**MEDICAL ASSISTANCE SERVICE**



**SYSTEM**

**“**A Healthcare Locator System Using ANDROID**”**

***Submitted by***

**KRISHNAN.D 412511205039**

**SAIRAJ.M 412511205074**

**ADITYA.G 412511205007**

**A PROJECT REPORT**

*Submitted to the*

**FACULTY OF INFORMATION TECHNOLOGY**

*In partial fulfillment for the award of the degree*

*Of*

**BACHELOR OF TECHNOLOGY**

**IN**

**INFORMATION TECHNOLOGY**

**DEPARTMENT OF INFORMATION TECHNOLOGY**

**SRI SAIRAM ENGINEERING COLLEGE, CHENNAI**

**ANNA UNIVERSITY :: CHENNAI - 600 025**

**APRIL 2015**

**ANNA UNIVERSITY::CHENNAI 600 025**

**BONAFIDE CERTIFICATE**

Certified that this project report “**MEDICAL ASSISTANCE SERVICE SYSTEM**” is the bonafide work of

**KRISHNAN.D 412511205039**

**SAIRAJ.M 412511205074**

**ADITYA.G 412511205007**

who carried out the project work under my supervision.

**HEAD OF THE DEPARTMENT INTERNAL GUIDE**

**Dr. T.SHEELA, B.E., M.S., Ph.D., Ms. P. PADMA, M.E.,**

**DEAN [NETWORK]** **SUPERVISOR**

**PROFESSOR AND HEAD ASSISTANT PROFESSOR**

Department of Information Technology Department of Information Technology

Sri Sairam Engineering College Sri Sairam Engineering College

Sai Leo Nagar, West Tambaram, Sai Leo Nagar, West Tambaram,

Chennai- 600 044. Chennai- 600 044.

Submitted for University Project Examination held on ……………. at Sri Sairam Engineering College, Chennai-600 044.

**INTERNAL EXAMINER EXTERNAL EXAMINER**

**ACKNOWLEDGEMENT**

We express our deep sense of gratitude to our beloved chairman **Thiru. MJF. Ln. LEO MUTHU** for the help and advice he has shared upon us.

We express our gratitude to our CEO and Managing Trustee **Mr. J. SAI PRAKASH LEO MUTHU** and our trustee **Ms. J. SHARMILA RAJAA** for their constant encouragement in completing this project.

We express our solemn thanks to our Principal **Dr.C.V.JAYAKUMAR** for having given us spontaneous and whole hearted encouragement for completing this project.

We are indebted to our Head of the Department **Dr.T. SHEELA** for her constant support during the entire course of this project work.

We express our gratitude and sincere thanks to our guide **Ms.J.JULIE**  for her constant support and encouragement for successful completion of this project.

Our sincere thanks to our project coordinator **Ms.S.ANITHA JEBAMANI** for her kind support in bringing out this project successfully.

Finally, we thank all the staff members of the **Department of Information Technology** and all others who contributed directly or indirectly for the successful completion of our project.

We thank our Almighty for His blessings to make this project a success.

**ABSTRACT**

The quality of healthcare in India is not uniform. The urban areas have adequate quality that is comparable with western standards, but in the rural areas, the situation is pretty grim. People lack access to adequate healthcare. Even in the metros, locating a pharmacy that would be open late at night or be sure that it would have the required medicine is very difficult. To uniformly distribute healthcare, mobile health can play a vital role. Helping the people to locate such hospitals and medical shops in the neighbourhood using an android application is the primary objective of this project. The user would be feeding his location as the main input, which would be obtained automatically. The coordinates of the user location can be used along with GPS to mark his current location in the map. To locate the nearest hospitals and healthcare centres, Haversine’s formula is used. Nearest pharmacies in the user’s vicinity can also be located by requesting for a particular medicine. This is used as a filter to display the pharmacies which have the required medicine. An ambulance requesting facility has also been provided for emergencies. Once an ambulance is requested, the user’s location is sent to a healthcare centre at which a Java application is used to receive and respond to the user request by dispatching an ambulance to the user location. Additionally, first aid measures along with current trends and news in the field of healthcare would also be available for the users to benefit from. The instantaneous information along with the reach of smart phones can be used to evenly distribute healthcare.

**TABLE OF CONTENTS**

|  |  |  |
| --- | --- | --- |
| CHAPTER | TITLE | PAGE NO |
| 1 | **INTRODUCTION** | **8** |
|  | 1.1 OBJECTIVE | 9 |
|  | 1.2 PROJECT DESCRIPTION | 9 |
|  |  |  |
| 2 | **SYSTEM ANALYSIS** | **11** |
|  | 2.1 EXISTING SYSTEM | 12 |
|  | 2.2 LITERATURE SURVEY | 13 |
|  | 2.3 PROPOSED SYSTEM | 15 |
|  |  |  |
| 3 | **REQUIREMENTS AND DESCRIPTION** | **17** |
|  | 3.1 REQUIREMENTS | 18 |
|  | 3.2 FEATURES OF HARDWARE COMPONENTS | 18 |
|  | 3.3 FEATURES OF SOFTWARE COMPONENTS | 20 |
|  |  |  |
| 4 | **SYSTEM DESIGN** | **24** |
|  | 4.1 SYSTEM ARCHITECTURE | 25 |
|  | 4.2 SYSTEM DESCRIPTION | 26 |
|  |  |  |
| 5 | **IMPLEMENTATION AND CODING** | **29** |
|  | 5.1 SOURCE CODING | 30 |
|  |  |  |
| 6 | **TESTING** | **48** |
|  | 6.1 TESTING OBJECTIVES | 49 |
|  | 6.2 SOFTWARE TESTING AND VALIDATION | 49 |
|  | 6.3 TESTING TACTICS | 51 |
|  |  |  |
| 7 | **OUTPUT SCREENSHOTS** | **54** |
|  | 7.1 PERMISSION SCREEN | 55 |
|  | 7.2 SPLASH SCREEN ACTIVITY | 56 |
|  | 7.3 HOME SCREEN ACTIVITY | 57 |
|  | 7.4 HEALTHCARE NEWS | 58 |
|  | 7.5 HOSPITAL LOCATOR | 59 |
|  | 7.6 PHARMACY LOCATOR | 60 |
|  | 7.7 EMERGENCY AMBULANCE REQUESTOR | 61 |
|  | 7.8 INTEGRATED INSTALLABLE APK | 62 |
|  | 7.9. AMBULANCE SERVER SIDE | 63 |
|  |  |  |
| 8 | **CONCLUSION** | **64** |
|  | 8.1 CONCLUSION | 65 |
|  | 8.2 FUTURE ENHANCEMENTS | 65 |
|  |  |  |
| 9 | **REFERENCES** | 67 |

**LIST OF FIGURES**

|  |  |  |
| --- | --- | --- |
| **Figure No** | **Figure Name** | **Page No** |
| 3.1 | MOBILE DEVICE WITH ANDROID OS | 19 |
| 3.2 | ANDROID ARCHITECTURE | 22 |
| 4.1 | SYSTEM ARCHITECTURE | 25 |
| 4.2 | MODULE BLOCK DIAGRAM | 26 |

**INTRODUCTION**

1. **INTRODUCTION**
   1. **OBJECTIVE**

The main objective of the project is to help users locate the nearest hospitals and pharmacies which have the requested healthcare service. This project is aimed at providing a faster and error free system for the distribution of healthcare in an efficient way.

* 1. **PROJECT DESCRIPTION**

The quality of healthcare in India is not uniform. The urban areas have adequate quality that is comparable with western standards, but in the rural areas, the situation is pretty grim. People lack access to adequate healthcare. Even in the metros, locating a pharmacy that would be open late at night or be sure that it would have the required medicine is very difficult. To uniformly distribute healthcare, mobile health can play a vital role. Helping the people to locate such hospitals and medical shops in the neighbourhood using an android application is the primary objective of this project.

The user would be feeding his location as the main input, which would be obtained automatically by using the hardware GPS locator. The coordinates of the user location can be used along with GPS to locate the hospitals and healthcare centres in their immediate vicinity. The nearest pharmacies in the user’s vicinity can also be located by requesting for a particular medicine. This is used as a filter to display the pharmacies which have the required medicine. If a particular pharmacy doesn’t have the requested medicine then the next farthest is shown as the nearest pharmacy.

Locating the nearest pharmacy and hospital are done using a basic Circle formula and using a simple radius technique to locate and plot the healthcare centres that fall inside the radius. The centres are all marked and located on the Google provided Google Maps.

An ambulance requesting facility has also been provided for emergencies. Once an ambulance is requested, the user’s location is sent to a healthcare centre using socket programming. The healthcare centre receives this request and then dispatches an ambulance to the user location. Additionally, first aid measures along with current trends and news in the field of healthcare and medicine would also be available for the users to benefit from. The instantaneous information along with the reach of smart phones can be used to evenly distribute healthcare. The reach of the Android OS platform and the advent of the smart phone revolution can help improve the reach of this project.

This chapter focuses on throwing some light on the primary objective of this project and its basic functionality. Further details about its functionality and system design and architecture will be covered in the upcoming chapters.

**SYSTEM ANALYSIS**

1. **SYSTEM ANALYSIS**
   1. **EXISTING SYSTEM**
      1. **UNARX**

The Unarx card website allows the users to search based on the **zip-code** and it marks the hospitals located in that area and is being used in the **United States Of America**. Also displays the prices of drugs based on the zip-code.

* + 1. **FIND A PHARMACY**

This website allows users to search based on their **address** and is available only in **Australia.** It also lets the users to choose the facilities that they would need in the hospital or pharmacy. The users can also find out the major languages spoken in the pharmacy and hospital.

* + 1. **APOLLO PHARMACY**

In **India**, Apollo Pharmacy lets the user call the pharmacy and inform them about the medicines required. The medicines would be delivered to their home.

**2.3.1.1.Online Shopping**:Apollo pharmacy also supports online shopping . The user has to provide the names of the drugs with the description and the drugs would be home-delivered.

* + 1. **LOCATION BASED EMERGENCY MEDICAL ASSISTANCE SERVICE SYSTEM**

It uses OpenStreetMap, to locate the nearest hospitals or pharmacies but is available only in **Bangladesh**.It is an open source android application that allows users to add medical amenities by providing the location.

Most of the existing systems are popular and used efficiently in Western Countries. Due to the lack of initiative required to map and plot the complex architectural design of populated Countries like ours, such mobile healthcare systems are extremely rare. In Bangladesh, very recently the Location Based Emergency Medical Assistance System has been setup using Android coupled with OpenStreetMap.

In India, the most popular online medical system is the Apollo Pharmacy System and its major drawback is that it basically only considers and shows the Apollo hospitals and pharmacies and doesn’t display any other privately owned healthcare centre.

**2.1.5 DISADVANTAGES**

* The existing system in our Country is an online web page system, hence losing the mobile aspect.
* OpenStreetMap is a wiki initiative and is open source and is rather inaccurate.
* There is no common application to map and locate all the hospitals and pharmacies in our country, not even local to a particular city
  1. **LITERATURE SURVEY**
     1. **LOCATION BASED EMERGENCY MEDICAL ASSISTANCE SYSTEM USING OPENSTREETMAPS**

This paper was proposed by Rajib Chnadra Das and Tauhidul Alam, at the IEEE International Conference On Informatics ,Electronics and Vision in the year 2014. The paper proposes a system which uses Android OS and OpenStreetMaps to map and mark hospitals and pharmacies. The concept is to use mobile health to distribute healthcare evenly.

**2.2.2** **AUTOMATIC AMBULANCE RESCUE SYSTEM**

**Automatic Ambulance Rescue System** was presented by K.Athavan,S.Jagadeeshwaran,G.Balasubramanian,N.Dinesh,G.Abhilash and G.Gokul and was published in the International Journal of Advanced Technology & Engineering Research in the year 2012. It proposes an automated system to locate accidents and dispatch ambulances immediately to the site of the accident.

* + 1. **LOCATION BASED MOBILE CARDIAC EMERGENCY SYSTEM**

This paper was presented by Keikhosrokiani, N. Mustaffa, N. Zakaria and M. I. Sarwar, at the Global Health conference in 2012. The proposed system uses location awareness to prevent cardiac arrest by having an emergency system in place, such as an emergency ambulance requesting service along with automatic calling of immediate family.

* + 1. **WIRELESS MULTIMEDIA COMMUNICATION TOWARD MOBILE TELEMEDICINE**

This paper was put forth by Chin-Feng Lin and Hsin-Wang Lee in the 9th WSEAS international conference on Applied informatics and communication Moscow, Russia in 2009. It proposes a system which uses user trust management to provide a mobile location aware healthcare system.

* 1. **PROPOSED SYSTEM**

The distribution of healthcare evenly is a vast country like India is a multi-layered issue which is not free from various hurdles, often accompanied by uncertainty. The proposed system uses the concept of mobile healthcare, and it uses Android OS to provide a location aware system to provide healthcare services. The project is designed as an Android application capable of providing location intelligent services such as locating nearby pharmacies and hospitals. The application can also be used to request for an ambulance to particular location.

In the proposed system, we are applying simple circle and Haversine formula in Google Map Places API to locate and mark the healthcare centers on the Map. The Haversine formula can be applied for spheres as follows,

\operatorname{haversin}\left(\frac{d}{r}\right) = \operatorname{haversin}(\phi_2 - \phi_1) + \cos(\phi_1) \cos(\phi_2)\operatorname{haversin}(\lambda_2-\lambda_1)

Where haversin is a function given by:

\operatorname{haversin}(\theta)=\sin^2\left(\frac{\theta}{2}\right)=\frac{1-\cos(\theta)}{2}

Considering the Earth to be an approximate sphere, the distance ‘d’ can be calculated by using inverse haversine or arcsine function as follows:

d = 2 r \arcsin\left(\sqrt{\operatorname{haversin}(\phi_2 - \phi_1) + \cos(\phi_1) \cos(\phi_2)\operatorname{haversin}(\lambda_2-\lambda_1)}\right)

Or

 = 2 r \arcsin\left(\sqrt{\sin^2\left(\frac{\phi_2 - \phi_1}{2}\right) + \cos(\phi_1) \cos(\phi_2)\sin^2\left(\frac{\lambda_2 - \lambda_1}{2}\right)}\right)

Either formula is only an approximation considering that Earth is not a perfect sphere.

Pharmacies can also be displayed by the proposed system. It uses a filter by medicine, to display only the pharmacies which have a particular medicine. Along with the location system, the application also has an ambulance requesting system which sends the healthcare centers the location of user to dispatch an ambulance to that location. It also has a RSS feed to provide the latest news and updates in the field of medicine

This project is aimed at developing a prototype for a location based healthcare service system using the simplicity of Android OS and the popularity of smart phones. The mobile aspect of the system along with Google Maps makes it very useful as a location aware service system and also provides high performance and accuracy, thus making this a better system in terms of functionality, performance and accuracy.

This chapter covers the proposed systems which have been studied and also the literature survey done in order to get a better understanding of how to implement our proposed system. Finally, considering the existing systems as a source of inspiration, the proposed system is explained along with its expected functionality and its potential.

**REQUIREMENTS AND  
DESCRIPTION**

**3.REQUIREMENTS AND DESCRIPTIONS**

**3.1 REQUIREMENTS**

The following descriptions list the basic hardware and software requirements for the project needed for its proper functioning.

**3.1.1. HARDWARE REQUIREMENTS**

* + - * Personal Computer
      * Mobile Device
      * Mobile USB Connector

**3.1.2. SOFTWARE REQUIREMENTS**

* + - * Eclipse SDK
      * Android Studio SDK
      * ADT Plugin for Eclipse
      * Google Maps V2

**3.2 FEATURES OF HARDWARE COMPONENTS**

**3.2.1 MOBILE DEVICE**

A mobile device is also referred to as handheld devices. Handheld device or handheld computer is a pint-sized computing device and usually come with a touch or non-touch display screen and sometimes, even a mini keyboard.



**Figure 3.1. Mobile device with Android OS**

The mobile device should support the Android Operating System for performing

multi-tasking, good user interface and rich notifications.

**3.2.2 USB CONNECTOR**

Universal Serial Bus (USB) is an industry standard that defines the cables,

connectors and communication protocols

used in a bus for connection, communication and power supply between computers and electronic devices.

Connecting a USB device to a computer is simple -- you find the USB connector on the back of your machine and plug the USB connector into it. If it's a new device, the operating system auto-detects it and asks for the driver disk. If the device has already been installed, the computer activates it and starts talking to it. USB devices can be connected and disconnected at any time. Many USB devices come with their own built-in cable, and the cable has an "A" connection on it. If not, then the device has a socket on it that accepts a USB "B" connector.

The USB standard uses "A" and "B" connectors to avoid confusion:

* + "A" connectors head "upstream" toward the computer.
  + "B" connectors head "downstream" and connect to individual devices.

**3.3 FEATURES OF SOFTWARE COMPONENTS**

**3.3.1 ECLIPSE SDK**

The Eclipse platform itself is structured as subsystems which are implemented in one or more plug-ins. The subsystems are built on top of a small runtime engine. The figure below depicts a simplified view. The term Workbench refers to the desktop development environment. The Workbench aims to achieve seamless tool integration and controlled openness by providing a common paradigm for the creation, management, and navigation of workspace resources. Each Workbench window contains one or more perspectives. Perspectives contain views and editors and control what appears in certain menus and tool bars. More than one Workbench window can exist on the desktop at any given time.

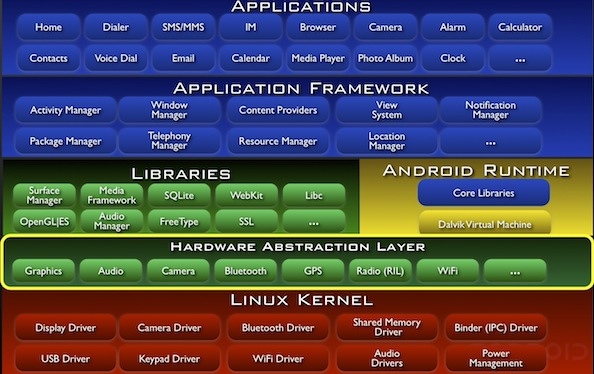
**3.3.2 ANDROID SDK**

Android is a software stack for mobile devices that includes an operating system, middleware and key applications. The Android SDK provides the tools and APIs necessary to begin developing applications on the Android platform using the Java programming language. A software development kit that enables developers to create applications for the Android platform. The Android SDK includes sample projects with source code, development tools, an emulator, and required libraries to build Android applications. Applications are written using the Java programming

language and run on Dalvik, a custom virtual machine designed for embedded use which runs on top of a Linux kernel.

**3.3.2.1 FEATURES**

* + - * + Application framework enabling reuse and replacement of components
        + Dalvik virtual machine optimized for mobile devices
        + Integrated browser based on the open source WebKit engine
        + Optimized graphics powered by a custom 2D graphics library; 3D graphics based on the OpenGL ES 1.0 specification (hardware acceleration optional)
        + SQLite for structured data storage
        + Media support for common audio, video, and still image formats (MPEG4,H.264, MP3, AAC, AMR, JPG, PNG, GIF)
        + GSM Telephony (hardware dependent)
        + Bluetooth, EDGE, 3G, and Wi-Fi (hardware dependent)
        + Camera, GPS, compass, and accelerometer (hardware dependent)
        + Rich development environment including a device emulator, tools for debugging, memory and performance profiling, and an ADT plug-in for the Eclipse IDE



**Figure 3.2. Android Architecture**

**3.3.3 Android NDK**

The Android NDK is a toolset that lets you embed components that make use of native code in your Android applications. Android applications run in the Dalvik virtual machine. The NDK allows you to implement parts of your applications using native-code languages such as C and C++. This can provide benefits to certain classes of applications, in the form of reuse of existing code and in some cases increased speed.

The NDK provides:

* + - A set of tools and build files used to generate native code libraries from C and C++ sources and now with JAVA APIs.
    - A way to embed the corresponding native libraries into an application package file (.apk) that can be deployed on Android devices.
    - A set of native system headers and libraries that will be supported in all future versions of the Android platform, starting from Android 1.5. Applications that use native activities must be run on Android 2.3 or later.
    - Documentation, samples, and tutorials.

**3.3.4 ADT PLUGIN**

Android Development Tools (ADT) is a plugin for the Eclipse IDE that is designed to give you a powerful, integrated environment in which to build Android applications. ADT extends the capabilities of Eclipse to let you quickly set up new Android projects, create an application UI, add components based on the Android Framework API, debug your applications using the Android SDK tools, and even export signed (or unsigned) .apk files in order to distribute your application.

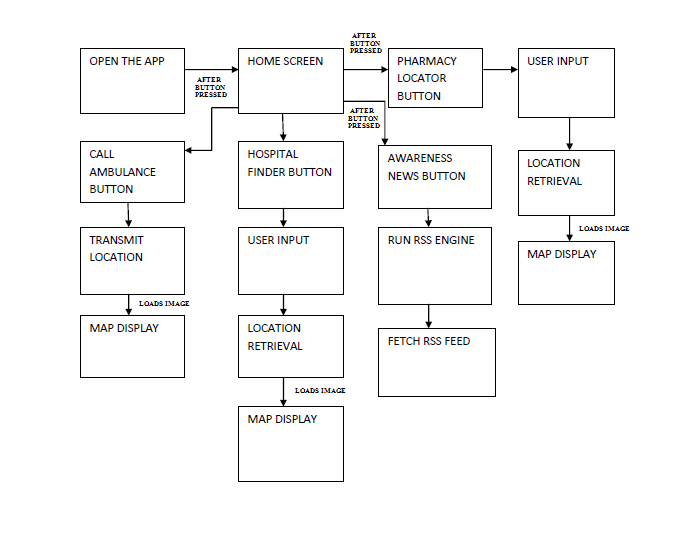
Developing in Eclipse with ADT is highly recommended and is the fastest way to get started. With the guided project setup it provides, as well as tools integration, custom XML editors, and debug output pane, ADT gives you an incredible boost in developing Android applications.

This chapter focuses on highlighting the hardware and software requirements of our proposed system and explains each requirement in detail to facilitate better understanding of how each is used later on.

**SYSTEM DESIGN**

1. **SYSTEM DESIGN**

**4.1 SYSTEM ARCHITECTURE**



**Figure 4.1 SYSTEM ARCHITECTURE**

The working of the System consists of four major parts. Each of the modules are listed below and are later described in detail:

* Receiving the user location using GPS coordinates
* Locating nearby Hospitals and Pharmacies
* Requesting ambulance using GPS Location
* Providing latest trends and news in the field of medicine

**4.2 SYSTEM DESCRIPTION**

**Figure 4.2 MODULE BLOCK DIAGRAM**

The detailed description of each of the module given in the Figure 4.2 is as given below.

* + 1. **HOSPITAL LOCATOR**

The function of this module is simply to use the obtained user location and to use a search option filter based on healthcare type such as Maternity Care, Cancer Treatment to locate the nearest hospital in the user’s vicinity. The nearest hospital is calculated using Haversine formula and using that distance, the healthcare centres in the vicinity are marked.

**4.2.2 PHARMACY LOCATOR**

The function of this module is search the database for a user requested medicine and to display all the pharmacies in the user’s vicinity which have the requested medicine. Again it uses Haversine formula, but uses the required medicine as a filter to show only the pharmacies which have the required medicine.

**4.2.3 AMBULANCE DISPATCH LOCATOR**

This module sends the user location to the servers located at the healthcare centre when the user requests for an ambulance. The application at the server end shows the user location on the map and it is used to dispatch an ambulance to the user location. A socket programming implementation is used to send the GPS coordinates from the user client side to the server side.

**4.2.4 INFORMATION SYSTEM**

The function of this module is to bring the latest trends and news regarding the field of healthcare and medicine at the fingertips of the user. Awareness goes a long way in preventing a majority of accidents and casualties and hence is an important module of this project.

**4.2.5 TESTING AND IMPLEMENTATION**

This module is the final stage of testing in which both separate unit and integration is performed. Since it is a real time project and accuracy is important, it is first tested using a sample space of predefined values followed by real-time testing. Once the various errors have been removed, the system is ready to be implemented by the users and will be available as a downloadable APK for the users.

This chapter focuses on describing in detail the various different modules of the proposed system, their functionality and their conceptualization. Each module is described and at its end the testing module is also described to show that complete testing and debugging of the proposed system will also be done to ensure system success and accuracy.

**IMPLEMENTATION AND**

**CODING**

1. **IMPLEMENTATION AND CODING**

**5.1 SOURCE CODING**

**AndroidManifest.xml**

<?xml version="1.0" encoding="utf-8"?>

<manifest xmlns:android="http://schemas.android.com/apk/res/android"

package="com.example.gpstracking"

android:versionCode="1"

android:versionName="1.0" >

<permission

android:name="info.androidhive.googlemapsv2.permission.MAPS\_RECEIVE"

android:protectionLevel="signature" />

<uses-sdk android:minSdkVersion="11" />

<uses-feature

android:glEsVersion="0x00020000"

android:required="true" />

<uses-permission android:name="info.androidhive.googlemapsv2.permission.MAPS\_RECEIVE" />

<uses-permission android:name="android.permission.ACCESS\_FINE\_LOCATION" />

<uses-permission android:name="android.permission.INTERNET" />

<uses-permission android:name="android.permission.ACCESS\_NETWORK\_STATE" />

<uses-permission android:name="android.permission.CHANGE\_NETWORK\_STATE" />

<uses-permission android:name="android.permission.ACCESS\_WIFI\_STATE" />

<uses-permission android:name="android.permission.CHANGE\_WIFI\_STATE" />

<uses-permission android:name="android.permission.WAKE\_LOCK" />

<uses-permission android:name="android.permission.ACCESS\_COARSE\_LOCATION" />

<uses-permission android:name="com.google.android.providers.gsf.permission.READ\_GSERVICES" />

<uses-permission android:name="android.permission.WRITE\_EXTERNAL\_STORAGE" />

<application

android:icon="@drawable/ic\_launcher"

android:label="M.A.S.S." >

<activity

android:name=".SplashScreen"

android:label="M.A.S.S."

android:screenOrientation="portrait"

android:theme="@android:style/Theme.Black.NoTitleBar" >

<intent-filter>

<action android:name="android.intent.action.MAIN" />

<category android:name="android.intent.category.LAUNCHER" />

</intent-filter>

</activity>

<!-- Main activity -->

<activity

android:name=".AndroidGPSTrackingActivity"

android:label="M.A.S.S." >

<intent-filter>

<action android:name="android.intent.action.MAIN" />

<category android:name="android.intent.category.LAUNCHER" />

</intent-filter>

</activity>

<activity

android:name=".Gmapdisplay"

android:label="M.A.S.S."

android:theme="@android:style/Theme.Black.NoTitleBar" >

<intent-filter>

<action android:name="android.intent.action.MAIN" />

<category android:name="android.intent.category.LAUNCHER" />

</intent-filter>

</activity>

<!-- Splash screen -->

<meta-data

android:name="com.google.android.maps.v2.API\_KEY"

android:value="AIXXXXXXXXXXXXXXXXXXXXXXkuSM" />

</application>

</manifest>

**AndroidGPSTrackingActivity**

package com.example.gpstracking;

import java.io.BufferedReader;

import java.io.IOException;

import android.location.Location;

import android.location.LocationListener;

import android.location.LocationManager;

import android.support.v4.app.FragmentActivity;

import android.widget.Button;

import com.google.android.gms.common.GooglePlayServicesUtil;

import com.google.android.gms.maps.GoogleMap;

import com.google.android.gms.maps.model.LatLng;

import com.google.android.gms.maps.model.MarkerOptions;

import android.app.Activity;

import android.content.Intent;

import android.os.Bundle;

import android.view.View;

import android.widget.Button;

import com.google.android.gms.maps.GoogleMap;

public class AndroidGPSTrackingActivity extends Activity {

private GoogleMap googleMap;

Button btnShowLocation;

@Override

public void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.main);

btnShowLocation = (Button) findViewById(R.id.btnShowLocation);

// show location button click event

btnShowLocation.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View arg0) {

// create class object

Intent i = new Intent(AndroidGPSTrackingActivity.this, Gmapdisplay.class);

startActivity(i);

// close this activity

finish();

// check if GPS enabled

if(gps.canGetLocation())

{

double latitude = gps.getLatitude();

double longitude = gps.getLongitude();

// \n is for new line

Toast.makeText(getApplicationContext(), "Your Location is - \nLat: " + latitude + "\nLong: " + longitude, Toast.LENGTH\_LONG).show();

}

else

{

// can't get location

// GPS or Network is not enabled

// Ask user to enable GPS/network in settings

gps.showSettingsAlert();

}

}

});

}

}

**Gmapdisplay**

package com.example.gpstracking;

import android.os.Bundle;

import android.support.v4.app.FragmentActivity;

import com.google.android.gms.maps.GoogleMap;

import com.google.android.gms.maps.MapFragment;

import com.google.android.gms.maps.model.LatLng;

import com.google.android.gms.maps.GoogleMap;

import com.google.android.gms.maps.MapFragment;

import com.google.android.gms.maps.model.MarkerOptions;

import org.apache.http.HttpRequestFactory;

import org.apache.http.HttpRequest;

import com.google.android.gms.maps.model.LatLng;

public class Gmapdisplay extends FragmentActivity {

private GoogleMap googleMap;

GPSTracker gps;

double latitude;

double longitude;

double radius;

String types[];

Location location = null;

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.mapdisplay);

gps = new GPSTracker(Gmapdisplay.this);

if(gps.canGetLocation())

{

latitude = gps.getLatitude();

longitude = gps.getLongitude();

}

else

{

// can't get location

// GPS or Network is not enabled

gps.showSettingsAlert();

}

try

{

// Loading map

initilizeMap();

}

catch (Exception e)

{

e.printStackTrace();

}

}

/\*\*

\* function to load map. If map is not created it will create it for you

\* \*/

private void initilizeMap()

{

if (googleMap == null)

{

googleMap = ((MapFragment) getFragmentManager().findFragmentById(R.id.map)).getMap();

googleMap.setMapType(GoogleMap.MAP\_TYPE\_NORMAL);

googleMap.getUiSettings().setZoomControlsEnabled(true);

MarkerOptions marker = new MarkerOptions().position(new LatLng(latitude, longitude)).title("Current Location");

googleMap.addMarker(marker);

googleMap.setMyLocationEnabled(true);

googleMap.getUiSettings().setMyLocationButtonEnabled(true);

CameraPosition cameraPosition = new CameraPosition.Builder().target(

new LatLng(latitude, longitude)).zoom(18).build();

googleMap.animateCamera(CameraUpdateFactory.newCameraPosition(cameraPosition));

String geoURI = String.format("geo:%f,%f?q=hospital", latitude, longitude);

Uri geo = Uri.parse(geoURI);

Intent geoMap = new Intent(Intent.ACTION\_VIEW, geo);

startActivity(geoMap);

// check if map is created successfully or not

if (googleMap == null)

{

Toast.makeText(getApplicationContext(),"Sorry! unable to create maps", Toast.LENGTH\_SHORT).show();

}

}

}

@Override

protected void onResume() {

super.onResume();

initilizeMap();

}

}

**GPSTracker**

package com.example.gpstracking;

import android.content.Intent;

import android.location.Location;

import android.location.LocationListener;

import android.location.LocationManager;

import android.os.Bundle;

import android.os.IBinder;

import android.provider.Settings;

import android.util.Log;

public class GPSTracker extends Service implements LocationListener {

private final Context mContext;

// flag for GPS status

boolean isGPSEnabled = false;

// flag for network status

boolean isNetworkEnabled = false;

// flag for GPS status

boolean canGetLocation = false;

Location location; // location

double latitude; // latitude

double longitude; // longitude

// The minimum distance to change Updates in meters

private static final long MIN\_DISTANCE\_CHANGE\_FOR\_UPDATES = 10; // 10 meters

// The minimum time between updates in milliseconds

private static final long MIN\_TIME\_BW\_UPDATES = 1000 \* 60 \* 1; // 1 minute

// Declaring a Location Manager

protected LocationManager locationManager;

public GPSTracker(Context context) {

this.mContext = context;

getLocation();

}

public Location getLocation() {

try {

locationManager = (LocationManager) mContext

.getSystemService(LOCATION\_SERVICE);

// getting GPS status

isGPSEnabled = locationManager

.isProviderEnabled(LocationManager.GPS\_PROVIDER);

// getting network status

isNetworkEnabled = locationManager

.isProviderEnabled(LocationManager.NETWORK\_PROVIDER);

if (!isGPSEnabled && !isNetworkEnabled) {

// no network provider is enabled

} else {

this.canGetLocation = true;

// First get location from Network Provider

if (isNetworkEnabled) {

locationManager.requestLocationUpdates(

LocationManager.NETWORK\_PROVIDER,

MIN\_TIME\_BW\_UPDATES,

MIN\_DISTANCE\_CHANGE\_FOR\_UPDATES, this);

Log.d("Network", "Network");

if (locationManager != null) {

location = locationManager

.getLastKnownLocation(LocationManager.NETWORK\_PROVIDER);

if (location != null) {

latitude = location.getLatitude();

longitude = location.getLongitude();

}

}

}

// if GPS Enabled get lat/long using GPS Services

if (isGPSEnabled) {

if (location == null) {

locationManager.requestLocationUpdates(

LocationManager.GPS\_PROVIDER,

MIN\_TIME\_BW\_UPDATES,

MIN\_DISTANCE\_CHANGE\_FOR\_UPDATES, this);

Log.d("GPS Enabled", "GPS Enabled");

if (locationManager != null) {

location = locationManager

.getLastKnownLocation(LocationManager.GPS\_PROVIDER);

if (location != null) {

latitude = location.getLatitude();

longitude = location.getLongitude();

}

}

}

}

}

} catch (Exception e) {

e.printStackTrace();

}

return location;

}

/\*\*

\* Stop using GPS listener

\* Calling this function will stop using GPS in your app

\* \*/

public void stopUsingGPS(){

if(locationManager != null){

locationManager.removeUpdates(GPSTracker.this);

}

}

// Function to get latitude

public double getLatitude(){

if(location != null){

latitude = location.getLatitude();

}

// return latitude

return latitude;

}

// Function to get longitude

public double getLongitude(){

if(location != null){

longitude = location.getLongitude();

}

// return longitude

return longitude;

}

/\* Function to check GPS/wifi enabled

@return boolean \*/

public boolean canGetLocation() {

return this.canGetLocation;

}

/\*Function to show settings alert dialog

On pressing Settings button will lauch Settings Options \*/

public void showSettingsAlert(){

AlertDialog.Builder alertDialog = new AlertDialog.Builder(mContext);

// Setting Dialog Title

alertDialog.setTitle("GPS is settings");

// Setting Dialog Message

alertDialog.setMessage("GPS is not enabled. Do you want to go to settings menu?");

// On pressing Settings button

alertDialog.setPositiveButton("Settings", new DialogInterface.OnClickListener() {

public void onClick(DialogInterface dialog,int which) {

Intent intent = new Intent(Settings.ACTION\_LOCATION\_SOURCE\_SETTINGS);

mContext.startActivity(intent);

}

});

// on pressing cancel button

alertDialog.setNegativeButton("Cancel", new DialogInterface.OnClickListener() {

public void onClick(DialogInterface dialog, int which) {

dialog.cancel();

}

});

// Showing Alert Message

alertDialog.show();

}

@Override

public void onLocationChanged(Location location) {

}

@Override

public void onProviderDisabled(String provider) {

}

@Override

public void onProviderEnabled(String provider) {

}

@Override

public void onStatusChanged(String provider, int status, Bundle extras) {

}

@Override

public IBinder onBind(Intent arg0) {

return null;

}

}

**SplashScreen**

package com.example.gpstracking;

import android.app.Activity;

import android.content.Intent;

import android.os.Bundle;

import android.os.Handler;

public class SplashScreen extends Activity {

// Splash screen timer

private static int SPLASH\_TIME\_OUT = 3000;

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_splash);

new Handler().postDelayed(new Runnable() {

@Override

public void run() {

// This method will be executed once the timer is over

// Start your app main activity

Intent i = new Intent(SplashScreen.this, AndroidGPSTrackingActivity.class);

startActivity(i);

// close this activity

finish();

}

}, SPLASH\_TIME\_OUT);

}

}

**activity\_splash.xml**

<?xml version="1.0" encoding="utf-8"?>

<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:background="@drawable/splashbg" >

<ImageView

android:id="@+id/imgLogo"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_centerInParent="true"

android:src="@drawable/ic\_launcher" />

<TextView

android:layout\_width="fill\_parent"

android:layout\_height="wrap\_content"

android:textSize="22dp"

android:textColor="#454545"

android:gravity="center\_horizontal"

android:text="Medical Assistance Service System"

android:layout\_below="@+id/imgLogo"

android:layout\_alignParentLeft="true"

android:layout\_alignParentStart="true" />

<TextView

android:layout\_width="fill\_parent"

android:layout\_height="wrap\_content"

android:textSize="20dp"

android:textColor="#454545"

android:gravity="center\_horizontal"

android:text="Aditya Krishnan Sairaj"

android:id="@+id/textView2"

android:layout\_alignParentBottom="true"

android:layout\_centerHorizontal="true" />

</RelativeLayout>

**main.xml**

<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"

xmlns:tools="http://schemas.android.com/tools" android:layout\_width="match\_parent"

android:layout\_height="match\_parent" android:paddingLeft="@dimen/activity\_horizontal\_margin"

android:paddingRight="@dimen/activity\_horizontal\_margin"

android:paddingTop="@dimen/activity\_vertical\_margin"

android:paddingBottom="@dimen/activity\_vertical\_margin"

android:background="@drawable/blackbg"

tools:context=".AndroidGPSTrackingActivity">

<TextView android:text="Press Button To Know Location" android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:id="@+id/textView"

android:textSize="18dp"

android:layout\_above="@+id/btnShowLocation"

android:layout\_centerHorizontal="true"

android:layout\_marginBottom="80dp" />

<Spinner

android:id="@+id/spr\_place\_type"

android:layout\_width="wrap\_content"

android:layout\_height="60dp"

android:layout\_alignParentTop="true" />

<Button

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="GPS"

android:id="@+id/btnShowLocation"

android:layout\_centerVertical="true"

android:layout\_centerHorizontal="true" />

</RelativeLayout>

**mapdisplay.xml**

<?xml version="1.0" encoding="utf-8"?>

<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"

xmlns:tools="http://schemas.android.com/tools"

android:layout\_width="match\_parent" android:layout\_height="match\_parent">

<fragment

android:id="@+id/map"

android:name="com.google.android.gms.maps.MapFragment"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:layout="@layout/mapdisplay"

tools:layout="@layout/mapdisplay" />

</RelativeLayout>

This chapter focuses on displaying the codes which have been used to implement our proposed system and to provide functionality to it.

**TESTING**

1. **TESTING**

**6.1 TESTING OBJECTIVES**

Testing is the process of executing a program with the aim of finding errors. The main objectives of testing are:

* To use a good test case, which has more probability of finding an undiscovered error.
* To use test cases to uncover a new errors with a minimum amount of time and effort employed.

**6.2 SOFTWARE TESTING AND VALIDATION**

**6.2.1 UNIT TESTING**

In computer programming, unit testing is a software design and development method where the programmer gains confidence that individual units of source code are fit for use. Here we have **tested the individual modules** of our project **using Eclipse IDE** and **Android Studio** by running each individual component on the mobile device separately for a set of test case values. For example, the test case values used by us includes sample GPS coordinates such as 12,80 and so on to see if the required marker for current coordinates is being marked and displayed on the map.

**6.2.2 INTEGRATING TESTING**

Integrating testing‟ (sometimes called Integrating and Testing, abbreviated

I&T) is the phase of grouping the distinct module units into an integrated component and then testing the integrated component in its entirety. We have **integrated all the modules of our project** under a single package and **exported it as a signed APK** and tested it . The generated signed APK is a single testable component which can be downloaded and installed on any Android powered phone and tested. After integration the system is checked for each of its four functionalities. Each function is tested to see if it works as expected.

**6.2.3 VERIFICATION AND VALIDATION**

Validation is the process of ensuring that a program operates on clean, correct and useful data sets. We have verified all the possible types of congestion, inaccuracy and false or outdated data errors. We also **validated the system** without any errors **using a sample set of real time data**. Haversine formula when used at junction points might cause complexities because two or more healthcare centres might have same distance from current user location. These are resolved by arbitrarily choosing one healthcare centre over the other.

**6.2.4 SYSTEM TESTING**

System testing of hardware or software is testing conducted on a complete,

integrated system to evaluate the system’s compliance with its specified requirements. System testing falls within the scope of black box testing, and as such, should require no knowledge of the inner design of the code or logic. It is required to test it to see if the system is complaint from the user view point. We test the system by **running the app from different user location**.

**6.2.5 DEBUGGING**

Debugging is a methodical process of finding and reducing the number of

bugs, errors or defects, in a computer program of electronic hardware thus making it behave as expected. We have debugged the errors occurred in our logic and syntax to finally rid the program of any bugs to create an executable code.

**6.3 TESTING TACTICS**

6.3.1 BLACK BOX TESTING

Black box testing is the testing of the system without knowledge of the internal workings of the code being tested. The success is determined by passing a set of test cases as inputs and receiving the expected outputs as the result. Black Box Testing has been done **on several mobile devices** to ensure project accuracy.

**6.3.2 WHITE BOX TESTING**

White box testing uses an internal perspective of the system to design test

cases based on the internal structure. These test cases are designed to work all the aspects of the code and test its various resulting possibilities and even tested to handle errors and exceptions. **Different test cases** have been used by us to test project’s successful working.

**6.3.3 BASIS PATH TESTING**

A testing mechanism proposed by McCabe whose Aim is to derive a logical

complexity measure of a procedural design and use this as a guide for defining a

basic set of execution paths. This testing considers the various paths the system might take to understand and determine the complexity of the project to ensure that the project is efficient and not too slow in performance and complexity. **Complex test cases** have been used by us to determine project **accuracy and performance.**

**6.3.4 CONTROL STRUCTURE TESTING**

Control structure testing is a group of white-box testing methods use to exhaust all the control structures used in the system to ensure that there exist a complete proper flow in the system. Its types are:

1. Branch Testing

2. Condition Testing

3. Data Flow Testing

4. Loop Testing

* **Branch Testing (Decision Testing):** For every decision, each branch

needs to be executed at least once.

* **Condition Testing:** Condition Testing is a test construction method

that focuses on exercising the logical conditions in each module.

* **Data Flow Testing:** It is used to test if the data flows as expected and in the required format throught each module in the system.
* **Loop Testing:** It will test the various looping constructs and jumps such as Simple Loops, Nested loops, returns, jumps etc.

This chapter focuses on the various testing techniques and how each has been used in relation with our proposed system.

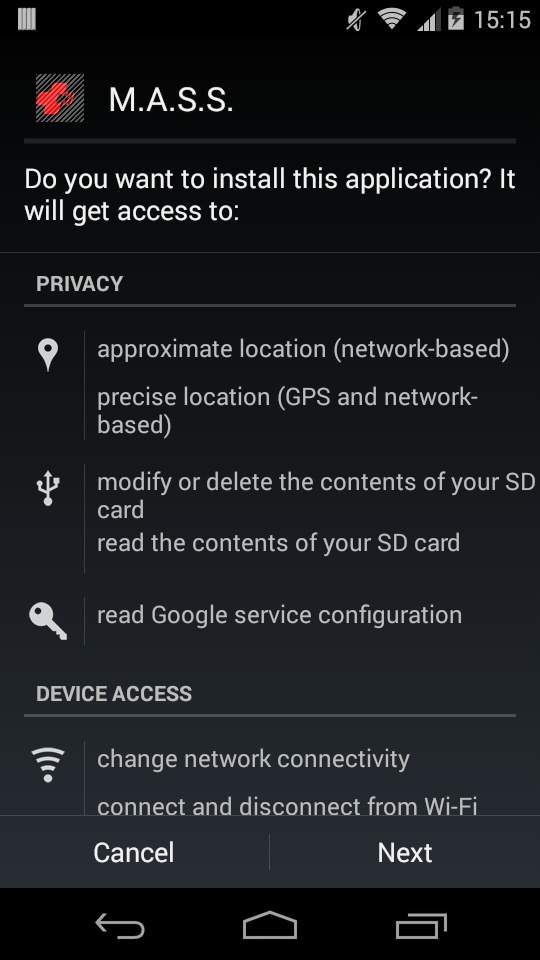
**OUTPUT SCREENSHOTS**

1. **OUTPUT SCREENSHOTS**

The following images show the project in its various stages of development, execution and the resulting Android Application.

**7.1 PERMISSION SCREEN**

To execute Android Apps that require user information such as user location, a permission screen is used to obtain user agreement.



**7.2 SPLASH SCREEN ACTIVITY**

This shows the opening screen, called a splash screen. It is used to display the logo and the authors of the application when the back end data is being received from the online servers.



It acts as a buffer timer while loading the required databases and also the Google Map initialization is done in this activity.

**7.3 HOME SCREEN ACTIVITY**

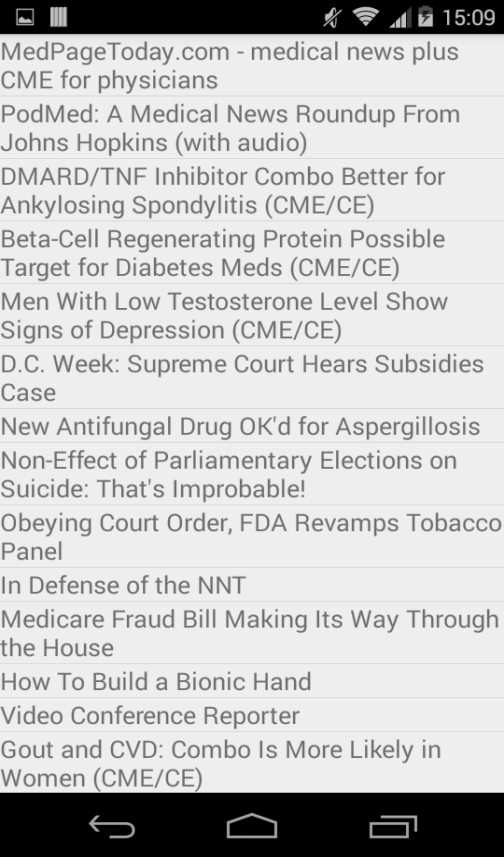
The image given below shows the home screen or the main menu activity through which all the four basic functionalities can be accessed by pressing the required button.



Each of the four buttons provided in the user face opens another distinct activity to perform the required operation

**7.4 HEALTHCARE NEWS**

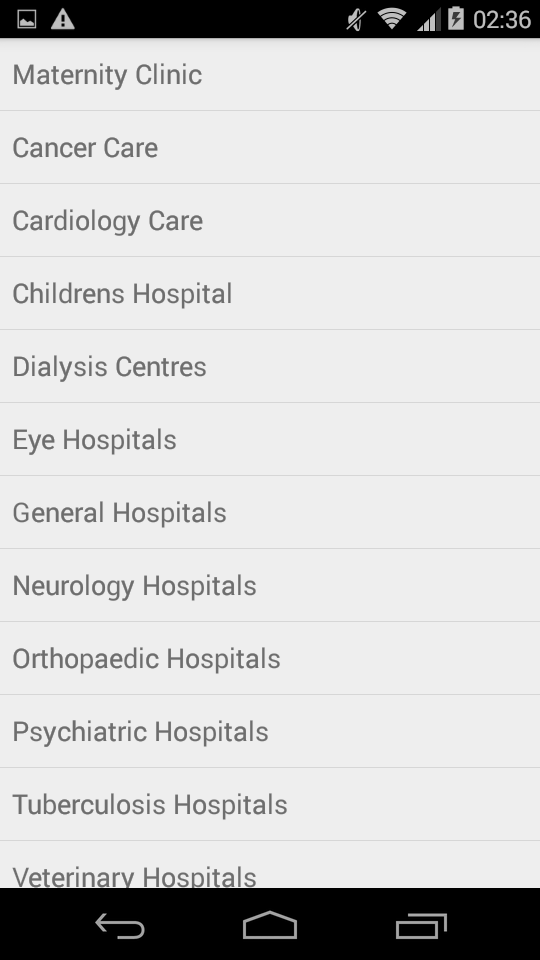
The following image shows the layout of the Healthcare News activity obtained by pressing the appropriate button on the home menu activity.



We see that the News feed has a list of the current trends and news information regarding the field of Medicine which can be read by clicking on the required link.

**7.5 HOSPITAL LOCATOR**

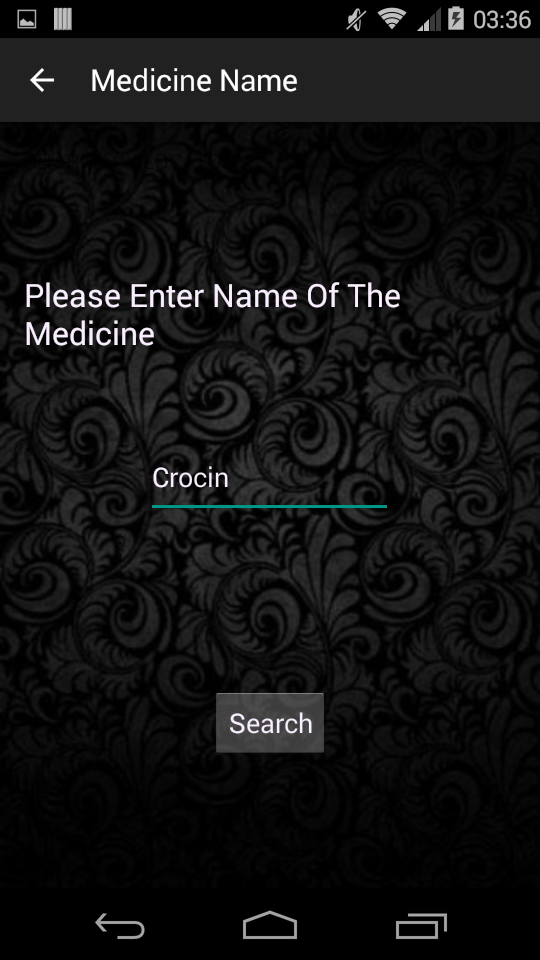
Once the Hospital Locator button is pressed, a list of type of Healthcare facility is first listed to choose the kind of healthcare required by the user.



Once the user selects a particular type of Healthcare, all the nearest healthcare centres which house that particular facility in his vicinity is listed. Along with the nearest, all the hospitals which house the particular facility is also listed to give the user a choice.

**7.6 PHARMACY LOCATOR**

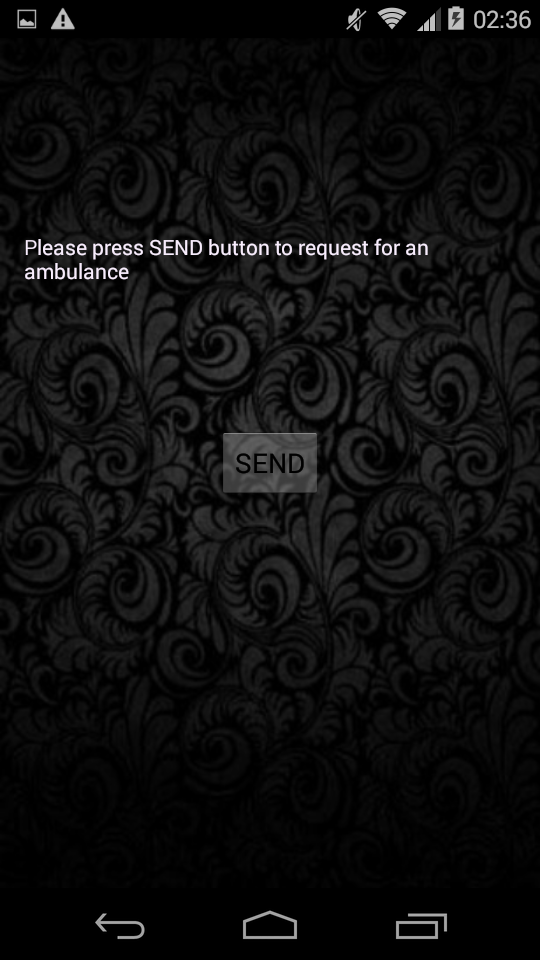
The pharmacy locator is accessed again from the main menu and asks for the required medicine as input from the user. It then uses it as a filter to list the nearest pharmacy that has the required medicine.



The similar concept is used for pharmacies, except that the filter used is the name of the medicine instead of the type of healthcare requested.

**7.7 EMERGENCY AMBULANCE REQUESTOR**

This can be accessed by pressing the Call Ambulance button on the main menu. The send button sends the request fro an ambulance along with user location.



The healthcare centre receives the request along with the user location which is then plotted on a map to guide the ambulance to the user locatio

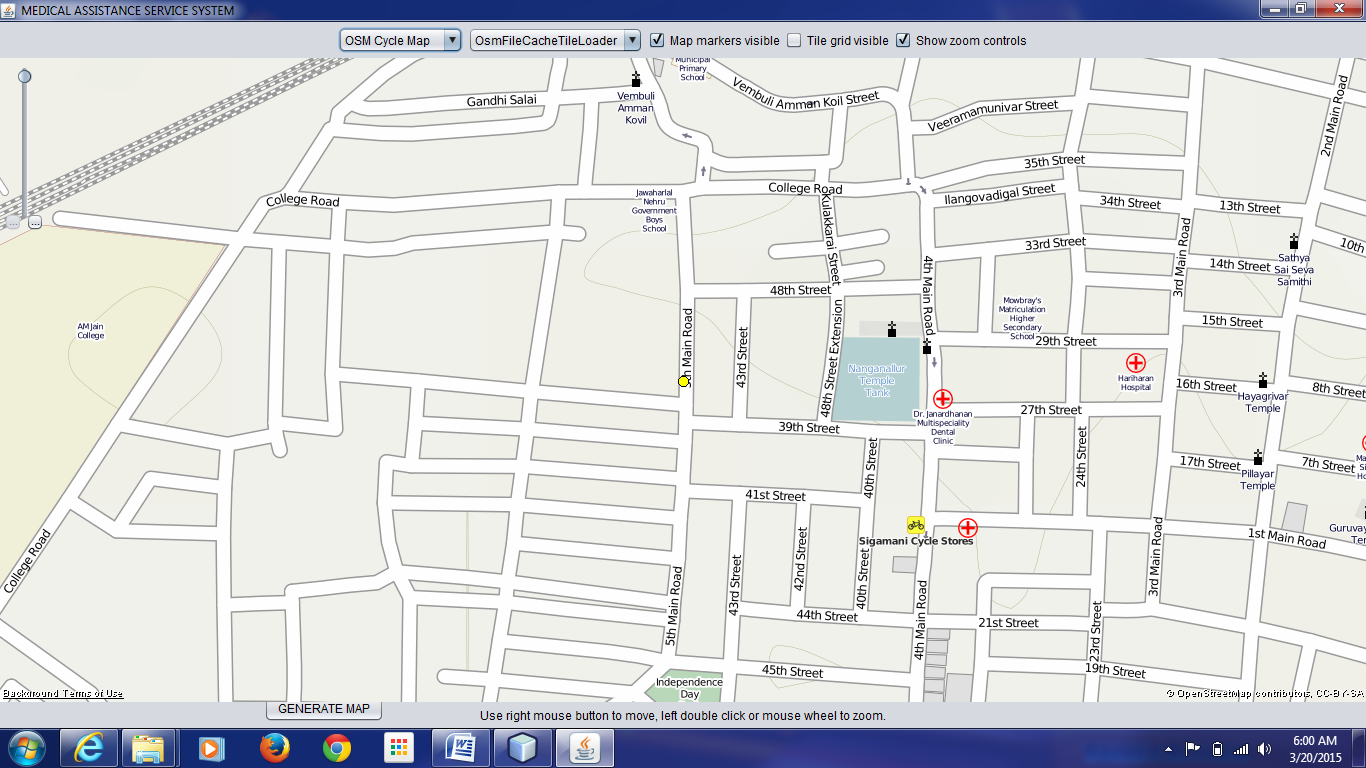
**7.8 INTEGRATED INSTALLABLE APK**

The final signed downloadable Android APK can be made available online for users to download and install and have it on their Android enabled smart phones as an application to avail the functionality of the system.



The installation APK file can be transferred and installed on multiple devices making the impact of mobile healthcare prominent.

**7.9 AMBULANCE SERVER SIDE**



The server side has been implemented using a socket program and the JxMapViewer API to receive the GPS location of the user who is requesting for an ambulance and to display it in Open Street Maps. Additionally satellite view of the location will be provided by Bing Aerial Maps.

This chapter showcases the different screenshots showing the different functionality and the different options available in the proposed system. Explanation of each screenshot is also present to facilitate easier understanding.

**CONCLUSION AND FUTURE ENHANCEMENTS**

1. **CONCLUSION AND FUTURE ENHANCEMENTS**

**8.1 CONCLUSION**

The advent of smart phones has showed us how amazing technology is and how much of an impact it has on our daily life. The purpose of research and development in the field of technology is to enhance the quality of life. As Engineers, it is our duty to contribute to the society. One such contribution is our project. Medical Assistance Service System along with Android powered devices can be very vital in evenly distributing the healthcare services in a populated country like ours. A look at this project shows us that this field of relating technology and medicine is rather new and relatively unexplored. Being a prototype system, if implemented and utilized in the real world, this project will have profound impact on healthcare and location based emergency services in India. A few seconds may save a life and that is the primary goal and the spirit with which this project has been designed and hopefully it will inspire further improvements and successors.

**8.2 FUTURE ENHANCEMENTS**

The following can be considered for further iterations of our project and can be implemented in future designs:

* The local aspect of Medical Assistance Service System can be scaled up to include various cities and metros.
* The app can include a system to plot the shortest route to a healthcare center.
* Trust management can be used to include even the smallest and privately owned hospitals and pharmacies.
* Patient history can be used along with possible symptoms to predict diseases.

This chapter concludes the proposed system and its explanation and also suggests some enhancements which can be made and implemented in the future to improve system functionality.

**REFERENCES**

1. **REFERENCES**

[1] http://unarxcard.com/pharmacy-search.php

[2] http://www.findapharmacy.com.au/Advanced.aspx

[3] http://www.apollopharmacy.in/prescription

[4] http://www.worldlifeexpectancy.com/country-health-profile/india

[5] http://hswsolutions.com/services/mobile-web-development/mobile-website-vs-apps/

[6] http://techcircle.vccircle.com/2014/05/29/india-had- third-largest-smartphone-base-with-117m-users-in-2013-to-grow-45-in-2014-mary-meeker-report

[7]Rajib Chandra Das and Tauhidul Alam , ”**Location Based Emergency Medical Assistance System Using OpenStreetMap**”, IEEE International Conference on Informatics, Electronics &Vision (ICIEV’14)

[8] K.Athavan,”**Automatic Ambulance Rescue System**”, International Journal of Advanced Technology & Engineering Research (2012)

[9] P. Keikhosrokiani, N. Mustaffa, N. Zakaria and M. I. Sarwar, “**A**

**Proposal to Design a Location-Based Mobile Cardiac Emergency**

**System (LMCES)**,” Global Telehealth 2012.

[10] Chin-Feng Lin and Hsin-Wang Lee, "**Wireless multimedia**

**communication toward mobile telemedicine,**" in Proceedings of the

9th WSEAS international conference on Applied informatics and

communications Moscow, Russia: World Scientific and Engineering

Academy and Society (WSEAS), 2009, pp. 232-237.

[11] Mobile Health, “http://en.wikipedia.org/wiki/MHealth”

[12] OpenStreetMap, “http://wiki.openstreetmap.org/wiki/OpenStreetMap”

[13] Android, “http://en.wikipedia.org/wiki/Android\_(operating\_system)”