**CHAPTER 1**

**INTRODUCTION**

* 1. **Overview of the Project**

With the improvement of living standards, our daily life is hasting in an inevitable race. It is impossible to remember all the events of the day. The existing applications allow the user to keep notes, calendar remainders and alarms. Keeping notes and reminders does not include location features. The alarm feature is set based on system time which limits the user to depend on uncertain time constraints. Also sending text messages while driving is not safe. So an application that has integrated all these features including location services for alerting the user about the tasks will serve better. GPS (Global Positioning System) function is implanted in smart phone, and the built-in electronic map can help its holder acquire the surrounding information. The user will be directed to their destination location with advanced navigation system. Once the application is engaged the alarm is used to alert the user about the task. The user is provided with choices for sending location as text message and a map for navigation purpose. Since text messaging, internet and GPS have become so common in all Smartphones, this application will be very easy to use with just the in-built features of the Smartphone.

* 1. **Objective**

Usage of mobile phone to alert the person at the destination location about the arrival is tedious while driving a vehicle. This may cause road accidents.

The existing system allows the user to send SMS using voice input. But the user has to use the mobile phone while driving.

The app keeps track of the user’s tasks using the calendar feature.

The current user location is tracked using Global Positioning System (GPS) and checks it with the destination location from time to time.

Estimates the distance & the approximate time required to reach the destination location and alerts the user.

When the user is nearing the destination location, a SMS is sent to the person at the destination location.

* 1. **Features**

A lot of recent research efforts have been devoted in the field of Mobile communication and GPS technology.

* End users can make use of their mobile phone to set reminders.
* We will have an application installed in the mobile phone which will alert the user about their meetings.
* The user will start traveling with the help of the navigation support provided by the application.
* The user also can alert the person at the destination with their current location, using text messages.

**CHAPTER 2**

**SYSTEM ANALYSIS**

**2.1 EXISTING SYSTEM**

The existing system allows the user to send SMS using voice input. But the user has to use the mobile phone while driving.

**Disadvantages of Existing System**

* Alarms are based on time.
* Arrival timings are subject to change due to traffic.
* Certain operations are available in separate applications but none of them are integrated properly to serve all the user requirements.

**2.2 PROPOSED SYSTEM**

The proposed system allows the user to use an app that keeps track of the user’s tasks using the calendar feature. The current user location is tracked using Global Positioning System (GPS) and checks it with the destination location from time to time. Estimates the distance & the approximate time required to reach the destination location and alerts the user. When the user is nearing the destination location, a SMS is sent to the person at the destination location.

**Advantages of Proposed System**

* User can set alarm along with the destination location.
* Uses in-built alarm features for less memory consumption.
* Less time consumption.
* Easy to use.

**2.3 FEASIBILITY STUDY**

Having established a system, one has to determine whether an alternative system is feasible compared to existing system. An analysis of the ability to complete a project successfully, taking into account legal, economic, technological, scheduling and other factors. Rather than just diving into a project and hoping for the best, a feasibility study allows project managers to investigate the possible negative and positive outcomes of a project before investing too much time and money. Feasibility study is conducted by analyzing the project. We develop system flowchart that identify the characteristics of the existing system, evaluate the performance of the system, cost data, and select the best system for the job.

**2.3.1 Economic feasibility**

It is the most frequently used method for evaluating the effectiveness of a candidate system. The weight cost is obtained then the decision is made to design and implement the system. Thus to pay a less cost of effort will be need for this project.

**2.3.2 Operational feasibility**

The purpose of Operational Feasibility study is to determine whether the new system will be used if it is developed and implemented or will there be resistance from the users that will take the possible application benefits.

**2.3.3 Technical feasibility**

It centers on the existing computer system and to what extent it can support the proposed addition. This involves financial consideration to accommodation technical feasibility.

**2.4. Software quality attributes**

* **Reliability:** Maturity, fault tolerance and recoverability are good in the system.
* **Usability:** The program is easy to understand and learn. The program is customizable and many shapes with their relevant result can be programmed.
* **Efficiency:** Depends on resource behavior like the camera resolution.
* **Maintainability:** The system maintenance is very easy as there are no interfaces or complex hardware being used.
* **Portability:** The program can be easily transferred to any other environment, including install ability.

**CHAPTER 3**

**SYSTEM SPECIFICATION**

**3.1 HARDWARE SPECIFICATIONS**

Processor : 1 GHz CPU

Memory : 512MB RAM

MOBILE : ANDROID 2.3 (API level 9-Gingerbread) or above.

**3.2 SOFTWARE SPECIFICATIONS**

Operating System : WINDOWS XP SP2 or higher.

Front – End : Android (Java & XML)

Open source Software : Eclipse 3.5 with

Android SDK & Phone emulator.

**3.3 SOFTWARE REQUIREMENTS SPECIFICATION (SRS)**

**3.3.1 Functional Requirements**

Function uses SMS manager, maps and alarm intent methods that are available in android. A user defined function is written to get the current location of the user and check it with the destination location at specific time intervals.

The input to the system is text, number and location coordinates. The text input for event name and person name. Number input is used for phone number field. Time picker is used to set the time for the alarm and maps are used to get the location inputs.

**3.3.2 Performance Requirements**

The performance is not affected by any number of simultaneous users. The system can even work with several events and track them.

The response time is quick. Once the event is stored, the system invokes the application to alert the user.

The program uses Eclipse IDE to work and thus error detection and isolation is very easy. The error descriptions are clearly dealt with this IDE.

**3.3.4 Design/Implementation constraints**

There is no hardware limitation with this system, as the program could work on any android system that has enabled GPS technology, the system must support SMS and GPRS facilities.

**CHAPTER 4**

**SOFTWARE DESCRIPTION**

**4.1. ANDROID**

**Android** is a [Linux](http://en.wikipedia.org/wiki/Linux)-based [operating system for mobile devices](http://en.wikipedia.org/wiki/Mobile_operating_system) such as [smart phones](http://en.wikipedia.org/wiki/Smartphone) and [tablet computers](http://en.wikipedia.org/wiki/Tablet_computer). It is developed by the [Open Handset Alliance](http://en.wikipedia.org/wiki/Open_Handset_Alliance) led by [Google](http://en.wikipedia.org/wiki/Google)

Android has a large community of developers writing applications ("[apps](http://en.wikipedia.org/wiki/Mobile_apps)") that extend the functionality of the devices. Developers write primarily in a customized version of [Java](http://en.wikipedia.org/wiki/Java_(programming_language)).

Android was listed as the best-selling Smartphone platform worldwide in Q4 2010 by Canalys with over 200 million Android devices in use by November 2011. According to Google's [Andy Rubin](http://en.wikipedia.org/wiki/Andy_Rubin), as of December 2011 there are over 700,000 Android devices activated every day.

**4.1.1. FOUNDATION**

Android, Inc. was founded in [Palo Alto, California](http://en.wikipedia.org/wiki/Palo_Alto,_California), United States in October, 2003 by [Andy Rubin](http://en.wikipedia.org/wiki/Andy_Rubin) (co-founder of [Danger](http://en.wikipedia.org/wiki/Danger_(company))), [Rich Miner](http://en.wikipedia.org/wiki/Rich_Miner) (co-founder of Wildfire Communications, Inc.), Nick Sears (once VP at [T-Mobile](http://en.wikipedia.org/wiki/T-Mobile_USA)), and Chris White (headed design and interface development at [WebTV](http://en.wikipedia.org/wiki/WebTV)) to develop, in Rubin's words "...smarter mobile devices that are more aware of its owner's location and preferences". Despite the obvious past accomplishments of the founders and early employees, Android Inc. operated secretly, revealing only that it was working on software for mobile phones.

**4.1.2. ACQUISITION BY GOOGLE**

[Google](http://en.wikipedia.org/wiki/Google) acquired Android Inc. on August 17, 2005, making Android Inc. a wholly owned subsidiary of Google Inc. Key employees of Android, including Andy Rubin, Rich Miner and Chris White, stayed at the company after the acquisition. Not much was known about Android Inc. at the time of the acquisition, but many assumed that Google was planning to enter the [mobile phone](http://en.wikipedia.org/wiki/Mobile_phone) market with this move. Google purchased android in 1 million US dollars.

## 4.1.3. DESIGN

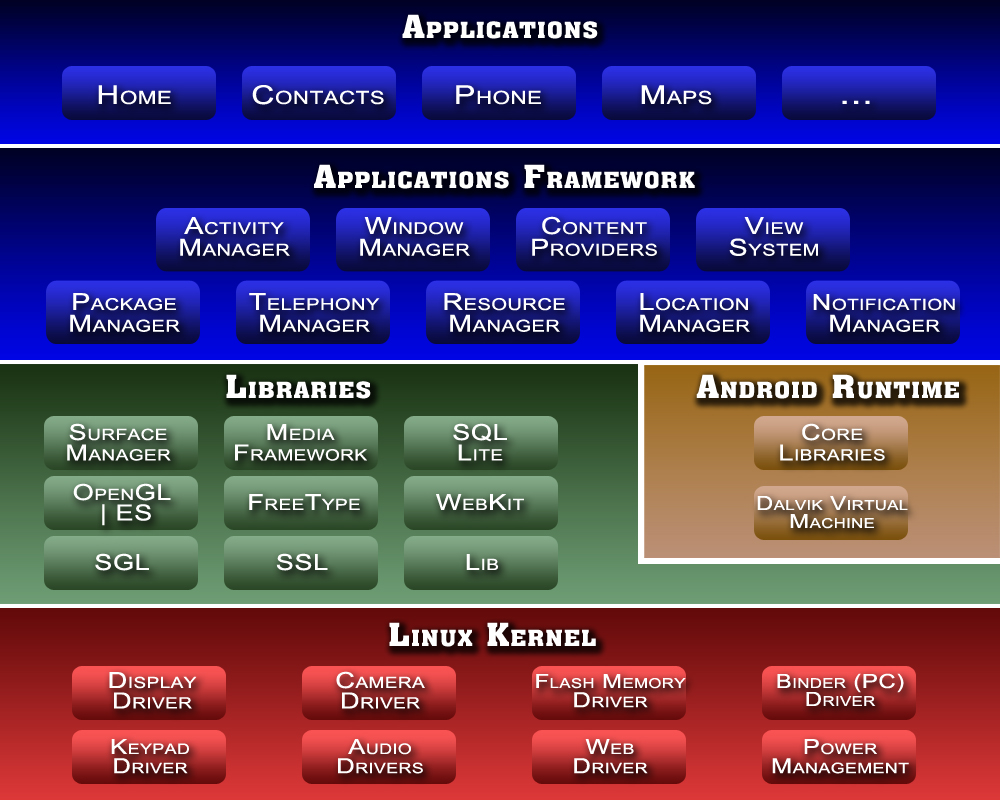


Figure 4.1: Android Architecture Diagram

Android consists of a kernel based on Linux, with middleware, libraries and APIs written in C and Application software running on an application framework which includes Java-compatible libraries based on Apache Harmony. Android uses Dalvik virtual Machine with just-in-time compilation to run Dalvik dex-code (Dalvik Executable), which is usually from java bytecode.

### 4.1.4. LINUX

Android's kernel is based on the [Linux kernel](http://en.wikipedia.org/wiki/Linux_kernel) and has further architecture changes by Google outside the typical Linux kernel development cycle. Android does not have a native [X Window System](http://en.wikipedia.org/wiki/X_Window_System) nor does it support the full set of standard [GNU](http://en.wikipedia.org/wiki/GNU) libraries, and this makes it difficult to port existing Linux applications or libraries to Android.

**4.2. FEATURES AND SPECIFICATIONS**

* + **Handset layouts**

The platform is adaptable to larger, [VGA](http://en.wikipedia.org/wiki/Video_Graphics_Array), [2D graphics](http://en.wikipedia.org/wiki/2D_computer_graphics) library, [3D graphics](http://en.wikipedia.org/wiki/3D_computer_graphics) library based on [OpenGL ES](http://en.wikipedia.org/wiki/OpenGL_ES) 2.0 specifications, and traditional Smartphone layouts.

* + **Storage**

Android supports connectivity technologies including GSM/EDGE, IDEN, CDMA, EV-O, UMTS, Bluetooth, Wi-Fi, LTE, NFC and WiMAX..

* + **Connectivity**

Android supports connectivity technologies including [GSM](http://en.wikipedia.org/wiki/GSM)/[EDGE](http://en.wikipedia.org/wiki/Enhanced_Data_Rates_for_GSM_Evolution), [IDEN](http://en.wikipedia.org/wiki/Integrated_Digital_Enhanced_Network), [CDMA](http://en.wikipedia.org/wiki/Code_division_multiple_access), [EV-O](http://en.wikipedia.org/wiki/Evolution-Data_Optimized), [UMTS](http://en.wikipedia.org/wiki/Universal_Mobile_Telecommunications_System), [Bluetooth](http://en.wikipedia.org/wiki/Bluetooth), [Wi-Fi](http://en.wikipedia.org/wiki/Wi-Fi), [LTE](http://en.wikipedia.org/wiki/LTE_Advanced), [NFC](http://en.wikipedia.org/wiki/Near_field_communication) and [WiMAX](http://en.wikipedia.org/wiki/WiMAX).

* + **Messaging**

[SMS](http://en.wikipedia.org/wiki/SMS) and [MMS](http://en.wikipedia.org/wiki/Multimedia_Messaging_Service) are available forms of messaging, including threaded [text messaging](http://en.wikipedia.org/wiki/Text_messaging) and now [Android Cloud To Device Messaging](http://en.wikipedia.org/wiki/Android_Cloud_To_Device_Messaging) (C2DM) is also a part of Android Push Messaging service.

* + **Multiple language support**

Android supports multiple languages.

* + **Web browser**

The web browser available in Android is based on the open-source [Web Kit](http://en.wikipedia.org/wiki/WebKit) layout engine, coupled with [Chrome's](http://en.wikipedia.org/wiki/Google_Chrome) [V8 JavaScript engine](http://en.wikipedia.org/wiki/V8_JavaScript_engine). The browser scores 100/100 on the [Acid3](http://en.wikipedia.org/wiki/Acid3#Mobile_browsers) test on Android 4.0.

* + **Java support**

While most Android applications are written in [Java](http://en.wikipedia.org/wiki/Java_(programming_language)), there is no [Java Virtual Machine](http://en.wikipedia.org/wiki/Java_Virtual_Machine) in the platform and Java byte code is not executed. Java classes are compiled into Dalvik executables and run on [Dalvik](http://en.wikipedia.org/wiki/Dalvik_virtual_machine" \o "Dalvik virtual machine), a specialized virtual machine designed specifically for Android and optimized for battery-powered mobile devices with limited memory and CPU. [J2ME](http://en.wikipedia.org/wiki/J2ME) support can be provided via third-party applications.

* **Streaming media support**

RTP/RTSP streaming ([3GPP PSS](http://en.wikipedia.org/w/index.php?title=3GPP_PSS&action=edit&redlink=1), [ISMA](http://en.wikipedia.org/wiki/Internet_Streaming_Media_Alliance)), HTML progressive download ([HTML5 <video> tag](http://en.wikipedia.org/wiki/HTML5_video)). Adobe Flash Streaming (RTMP) and HTTP Dynamic streaming are supported by the [Flash plug-in](http://en.wikipedia.org/wiki/Adobe_Flash_Player#Mobile_platforms). Apple HTTP Live Streaming is supported by [RealPlayer for Android](http://en.wikipedia.org/wiki/RealPlayer_for_Android), and by the operating system in Android 3.0 (Honeycomb).

* + **Multi-touch**

Android has native support for [multi-touch](http://en.wikipedia.org/wiki/Multi-touch) which was initially made available in handsets such as the [HTC Hero](http://en.wikipedia.org/wiki/HTC_Hero). The feature was originally disabled at the kernel level (possibly to avoid infringing Apple's patents on touch-screen technology at the time). Google has since released an update for the [Nexus One](http://en.wikipedia.org/wiki/Nexus_One) and the [Motorola Droid](http://en.wikipedia.org/wiki/Motorola_Droid) which enables multi-touch natively.

* + **Bluetooth**

Supports [A2DP](http://en.wikipedia.org/wiki/A2DP), [AVRCP](http://en.wikipedia.org/wiki/AVRCP), sending files ([OPP](http://en.wikipedia.org/wiki/Object_Push_Profile)), accessing the phone book ([PBAP](http://en.wikipedia.org/wiki/Bluetooth_profile#Phone_Book_Access_Profile_.28PBAP.2C_PBA.29)), voice dialing and sending contacts between phones. Keyboard, mouse and joystick ([HID](http://en.wikipedia.org/wiki/Bluetooth_profile#Human_Interface_Device_Profile_.28HID.29)) support is available in Android 3.1+, and in earlier versions through manufacturer customizations and third-party applications.

* + **Video calling**

Android does not support native video calling, but some handsets have a customized version of the operating system that supports it, either via the [UMTS](http://en.wikipedia.org/wiki/UMTS) network (like the [Samsung Galaxy S](http://en.wikipedia.org/wiki/Samsung_Galaxy_S)) or over IP. Video calling through Google Talk is available in Android 2.3.4 and later. Gingerbread allows [Nexus S](http://en.wikipedia.org/wiki/Nexus_S) to place Internet calls with a SIP account. This allows for enhanced VoIP dialing to other SIP accounts and even phone numbers. Skype 2.1 offers video calling in Android 2.3, including front camera support.

* + **Multitasking**

Multitasking of applications is available.

* + **Voice based features**

Google search through voice has been available since initial release. Voice actions for calling, texting, navigation, etc. are supported on Android 2.2 onwards.

* + **Tethering**

Android supports [tethering](http://en.wikipedia.org/wiki/Tethering), which allows a phone to be used as a wireless/wired [Wi-Fi hotspot](http://en.wikipedia.org/wiki/Wi-Fi_hotspot). Before Android 2.2 this was supported by third-party applications or manufacturer customizations.

* + **Screen capture**

Android supports capturing a [screenshot](http://en.wikipedia.org/wiki/Screenshot) by pressing the power and volume-down buttons at the same time. Prior to Android 4.0, the only methods of capturing a screenshot were through manufacturer and third-party customizations or otherwise by using a PC connection (DDMS developer's tool). These alternative methods are still available with the latest Android.

* + **External storage**

Most Android devices include microSD slot and can read microSD cards formatted with [FAT32](http://en.wikipedia.org/wiki/FAT32), [Ext3fs](http://en.wikipedia.org/wiki/Ext3fs) or [Ext4fs](http://en.wikipedia.org/wiki/Ext4fs) file system. To allow use of high-capacity storage media such as [USB flash drives](http://en.wikipedia.org/wiki/USB_flash_drive) and [USB HDDs](http://en.wikipedia.org/wiki/USB_HDD), many Android tablets also include [USB](http://en.wikipedia.org/wiki/USB) 'A' receptacle. Storage formatted with [FAT32](http://en.wikipedia.org/wiki/FAT32) is handled by [Linux Kernel](http://en.wikipedia.org/wiki/Linux_Kernel) VFAT driver, while 3rd party solutions are required to handle other popular file systems such as [NTFS](http://en.wikipedia.org/wiki/NTFS), [HFS Plus](http://en.wikipedia.org/wiki/HFS%2B) and [exFAT](http://en.wikipedia.org/wiki/ExFAT" \o "ExFAT).

**4.3. USES**

While Android is designed primarily for Smartphones and tablets, the open and customizable nature of the operating system allows it to be used on other electronics, including [laptops](http://en.wikipedia.org/wiki/Laptop) and [netbooks](http://en.wikipedia.org/wiki/Netbook" \o "Netbook), [smart books](http://en.wikipedia.org/wiki/Smartbook), and [ebook readers](http://en.wikipedia.org/wiki/Ebook_reader" \o "Ebook reader). Further, Google intends to bring Android to televisions with [Google TV](http://en.wikipedia.org/wiki/Google_TV), and the OS has seen niche applications on [wristwatches](http://en.wikipedia.org/wiki/Wristwatch), [headphones](http://en.wikipedia.org/wiki/Headphones), car CD and DVD players, [digital cameras](http://en.wikipedia.org/wiki/Digital_cameras), [portable media players](http://en.wikipedia.org/wiki/Portable_media_player) and [landlines](http://en.wikipedia.org/wiki/Landlines).

The first commercially available phone to run Android was the [HTC Dream](http://en.wikipedia.org/wiki/HTC_Dream), released on 22 October 2008. In early 2010 Google collaborated with [HTC](http://en.wikipedia.org/wiki/HTC) to launch its flagship Android device, the [Nexus One](http://en.wikipedia.org/wiki/Nexus_One). This was followed later in 2010 with the [Samsung](http://en.wikipedia.org/wiki/Samsung)-made [Nexus S](http://en.wikipedia.org/wiki/Nexus_S) and in 2011 with the [Galaxy Nexus](http://en.wikipedia.org/wiki/Galaxy_Nexus).

**CHAPTER 5**

**PROJECT DESCRIPTION**

**5.1. MODULE DESCRIPTIONS**

There are three modules in this project, which are processing makes the location based alerting system an efficient and flexible application. Here the modules processed are working and performing tasks in easiest way. The followings are the modules of the project.

**MODULES**

* SMS Sending Module
* Alarm Module
* Maps Module

**SMS SENDING MODULE**

This module gets the phone number as input from the user and sends the current location of the user to that phone number using network carrier. The text message function is manually started and stopped using buttons in the user interface. This modules is responsible for the automated communication between the user and the person at the destination location using text messages. Network carrier charges apply for this process and the user will be charged for the messaging services being used.

**ALARM MODULE**

This module is used to call the in-built alarm clock for setting time for the event. The alarm module helps in invoking the alarm intent for starting the event, thereby giving the user choices to send message and navigation services. It also provides options for the user to set different tones for the alarm, change volume and type of the alarm.

**MAPS MODULE**

This module is used to select the destination location and get directions to that location from the user’s current location. The open source application Google Maps intent is invoked for this process. The map helps the user in giving location input and getting directions, thereby serving both purposes effectively.

**5.2. DATA FLOW DIAGRAMS**

**5.2.1. ARCHITECTURAL DIAGRAM**

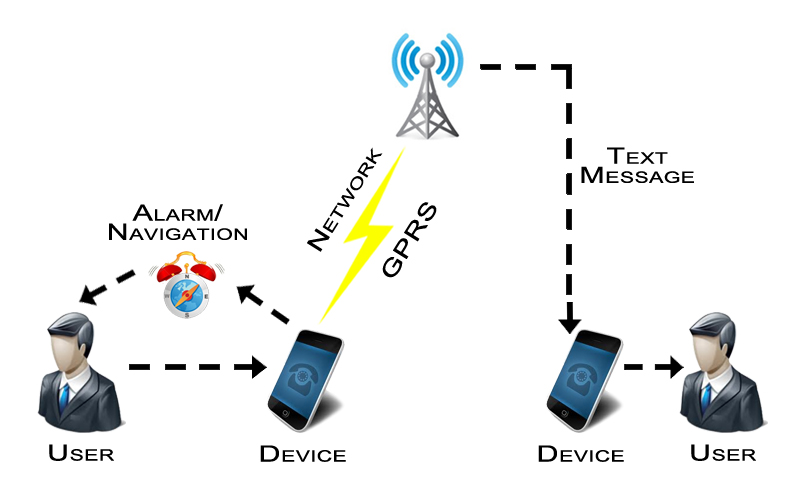
****

Figure 5.1 : Application Architecture Diagram

The architectural diagram shows the basic architecture of the system. Here the user uses the android device to connect to network and send information to the other user. The operations involved are alarm, text messages and navigation using GPRS/GPS technologies.

**5.2.2. FLOW CHART**

ADD EVENT

Start Alarm

GPS Tracking /

Sending Location

View Current location

Text Message sent to the person

Alert the user when the location is reached.

Get navigation details to destination location

Figure 5.2 : Flow Chart

The flow chart represents the direction of the flow of data within the system and the corresponding operations that are carried out. It describes the different stages of execution of the application from user inputs to the output given by the system.

**5.3 UML DIAGRAMS**

**5.3.1. USE CASE DIAGRAM**

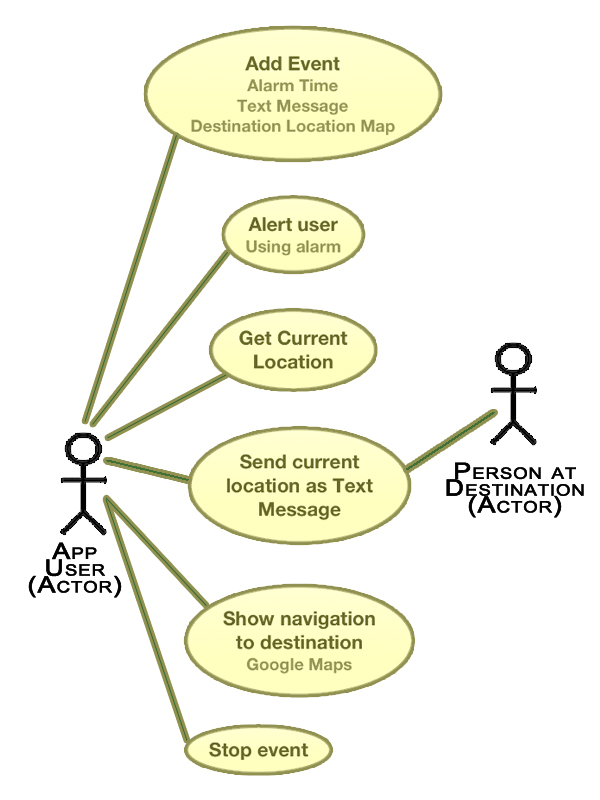


Figure 5.3 : Usecase Diagram

The use case diagram is the simplest representation of the user’s interaction with the system and depicting the specifications of a use case. Here the application user interacts with the application through various inputs and uses the system to send text messages of his location to another user.

**5.3.2. CLASS DIAGRAM**

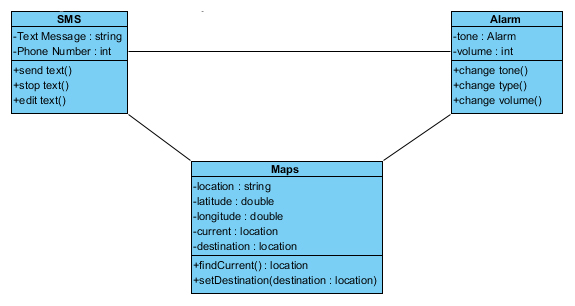


Figure 5.4 : Class Diagram

A class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among the classes. Here three class namely SMS, Alarm and Maps are used to describe the structure of the application. The various attributes and operations are also specified.

**5.3.3. ACTIVITY DIAGRAM**

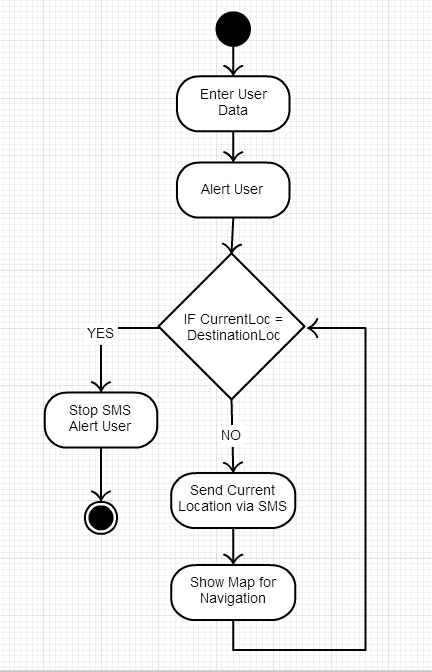
****

Figure 5.5 : Activity Diagram

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the Unified Modeling Language, activity diagrams can be used to describe the business and operational step-by-step workflows of components in a system. An activity diagram shows the overall flow of control.

**5.3.4. SEQUENCE DIAGRAM**

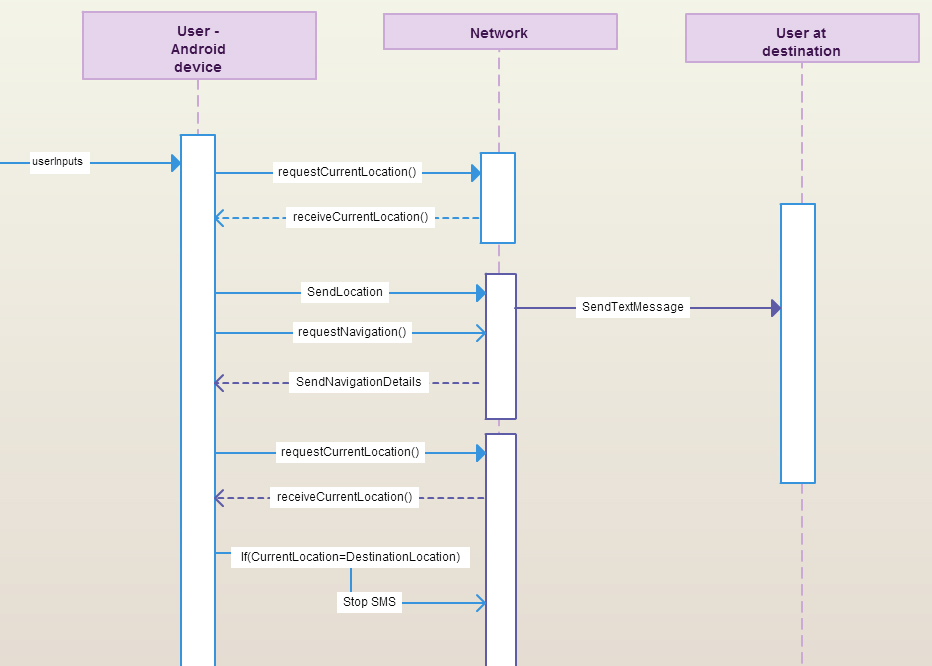


Figure 5.6 : Sequence Diagram

A sequence diagram is a kind of interaction diagram that shows how processes operate with one another and in what order. A sequence diagram shows object interactions arranged in time sequence. It depicts the objects and classes involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario. There are three objects – user android device, network and user at destination location. The interactions between these objects are shown by the messages and the activity boxes on top of the lifelines to represent the processes that are being performed.

**5.3.5. COLLABORATION DIAGRAM**

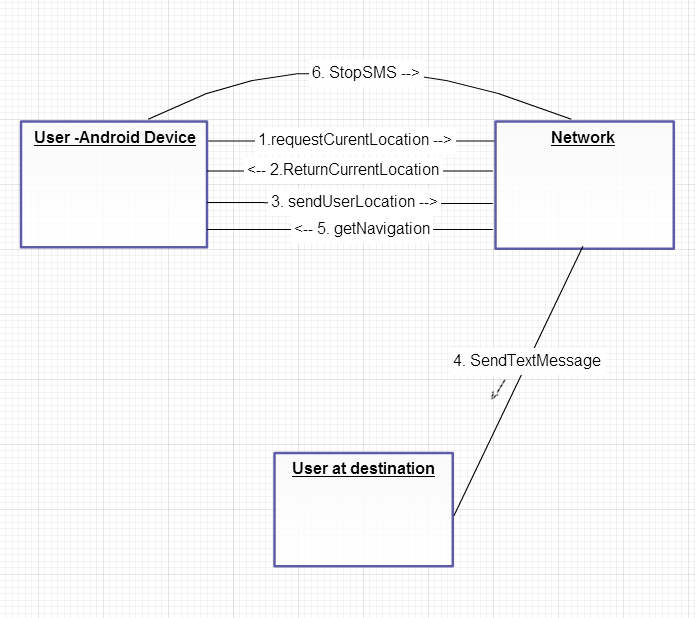


Figure 5.7 : Colloboration Diagram

A Collaboration diagram models the interactions between objects or parts in terms of sequenced messages. Collaboration diagrams represent a combination of information taken from Class, Sequence, and Use Case Diagrams describing both the static structure and dynamic behavior of a system.

**CHAPTER 6**

**SYSTEM IMPLEMENTATION**

Proper implementation is essential to provide a reliable system to meet the organization requirements. Successful implementation may not guarantee improvement in the organization using the new system, but proper installation will improve it. The implementation needs for easy and quick using and time saving programs.

The three aspects of implementation

* Training personnel
* Conversion procedures
* Post-implementation review

**6.1. CONCLUSIONS**

The advantage of the application is the options available for texting the user location automatically and also using navigator for proper driving directions. Using this system, one can maintain road safety. The system can also help the user to keep the person at the destination informed about the current location. This system can still be improved by identifying more user needs and satisfying them all within the same application. The utilization rate of many features such as SMS, GPRS, GPS and other high end applications will directly affects the durability of mobile phone batteries.

**6.2. FUTURE ENHANCEMENT**

Smartphone usage is increasing drastically. While the use of GPRS and GPS in some phones has been slow to begin, the technology is gaining some traction in the Smartphone market. Many Android, Nokia, Blackberry handsets, and the [Nintendo 3DS](http://en.wikipedia.org/wiki/Nintendo_3DS), come with lots of latest features installed. Location based alarm software are available for most mobile platforms. Moreover, there are a number of online applications that enable users to meet their needs effectively. In the future, such applications can be further developed to have much more integrated features with low cost and lesser usage of mobile data.

**APPENDICES A : SAMPLE SOURCE CODE**

**SMS Sending**

Text messages are sent using the network carrier. The Coding to send text message to a mobile number in android is given below.

SmsManager sms=SmsManager.*getDefault*();

sms.sendTextMessage(num, **null**, txt, **null**, **null**);

Here txt refers to the message and num is the phone num to which the message is sent.

**package** com.example.hereicome;

**import** android.app.Activity;

**import** android.content.Context;

**import** android.location.Location;

**import** android.location.LocationListener;

**import** android.location.LocationManager;

**import** android.os.Bundle;

**import** android.telephony.SmsManager;

**import** android.text.format.Time;

**import** android.widget.TextView;

**import** android.widget.Toast;

**public** **class** smsAlert **extends** Activity {

**public** **class** Mylocation **implements** LocationListener {

@Override

**public** **void** onLocationChanged(Location location) {

Toast.*makeText*(getApplicationContext(), "latitude: "+location.getLatitude()+" longitude: "+location.getLongitude(),Toast.*LENGTH\_LONG*).show();

today = **new** Time(Time.*getCurrentTimezone*());

today.setToNow();

time=today.format("%k:%M:%S");

date=today.format("%h-%d-%Y");

Toast.*makeText*(getApplicationContext(),"Time: "+time+" Date: "+date,Toast.*LENGTH\_LONG*).show();

**double** lat=location.getLatitude();

**double** lon=location.getLongitude();

**try** {

JSONObject json= JSONfunctions.*getJSONfromURL* ("http://maps.googleapis.com/maps/api/geocode/json?latlng="+lat+","+lon+"&sensor=false");

contacts = json.getJSONArray("results");

**for**(**int** i = 0; i < 1; i++) {

JSONObject c = contacts.getJSONObject(0);

String id = c.getString("formatted\_address");

da=time+" "+date;

SmsManager sms=SmsManager.*getDefault*();

sms.sendTextMessage(num, **null**, id+"\n"+da,**null**, **null**);

Toast.*makeText*(smsAlert.**this**,"Message Sent", 300).show();

t1.setText(id+"\n"+da);

}

}

**catch** (JSONException e) {

e.printStackTrace();

Toast.*makeText*(smsAlert.**this**,"Error"+e, 300).show();

}

}

@Override

**public** **void** onProviderDisabled(String arg0) {

}

@Override

**public** **void** onProviderEnabled(String arg0) {

}

@Override

**public** **void** onStatusChanged(String arg0, **int** arg1, Bundle arg2) {

}

}

String num;TextView t1;

LocationManager lm;LocationListener ll;

Time today;

String time,date;

String da;

JSONArray contacts = **null**;

**boolean** isGPSEnabled = **false**;

**boolean** isNetworkEnabled = **false**;

**boolean** canGetLocation = **false**;

@Override

**protected** **void** onCreate(Bundle savedInstanceState) {

**super**.onCreate(savedInstanceState);

setContentView(R.layout.*activity\_main*);

Bundle d=**new** Bundle();

d=getIntent().getExtras();

num=d.getString("num");

t1=(TextView)findViewById(R.id.*textView1*);

lm=(LocationManager)getSystemService(Context.*LOCATION\_SERVICE*);

ll=**new** Mylocation();

isGPSEnabled = lm

.isProviderEnabled(LocationManager.*GPS\_PROVIDER*);

isNetworkEnabled = lm

.isProviderEnabled(LocationManager.*NETWORK\_PROVIDER*);

**if** (!isGPSEnabled && !isNetworkEnabled) {

}

**else** {

**this**.canGetLocation = **true**;

**if** (isNetworkEnabled) {

lm.requestLocationUpdates(LocationManager.*NETWORK\_PROVIDER*,1000,0, ll);

}

**if** (isGPSEnabled) {

lm.requestLocationUpdates(LocationManager.*GPS\_PROVIDER*, 1000,0, ll);}

}

}

}

**Alarm Clock**

The Alarm Clock module is used to invoke the existing Alarm intent for setting alarm at a user specific clock time. The coding to invoke the alarm clock is given below.

**package** com.example.hereicome;

**import** java.util.Calendar;

**import** android.os.Bundle;

**import** android.app.Activity;

**import** android.app.AlarmManager;

**import** android.app.PendingIntent;

**import** android.app.TimePickerDialog;

**import** android.app.TimePickerDialog.OnTimeSetListener;

**import** android.content.Context;

**import** android.content.Intent;

**import** android.view.Menu;

**import** android.view.View;

**import** android.view.View.OnClickListener;

**import** android.widget.Button;

**import** android.widget.TextView;

**import** android.widget.TimePicker;

**public** **class** MainActivity **extends** Activity {

TimePicker myTimePicker;

Button buttonstartSetDialog;

TextView textAlarmPrompt;

TimePickerDialog timePickerDialog;

**final** **static** **int** *RQS\_1* = 1;

@Override

**protected** **void** onCreate(Bundle savedInstanceState) {

**super**.onCreate(savedInstanceState);

setContentView(R.layout.*activity\_main*);

textAlarmPrompt = (TextView)findViewById(R.id.*alarmprompt*);

buttonstartSetDialog = (Button)findViewById(R.id.*startSetDialog*);

buttonstartSetDialog.setOnClickListener(**new** OnClickListener(){

**public** **void** onClick(View v) {

textAlarmPrompt.setText("");

openTimePickerDialog(**false**);

}

});

}

**private** **void** openTimePickerDialog(**boolean** is24r){

Calendar calendar = Calendar.*getInstance*();

timePickerDialog = **new** TimePickerDialog(

MainActivity.**this**,

onTimeSetListener,

calendar.get(Calendar.*HOUR\_OF\_DAY*),

calendar.get(Calendar.*MINUTE*),

is24r);

timePickerDialog.setTitle("Set Alarm Time");

timePickerDialog.show();

}

OnTimeSetListener onTimeSetListener=**new** OnTimeSetListener(){

@Override

**public** **void** onTimeSet(TimePicker view, **int** hourOfDay, **int** minute) {

Calendar calNow = Calendar.*getInstance*();

Calendar calSet = (Calendar) calNow.clone();

calSet.set(Calendar.*HOUR\_OF\_DAY*, hourOfDay);

calSet.set(Calendar.*MINUTE*, minute);

calSet.set(Calendar.*SECOND*, 0);

calSet.set(Calendar.*MILLISECOND*, 0);

**if**(calSet.compareTo(calNow) <= 0){

calSet.add(Calendar.*DATE*, 1);

}

setAlarm(calSet);

}};

**private** **void** setAlarm(Calendar targetCal){

textAlarmPrompt.setText(

"\n\n\n"

+ "Alarm is set at " + targetCal.getTime() + "\n"

+ "\n");

Intent intent = **new** Intent(getBaseContext(), AlarmReceiver.**class**);

PendingIntent pendingIntent = PendingIntent.*getBroadcast*(getBaseContext(), *RQS\_1*, intent, 0);

AlarmManager alarmManager = (AlarmManager)getSystemService(Context.*ALARM\_SERVICE*);

alarmManager.set(AlarmManager.*RTC\_WAKEUP*, targetCal.getTimeInMillis(), pendingIntent);

}

@Override

**public** **boolean** onCreateOptionsMenu(Menu menu) {

getMenuInflater().inflate(R.menu.*activity\_main*, menu);

**return** **true**;

}

}

**Maps**

The Maps module uses the open source application Google Maps to display the map and get directions from the current user location to the destination. Its also fetches the latitude and longitude coordinates accurately for the same purpose. The coding for maps is given below.

**package** com.example.hereicome;

**import** android.content.Context;

**import** android.content.Intent;

**import** android.location.Location;

**import** android.location.LocationListener;

**import** android.location.LocationManager;

**import** android.os.Bundle;

**import** android.telephony.SmsManager;

**import** android.text.format.Time;

**import** android.view.GestureDetector;

**import** android.view.MotionEvent;

**import** android.widget.Toast;

**import** com.google.android.maps.GeoPoint;

**import** com.google.android.maps.MapActivity;

**import** com.google.android.maps.MapView;

**import** com.google.android.maps.Overlay;

**public** **class** Map1 **extends** MapActivity{

**double** lat,lon;String time,date;

String da;

JSONArray contacts = **null**;

**public** **class** MyLoc **implements** LocationListener {

@Override

**public** **void** onLocationChanged(Location location) {

String coordinates[]={""+location.getLatitude(),""+location.getLongitude()};

lat=Double.*parseDouble*(coordinates[0]);

lon=Double.*parseDouble*(coordinates[1]);

Toast.*makeText*(getBaseContext(), lat + "," +lon , 3000).show();

Time today = **new** Time(Time.*getCurrentTimezone*());

today.setToNow();

time=today.format("%k:%M:%S");

date=today.format("%h-%d-%Y");

Toast.*makeText*(getApplicationContext(),"Time: "+time+" Date: "+date,Toast.*LENGTH\_LONG*).show();

**double** lat=location.getLatitude();

**double** lon=location.getLongitude();

**try** {

JSONObject json = JSONfunctions.*getJSONfromURL*("http://maps.googleapis.com/maps/api/geocode/json?latlng="+lat+","+lon+"&sensor=false");

contacts = json.getJSONArray("results");

**for**(**int** i = 0; i < 1; i++){

JSONObject c = contacts.getJSONObject(0);

String id = c.getString("formatted\_address");

da=time+" "+date;

SmsManager sms=SmsManager.*getDefault*();

sms.sendTextMessage(num, **null**, id+"\n"+da,**null**, **null**);

Toast.*makeText*(Map1.**this**,"Message Sent", 300).show();

}

}

**catch** (JSONException e) {

e.printStackTrace();

Toast.*makeText*(Map1.**this**,"Error"+e, 300).show();

}

}

@Override

**public** **void** onProviderDisabled(String provider) {

}

@Override

**public** **void** onProviderEnabled(String provider) {

}

@Override

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

}

}

MapView m; GestureDetector mGestureDetector;LocationManager lm;

LocationListener ll;

**boolean** isGPSEnabled = **false**;

String num;

**boolean** isNetworkEnabled = **false**;

**boolean** canGetLocation = **false**;

@Override

**protected** **void** onCreate(Bundle icicle) {

**super**.onCreate(icicle);

setContentView(R.layout.*map1*);

num=getIntent().getExtras().getString("loc");

Toast.*makeText*(Map1.**this**, num, Toast.*LENGTH\_LONG*).show();

lm=(LocationManager)getSystemService(Context.*LOCATION\_SERVICE*);

ll=**new** MyLoc();

isGPSEnabled = lm

.isProviderEnabled(LocationManager.*GPS\_PROVIDER*);

isNetworkEnabled = lm

.isProviderEnabled(LocationManager.*NETWORK\_PROVIDER*);

**if** (!isGPSEnabled && !isNetworkEnabled) {

}

**else** {

**this**.canGetLocation = **true**;

**if** (isNetworkEnabled) {

lm.requestLocationUpdates(LocationManager.*NETWORK\_PROVIDER*,120000,0, ll);

}

**if** (isGPSEnabled) {

lm.requestLocationUpdates(LocationManager.*GPS\_PROVIDER*, 120000,0, ll);

}

}

m=(MapView)findViewById(R.id.*mapview*);

m.setBuiltInZoomControls(**true**);

m.setSatellite(**true**);

m.getOverlays().add(**new** Overlay() {

@Override

**public** **boolean** onTouchEvent(MotionEvent event, MapView mapView) {

mGestureDetector.onTouchEvent(event);

**return** **super**.onTouchEvent(event, mapView);

}

});

mGestureDetector = **new** GestureDetector(**new** GestureDetector.SimpleOnGestureListener() {

@Override

**public** **boolean** onDoubleTap(MotionEvent e) {

GeoPoint p = m.getProjection().fromPixels((**int**)e.getX(), (**int**)e.getY());

Toast.*makeText*(getBaseContext(), p.getLatitudeE6() / 1E6 + "," + p.getLongitudeE6() /1E6 , 3000).show();

Intent i=**new** Intent(Map1.**this**,MapAct.**class**);

i.putExtra("latd", p.getLatitudeE6()/1E6);

i.putExtra("lond", p.getLongitudeE6()/1E6);

i.putExtra("lat", lat);

i.putExtra("lon", lon);

startActivity(i);

**return** **super**.onDoubleTap(e);

}

});

}

**protected** **boolean** isRouteDisplayed() {

**return** **false**;

}

}

**import** android.location.LocationListener;

**import** android.location.LocationManager;

**import** android.net.Uri;

**import** android.os.Bundle;

**import** android.app.Activity;

**import** android.content.Intent;

**public** **class** MapAct **extends** Activity {

**double** s,s1,lat,lon;

LocationManager lm;

LocationListener ll;

@Override

**public** **void** onCreate(Bundle savedInstanceState) {

**super**.onCreate(savedInstanceState);

setContentView(R.layout.*activity\_map*);

lat=getIntent().getExtras().getDouble("lat");

lon=getIntent().getExtras().getDouble("lon");

s=getIntent().getExtras().getDouble("latd");

s1=getIntent().getExtras().getDouble("lond");

Intent i=**new** Intent(android.content.Intent.*ACTION\_VIEW*,Uri.*parse*("http://maps.google.com/maps?saddr="+lat+","+lon+"&daddr="+s+","+s1));

startActivity(i);

}

}

**User Input**

The user inputs such as Event Name, Contact Name, Phone number, Alarm Time, Date and Destination Location are fetched from the user using these functions. The main activity is written in Android xml format and the functions are specified in java language. The code for the user inputs is given below.

**package** com.example.hereicome;

**import** android.os.Bundle;

**import** android.app.Activity;

**import** android.content.Intent;

**import** android.view.Menu;

**import** android.view.View;

**import** android.view.View.OnClickListener;

**import** android.widget.Button;

**import** android.widget.TextView;

**public** **class** MainActivity **extends** Activity **implements** OnClickListener {

Button b1,b2;TextView t1;

@Override

**protected** **void** onCreate(Bundle savedInstanceState) {

**super**.onCreate(savedInstanceState);

setContentView(R.layout.*activity\_main*);

b1=(Button)findViewById(R.id.*button1*);

b2=(Button)findViewById(R.id.*button2*);

t1=(TextView)findViewById(R.id.*textView1*);

b1.setOnClickListener(**this**);

b2.setOnClickListener(**this**);

Bundle extras=getIntent().getExtras();

String value1 = extras.getString("tag\_person\_name");

String value2 = extras.getString("tag\_person\_pin");

String value3 = extras.getString("e1");

String value4 = extras.getString("tag\_person\_pin3");

t1.setText(value1+"\n"+value2+"\n"+value3+"\n"+value4);

}

@Override

**public** **boolean** onCreateOptionsMenu(Menu menu) {

getMenuInflater().inflate(R.menu.*activity\_main*, menu);

**return** **true**;

}

@Override

**public** **void** onClick(View v) {

**switch**(v.getId()){

**case** R.id.*button1*:

Intent i=**new** Intent(MainActivity.**this**,Enter2.**class**);

startActivity(i);

**break**;

**case** R.id.*button2*:

t1.setText("");

**break**;

}

}

**package** com.example.hereicome;

**import** android.content.ContentValues;

**import** android.content.Context;

**import** android.database.Cursor;

**import** android.database.sqlite.SQLiteDatabase;

**import** android.database.sqlite.SQLiteOpenHelper;

**import** android.util.Log;

**public** **class** PersonDatabaseHelper {

**private** **static** **final** String *TAG* = PersonDatabaseHelper.**class**.getSimpleName();

**private** **static** **final** **int** *DATABASE\_VERSION* = 1;

**private** **static** **final** String *DATABASE\_NAME* = "mydatabaseDB";

**private** **static** **final** String *TABLE\_NAME* = "meeting"; **private** **static** **final** String *PERSON\_TABLE\_COLUMN\_ID* = "\_id";**private** **static** **final** String *PERSON\_TABLE\_COLUMN\_NAME* = "person\_name";

**private** **static** **final** String *PERSON\_TABLE\_COLUMN\_PERSON* = "person\_pin";

**private** **static** **final** String *PERSON\_TABLE\_COLUMN\_PLACE* = "person\_place";

**private** **static** **final** String *PERSON\_TABLE\_COLUMN\_DATE* = "person\_date";

**private** **static** **final** String *PERSON\_TABLE\_COLUMN\_NO* = "person\_no";

**private** DatabaseOpenHelper openHelper;

**private** SQLiteDatabase database;

**public** PersonDatabaseHelper(Context aContext) {

openHelper = **new** DatabaseOpenHelper(aContext);

database = openHelper.getWritableDatabase();

}

**public** **void** insertData (String aPersonName, String aPersonPin, String aPersonPlace, String aPersonDate, String aPersonno ) {

ContentValues contentValues = **new** ContentValues();

contentValues.put(*PERSON\_TABLE\_COLUMN\_NAME*, aPersonName);

contentValues.put(*PERSON\_TABLE\_COLUMN\_PERSON*, aPersonPin);

contentValues.put(*PERSON\_TABLE\_COLUMN\_PLACE*, aPersonPlace);

contentValues.put(*PERSON\_TABLE\_COLUMN\_DATE*, aPersonDate);

contentValues.put(*PERSON\_TABLE\_COLUMN\_NO*, aPersonno);

database.insert(*TABLE\_NAME*, **null**, contentValues);

}

**public** Cursor getAllData () {

String buildSQL = "SELECT \* FROM " + *TABLE\_NAME*;

Log.*d*(*TAG*, "getAllData SQL: " + buildSQL);

**return** database.rawQuery(buildSQL, **null**);

}

**private** **class** DatabaseOpenHelper **extends** SQLiteOpenHelper {

**public** DatabaseOpenHelper(Context aContext) {

**super**(aContext, *DATABASE\_NAME*, **null**, *DATABASE\_VERSION*);

}

@Override

**public** **void** onCreate(SQLiteDatabase sqLiteDatabase) {

String buildSQL = "CREATE TABLE " + *TABLE\_NAME* + "( " + *PERSON\_TABLE\_COLUMN\_ID* + " INTEGER PRIMARY KEY, " +

*PERSON\_TABLE\_COLUMN\_NAME* + " TEXT, " + *PERSON\_TABLE\_COLUMN\_PERSON* + " TEXT, " + *PERSON\_TABLE\_COLUMN\_PLACE* + " TEXT, " + *PERSON\_TABLE\_COLUMN\_DATE* + " TEXT, " + *PERSON\_TABLE\_COLUMN\_NO* + " TEXT )";

Log.*d*(*TAG*, "onCreate SQL: " + buildSQL);

sqLiteDatabase.execSQL(buildSQL);

}

@Override

**public** **void** onUpgrade(SQLiteDatabase sqLiteDatabase, **int** oldVersion, **int** newVersion) {

String buildSQL = "DROP TABLE IF EXISTS " + *TABLE\_NAME*;

Log.*d*(*TAG*, "onUpgrade SQL: " + buildSQL);

sqLiteDatabase.execSQL(buildSQL);

onCreate(sqLiteDatabase);

}

}

}

**APPENDICES B : SCREENSHOTS**

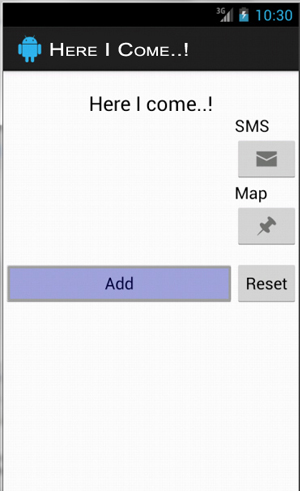


Figure B.i : Application Home

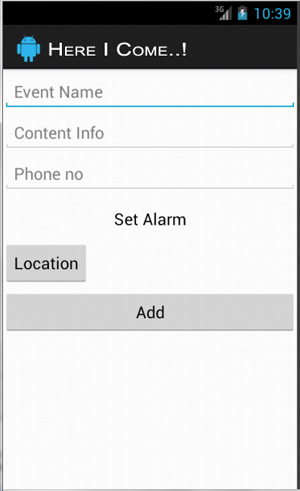


Figure B.ii : Add Event

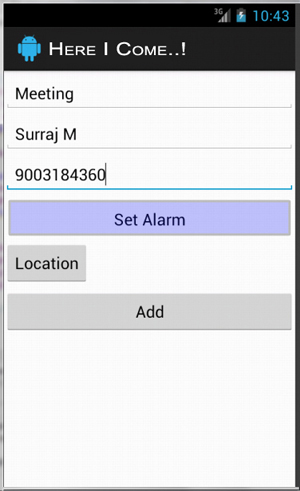


Figure B.iii : Entering Data

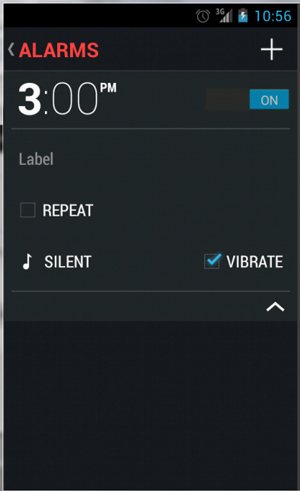


Figure B.iv : Setting Alarm

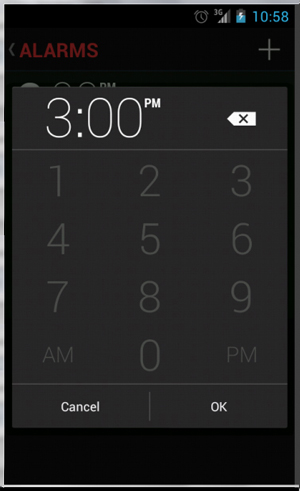


Figure B.v : Change Alarm Time

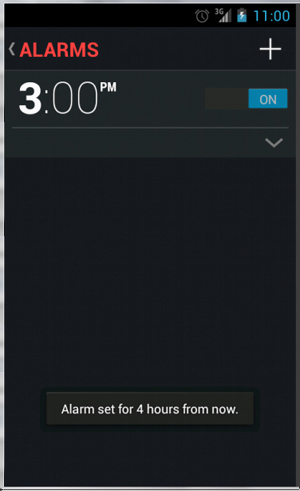


Figure B.vi : Alarm is set ON

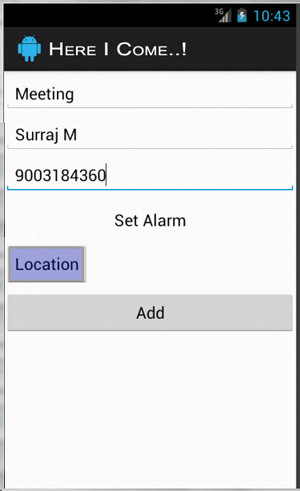


Figure B.vii : Location Setting

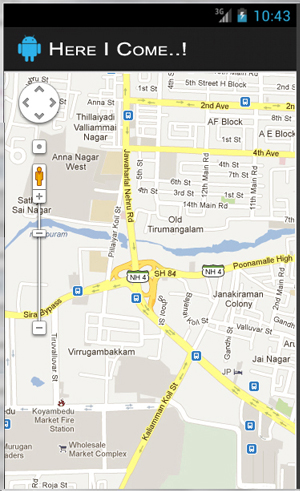


Figure B.viii : Map View

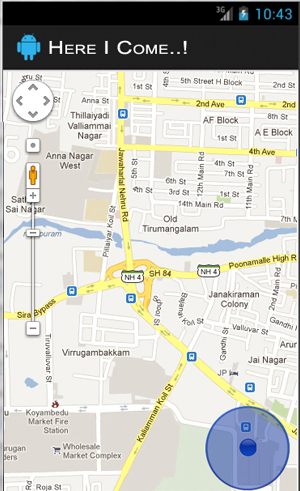


Figure B.ix : Current Location

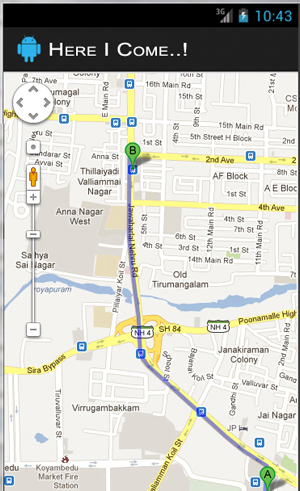


Figure B.x : Directions From A to B

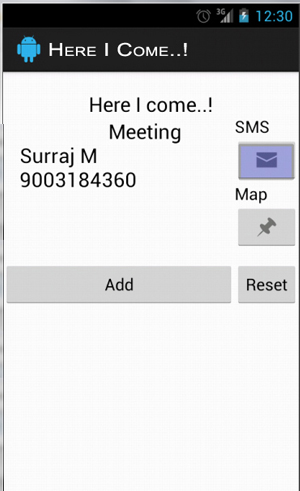


Figure B.xi : Send SMS

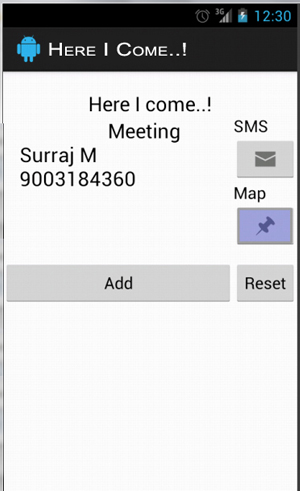


Figure B.xii : View Location and Navigation

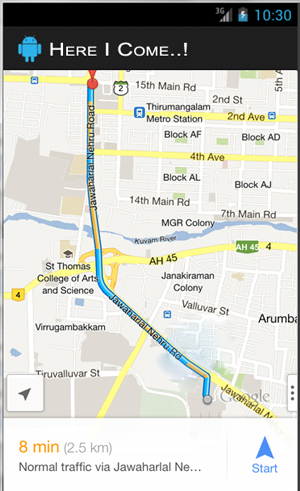


Figure B.xiii : Navigation

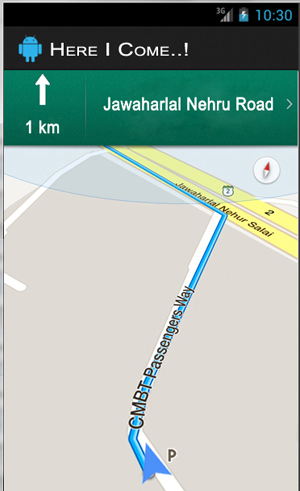


Figure B. xiv : Driving Directions 1

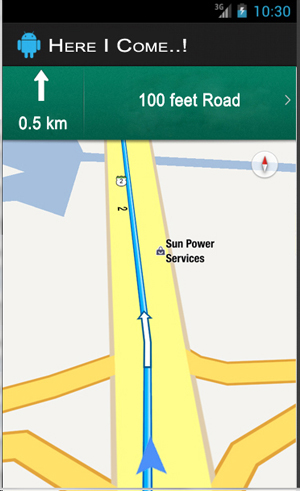


Figure B.xv : Driving Directions 2

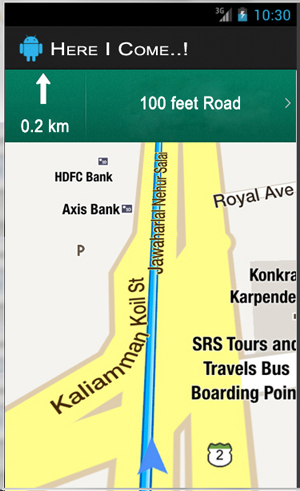


Figure B.xvi : Driving Directions 3

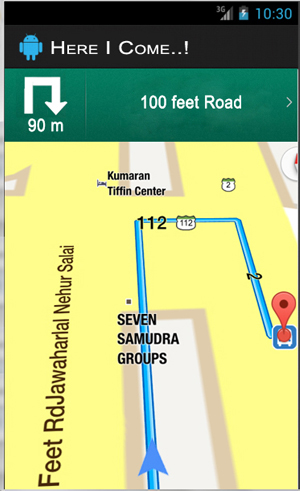


Figure B.xvii : Driving Directions 4

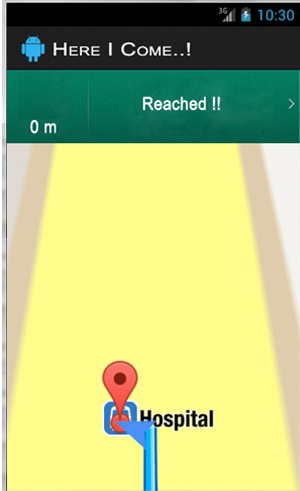


Figure B.xviii : Driving Directions : Reached!

**BIBLIOGRAPHY**

[1] Gu Xiaoyu. Over 60 per hundred people has a mobile phone in China.

Beijing Times, 2011.2

[2] Analysis think tank. Reached 288 million mobile Internet

users in 4th quarter of 2010. Net ease reported, 2011.2.

[3] Cui Ping. Marketing science. Machinery Industry Press, 2005.

[4] Li Tianwen. Principles and applications of GPS. Science Press, 2007

[5] Android (operating system) updated, 25 April 2013.

<http://en.wikipedia.org/wiki/Android_(operating_system)>

[6] Android Developers Site <http://developer.android.com/index.html>