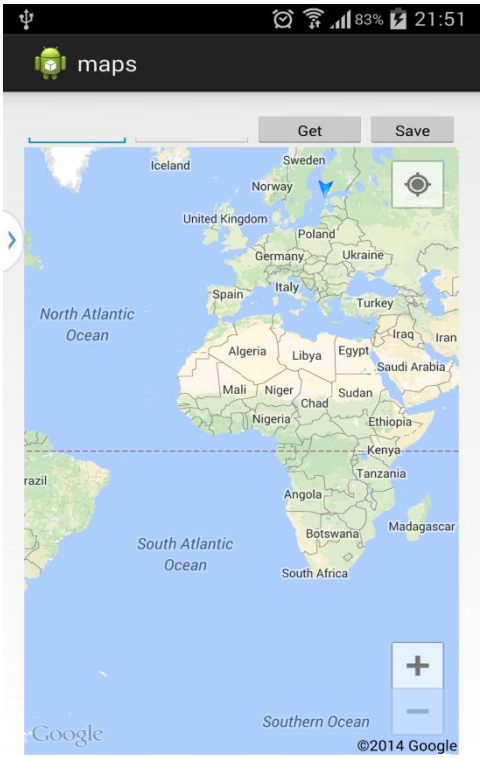

Figure 6. Google Map

5.1.2 Point of Interest (POI)

Point of Interest (POI) is the point or place to be saved on the map by the app user. The user can either enter the co-ordinates or tap the get button to get the current location. The user should name the shop and the place for getting popup if the shop is available around. The data is saved on the device

database.

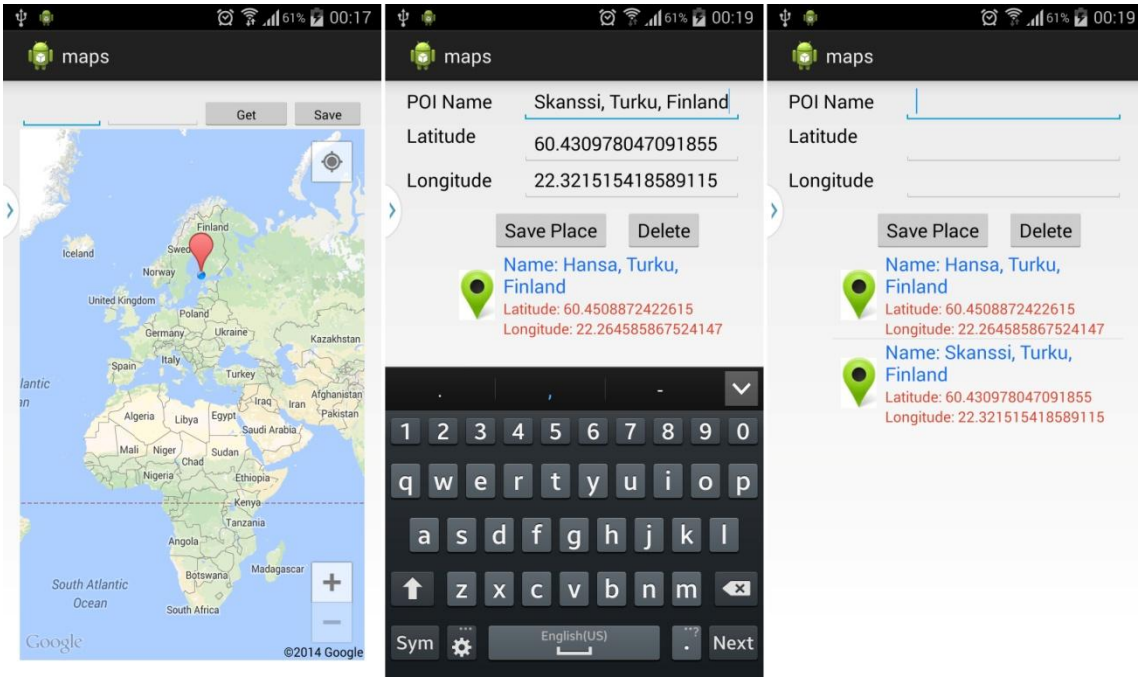

Figure 7. Point of Interest

5.1.3 Map Database

The entered data, i.e., the name of the shop and their location is saved on the device. It is easy to track the movement of user and inform them when they approach the saved location. It is saved on the SQLite database and consumes a small amount of memory in device.

5.1.4 Alert

When the user approaches the location, the alert is created in the notification panel. Once the user opens it, the entire sale item is displayed. In addition, the option of categorizing the product is provided.

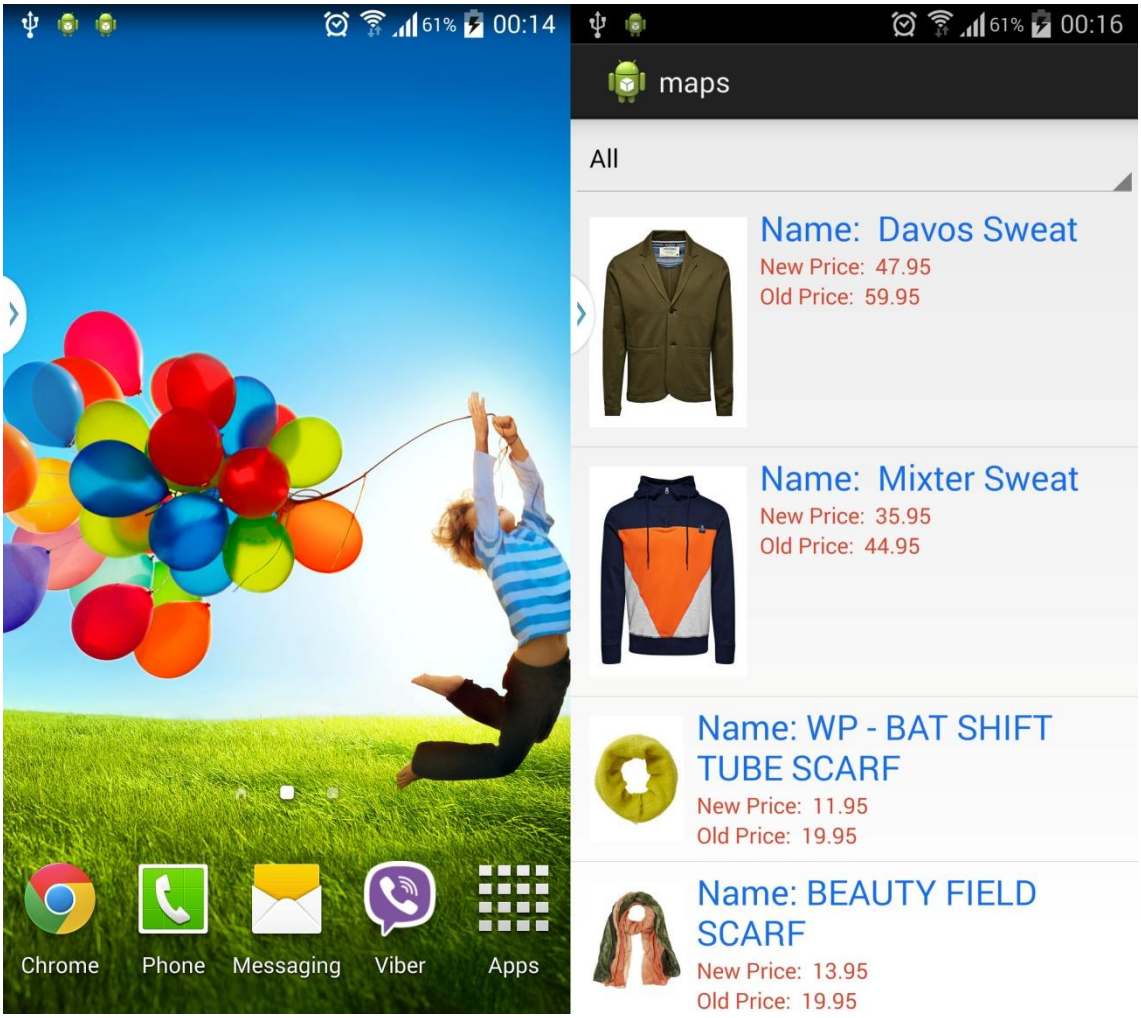

Figure 8. Alert notification

5.2 Implementing the Interface

5.2.1 MainActivity.java

This is the source file of the project. It is located in the scr folder. Basically, it contains the package name and all the imports required by the program. It might have several other public and private methods, constructors and various interfaces. The applications default code page for MainActivtiy.java looks like

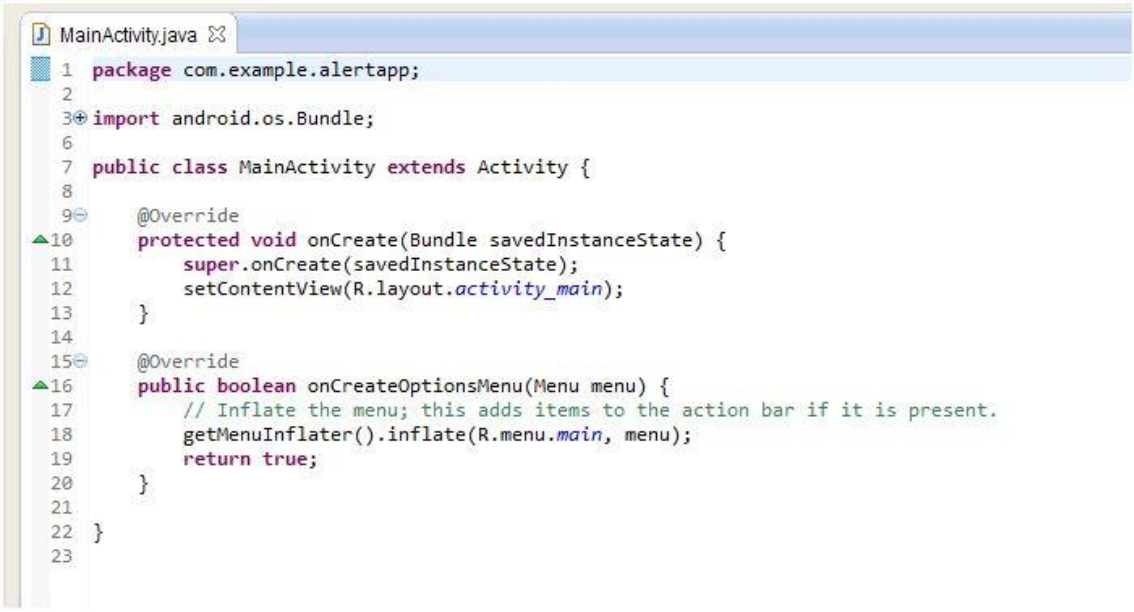

Figure 9. MainActivity.java

5.2.2 activity_main.xml

activity_main is a graphical layout editor for designing Android app. It is stored in res/layout directory with the extension of .xml. The visual container of the User Interface contains a single root element. It can be either View or ViewGroup like ListView, GridView, LinearLayout. A palette, like Form Widgets, Text Field, Layouts, Image and Media, Advanced and other can be found here

for fine tuning of the User Interface.

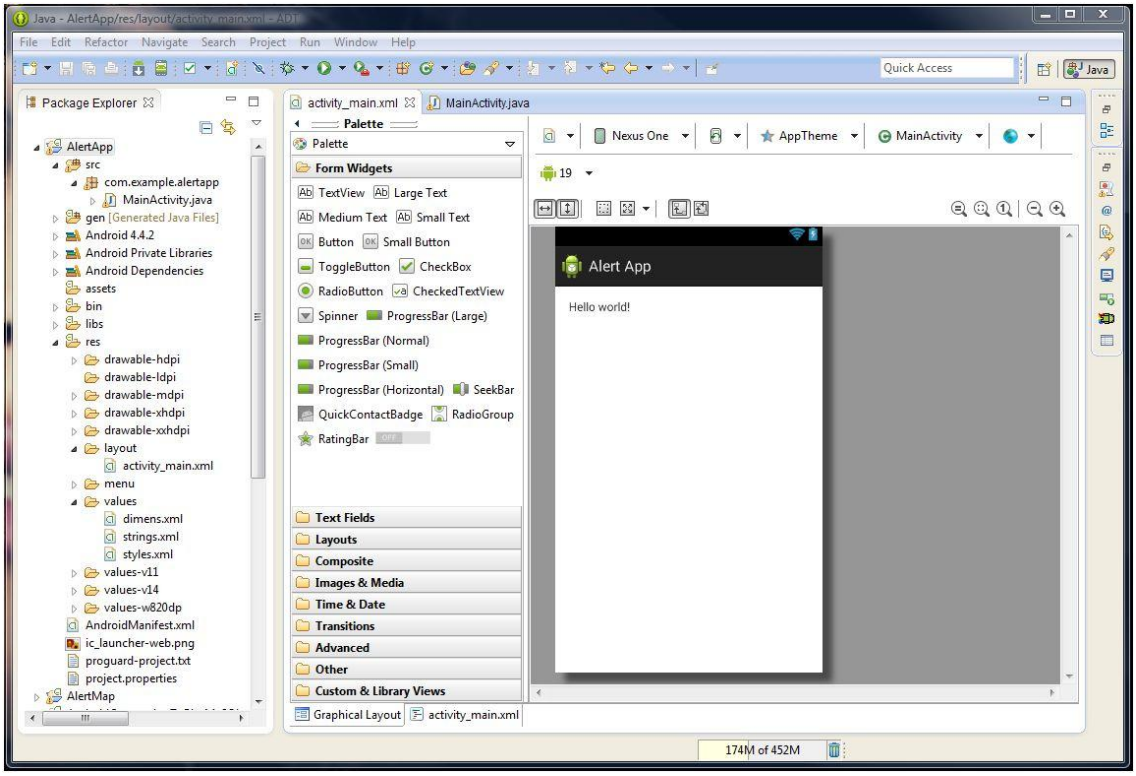

Figure 10. activity_main.xml

5.2.3 AndroidManifest.xml

Every Android application has an AndroidManifest.XML file that explicitly describes the content of the application. It contains essential information, such as the name of java package for the app, describes the components of app, determines host application process, declares the permissions required by the app, and declares the minimum level of Android API required by the app. It works as an interface between the application and the Android OS. It organizes the app in a well-defined structure for loading and executing the app by Android OS in the managed environment. Components developed as a part of application are declared here such as min version of Android required and other hardware configuration requirements. The declarations to the main components of app like Activity, Services, Content Provider and Broadcast Receiver are made here in AndroidManifest.xml file. (App Manifest | Android Developers.) The Default AndroidManifest.xml file looks like this:

MainActivity.java | activity_main.xml | AlertApp Manifest

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.alertapp"
    android:versionCode="1"
    android:versionName="1.0" >

    <uses-sdk
        android:minSdkVersion="8"
        android:targetSdkVersion="18" />
    <uses-permission android:name="android.permission.ACCESS_FINE_LOCATION"/>
    <uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION"/>
    <uses-permission android:name="android.permission.INTERNET"/>
    <uses-permission android:name="android.permission.ACCESS_NETWORK_STATE"/>
    <uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE"/>
    <uses-permission />

    <application
        android:allowBackup="true"
        android:icon="@drawable/ic_launcher"
        android:label="@string/app_name"
        android:theme="@style/AppTheme" >
        <activity
            android:name="com.example.alertapp.MainActivity"
            android:label="@string/app_name" >
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>

</manifest>

```

Figure 11. AndroidManifest.xml

5.3 JSON

JSON is JavaScript Object Notation. It is a text-based open standard format that interchanges human readable data i.e. stores and exchanging data. It stores data in organized way that helps in easy access. The outputs are logically arranged human readable data. (JSON. 2014)

5.4 Web Crawler

The Web crawler, also known as web spider, bots, robots, ants is an automated script which browses the WWW in an automated manner for web indexing. It visits the page in the Web, fetches specific information, downloads it, creates an index and stores it in the database in a systematic manner. It helps to provide the user with more relevant result in faster time. Page validation, update

notification, visualization, structural analysis, mirroring, etc are some other useful purposes of Crawler. (Web crawler. 2014)

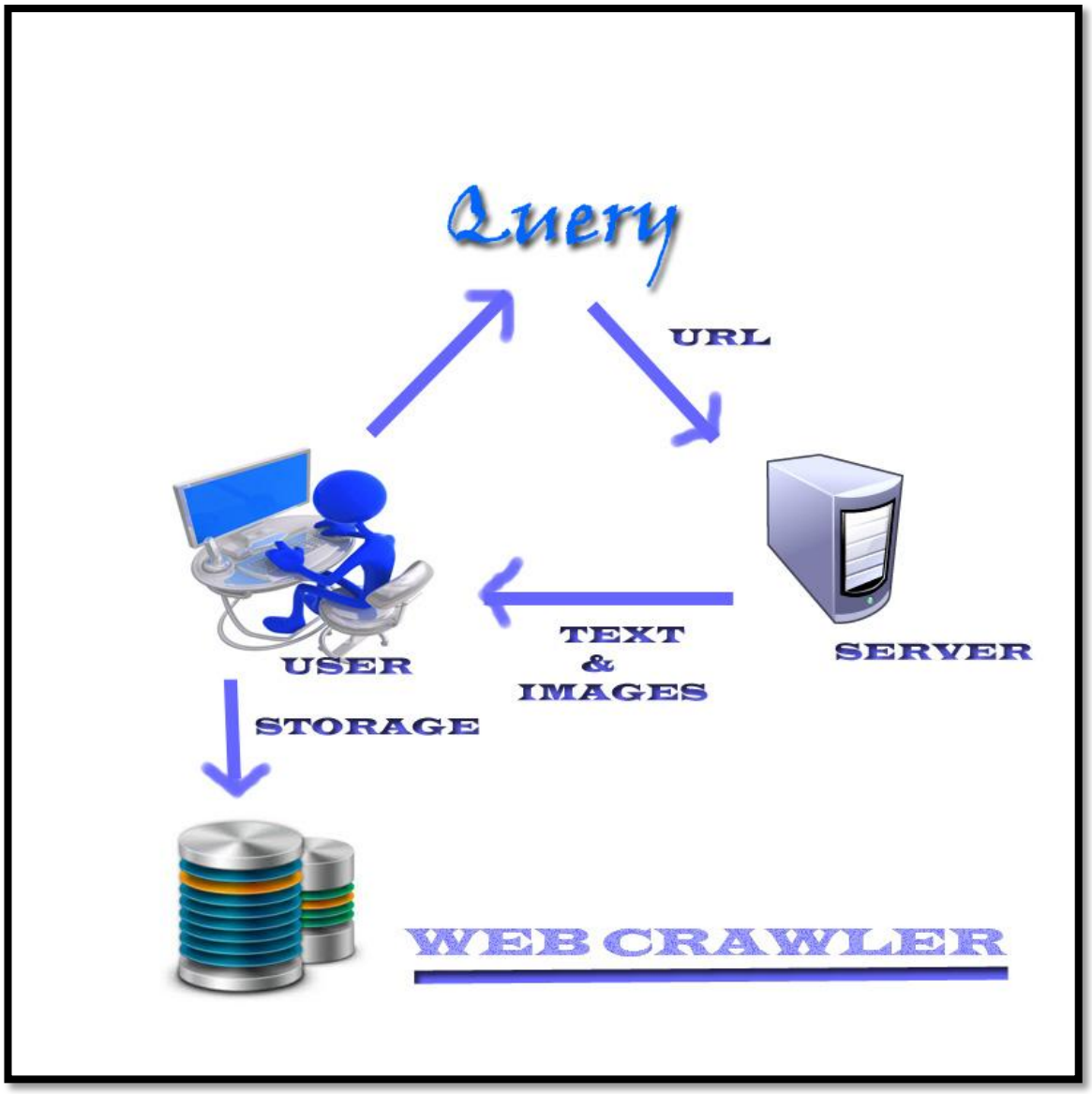

Figure 12. Web Crawler

The query is sent to server. The query is like the contents page of book which tells the exact location of the contents. The query retrieves the document from the server and searches the result. The result is then returned to the user. The user saves the result and then again runs the query.

6 TEST AND RESULTS

The test was carried out several times on Android phones based on the requirements. The app was installed on several Android devices. The users were the author and his friends. The test was successful and satisfactory. However, further improvements are discussed in the Future Improvement section below.

6.1 JSON Functional Testing

The MYSQL database field's property is fixed. So in order for JSON to function accurately, JSON needs to filter the data received from the web crawler according to the property sets in the product table. In our application, it works fine for already added pages as they are filtered. However, for new web pages it needs to be done manually again and again in the code as since they differ from design to design.

6.2 Run-Time Problem

Run-time problems are the errors that prevented the program to work correctly. These might be either due to a software or hardware problem. Sometimes the application was unable to load the runtime library files as well as crashing the app. The app was tested on a Samsung Galaxy S4 mobile phone. An Android Virtual device was also used. Using the real device was much more efficient and less time-consuming then using the virtual device during the setup and installation of the application.

6.3 Security testing

The external database is used to store the data retrieved from the stores webpage. A personal computer functioned as the local server with WAMP installed on it. The database id was made fully secure so that no external party could access it.

7 FUTURE IMPROVEMENT

After the necessary tests, we were able to fix some of the problems, but most of the following problems have been left for future improvements as explained below:

- Since we are dealing only with lists of sale products which can be handled by SQLite, we could remove the MYSQL database to make the application even faster.
- At the moment, we can save as many shops names without having their database in our server so it should be fixed by letting the application add only those shops in point of interest whose database are there.
- Saving a shop's latitude and longitude can be minimized to its address as it is not an easy task to remember latitude and longitude compared to an address.
- More filtering options can be added to the Alert window such as filtering by price, gender, etc.

8 CONCLUSION

The outcome of the thesis is a fully functional alert app based on the Android operating system. It is designed for customers seeking for the sale items. It allows the user to see the entire on-going sale in a real environment in particular shop in a particular place. The application works on Android smartphone and tablets. Initially, the app was designed and implemented in the Turku City region. The thesis achieved the desired result.

Working on this project was a good experience for me while programming in Android. I got to know the various usages of Android Software Development Kit on the Eclipse Development Environment. Dealing with a number of different app techniques was instructive, challenging and meaningful to me.

REFERENCES

Google Maps - Wikipedia, the free encyclopedia. 2014. *Google Maps - Wikipedia, the free encyclopedia*. [ONLINE] Available at:http://en.wikipedia.org/wiki/Google_Maps. [Accessed 13 May 2014].

App Manifest | Android Developers. 2014. *App Manifest | Android Developers*. [ONLINE] Available at:http://developer.android.com/guide/topics/manifest/manifest-intro.html. [Accessed 26 May 2014].

ADT Plugin | Android Developers. 2014. *ADT Plugin | Android Developers*. [ONLINE] Available at: http://developer.android.com/tools/sdk/eclipse-adt.html. [Accessed 14 June 2014].

Web crawler. 2014. *Web crawler*. [ONLINE] Available at:http://www.sciencedaily.com/articles/w/web_crawler.htm. [Accessed 14 June 2014].

MySQL :: MySQL 3.23, 4.0, 4.1 Reference Manual :: 1.3.1 What is MySQL?. 2014. *MySQL :: MySQL 3.23, 4.0, 4.1 Reference Manual :: 1.3.1 What is MySQL?*. [ONLINE] Available at: http://dev.mysql.com/doc/refman/4.1/en/what-is-mysql.html. [Accessed 15 June 2014].

JSON. 2014. *JSON*. [ONLINE] Available at: http://json.org/. [Accessed 15 June 2014].

Appendix

Extract from MainActivity.java

```
public class MainActivity extends FragmentActivity implements
                        OnMapClickListener, OnMarkerDragListener {

.

.// codes in-between

.

.

public void onMarkerDragEnd(Marker marker) {
            // TODO Auto-generated method stub
            LatLng dragPosition = marker.getPosition();
            double dragLat = dragPosition.latitude;
            double dragLong = dragPosition.longitude;
            Intent proximityIntent = new
Intent("com.android.activity.proximity");
                        pendingIntent =
PendingIntent.getActivity(getBaseContext(), 0,
                                                proximityIntent,
Intent.FLAG_ACTIVITY_NEW_TASK);
                        locationManager.addProximityAlert(dragLat, dragLong,
20, -1,
                                                pendingIntent);

                        Toast.makeText(getApplicationContext(), "Marker
Dragged..!",
                                                Toast.LENGTH_LONG).show();
            }

.

.

.

.} (end of the code)
```