**Mini Project Report On**

**ANTI -THEFT MOBILE APPLICATION**

**Submitted by**

**M.Rahul Narayana (13B81A1265)**

**L.Varun (13B81A12B0)**

**P.V.Saideep (13B81A12B4)**

**Under the esteemed guidance of**

**Mrs. N. PAVANI, Sr. Associate Professor**

**IT Department**

**CVR College of Engineering**



**DEPARTMENT OF INFORMATION TECHNOLOGY**

**CVR COLLEGE OF ENGINEERING**

(An Autonomous Institution, Accredited by NBA, NAAC ’A’ Grade, Affiliated to JNTUH)

Mangalpalli(V), Ibrahimpatnam-501 510

**2016-2017**



**CVR COLLEGE OF ENGINEERING**

(An Autonomous Institution)

**ACCREDITED BY NATIONAL BOARD OF ACCREDITATION, AICTE**

(Approved by AICTE & Govt of Telangana and Affiliated to JNTUH) Vastunagar, Mangalpally (V), Ibrahimpatnam (M), R.R District, PIN – 501510

Web :  [http://cvr.ac.i](http://cvr.ac.in/)n, email:  [info@cvr.ac.i](mailto:info@cvr.ac.in)n

Ph : 08414-252222, 252369, Office Telefax : 252396, Principal : 252396 (O)

**CERTIFICATE**

This is to certify that the Project Report entitled “**Anti-theft Mobile Application”** is a bonafide record of work carried out by **M.Rahul Narayana (13B81A1265),** **L.Varun (13B81A12B0)** and **P.V.Saideep (13B81A12B4)** under my guidance and supervision in partial fulfillment of the requirements for the award of Bachelor of Technology degree in Information Technology of Jawaharlal Nehru Technological University, Hyderabad during the academic year 2016-2017.

**SIGNATURE OF INTERNAL GUIDE SIGNATURE OF HOD &**

**Mrs. N. PAVANI PROJECT COORDINATOR**

**Sr. ASSOCIATE PROFESSOR Dr. BIPIN BIHARI JAYASINGH**

**IT DEPARTMENT IT DEPARTMENT**

**CVR COLLEGE OF ENGINEERING CVR COLLEGE OF ENGINEERING**

**DECLARATION**

We hereby declare that the project entitled **“ANTI-THEFT MOBILE APPLICATION”** submitted by us to **CVR College of Engineering** in partial fulfillment of the requirement for the award of degree of **B.Tech in INFORMATION TECHNOLOGY** is a record of bonafide project work carried out by us under the guidance of **Mrs. N. Pavani, Sr.Associate Professor, Information Technology.** We further declare that the work reported in this project has not been submitted and will not be submitted, either in part or in full, for the award of any other degree or diploma in this institute or in any other institute or university.

M.Rahul Narayana(13B81A1265)

L.Varun(13B81A12B0)

P.V.Saideep(13B81A12B4)

**ACKNOWLEDGEMENT**

The satisfaction of completing this project would be incomplete without mentioning our thanks to all the people who have supported us with constant guidance and encouragement which have been instrumental in its completion.

We offer our sincere gratitude to Mrs. **N.Pavani,** Sr.Associate Professor, Department of Information Technology, CVR College of Engineering for her immense support, timely co-operation and valuable advice throughout the course of our project work.

We are thankful to **Dr. Bipin Bihari JayaSingh**, Project Coordinator and Head Of Department of Information Technology, CVR College of Engineering for his supportive guidelines and for having provided the necessary support to carry forward this project without any obstacles and hindrances.

M. Rahul Narayana(13B81A1265)

L.Varun(13B81A12B0)

P.V. Saideep(13B81A12B4)

**ABSTRACT**

Mobile Anti-Theft Application (MAT) is an application based on Android used for tracing back stolen or lost mobile. Once our MAT is installed in a mobile phone, an alternate number is fed into the application. The owners SIM mobile number gets registered in the database. Whenever phone is rebooted, (MAT) is invoked in stealth mode and verifies whether if the current SIM number present in mobile phone is of owner. If the SIM is changed and does not matches with the number saved in the database then, MAT sends a message to the alternative mobile number (friends/relatives number) provided during registration process. Since our system is based on GOOGLE Android operating system our system would send the complete address (postal address) as to where the mobile is.

**TABLE OF CONTENTS**

|  |  |  |
| --- | --- | --- |
| **S. NO** | **TITLE** | **PG.NO** |
| 1 | Introduction | 08 |
| 2. | System Requirement Specification | 12 |
| 3. | Design | 17 |
| 4. | Implementation | 23 |
| 5. | Testing | 41 |
| 6. | Conclusion and Future scope | 46 |
| 7. | Bibliography | 47 |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

CHAPTER – 1

INTRODUCTION

1. **INTRODUCTION**

In today’s world cell phones play a very important role in everyone’s life? It provides multiple options like voice and video conversation, GPS tracking system, multimedia and internet usage. Radio signals are constantly being broadcasted by cell phones and hence can trace a lost mobile. Mobile phone tracking system technology can be deployed in any GSM supported handsets. All cell phones constantly send signals to its nearest towers hence it is been possible to track a lost phone accurately.

Mobile Anti Theft (MAT) is a project which is based on GPS satellite tracking system which is been used by cars and road transport industry from years (Hou Rui, 2012). GSM phone tracking is likely to show the radius in which the phone is located but not the exact place and even accuracy depends on factors like network coverage, landscape and weather conditions also. When we couple GPS with Google maps the postal address of the lost mobile can be found accurately.

* 1. **Aim and Objectives**

Mobile anti theft system is a project which helps us to track the location of the smart phones. It consists of Android client application which will automatically send SMS when SIM card is changed. Position tracker works on GPS (Global Positioning System) and GPRS. When requested MAT will fetch latitude and longitude satellites and send it as a SMS, moreover if mobile is connected to the internet it retrieves the postal address from Google maps.

To recover lost mobile phones thus creating a fear psyche in thieves that they cannot get away with mobile phones which means we can monitor any cell phone and helping those who cannot afford to lose their expensive cell phones. Android is a new and very user friendly operating system for mobile devices which includes key applications, middleware and even uses Linux Kernel modified version.

* 1. **What Androids Are Made Of**

While writing applications on desktop, you are “master of your own domain”. You can launch main window and no of child windows as in dialog boxes. From our viewpoint we are on our own, features which are supported by OS, and mostly unaware of any other programs running at same time. While there is no interaction between other programs we can communicate with MySQL or any other database typically using an API like frameworks atop it. Android has comparably same concepts packaged in a different way and structured for crash-resistant.

**Activities**

Activity can be explained as building block of user interface. We can consider activity as Android analogue for window or dialog in desktop application. It can even be possible that activity not having a user interface, while the code packaged in the structure of services or content providers.

**Content Providers**

For any data stored in a device Content Providers offer a level of abstraction which is accessible by various applications. In android development it encourage us to make our own data which can be accessed by other applications and even build our own content provider which gives you a complete control over how that data can be accessed.

**Intents**

System messages which run inside the device, various applications notification such as hardware changes like SD card inserted, notifications of incoming data like SMS arrived and even application events are called as Intents.

It doesn’t only allow you to respond to such intents but also to initiate other activities or let know when particular event occurs such as suggest WIFI availability when in range.

**Services**

All the above stated Activities, intent receivers and content providers are all short term and can be terminated any time whereas services are intended to run continuously independent of other activities such as play music while using other applications, in here music controlling is no longer available but the service keep running in the background.

* 1. **Stuff at Your Disposal**
* *Network*

Android based devices are generally with Internet ready. We can take benefit of internet as we wish in any level from raw java sockets to built-in Web browser which is based on Webkit.

* *GPS*

Most of the android devices have access to GPS which can tell where the device is exactly located on the earth using Google Maps. GPS also helps in locating the desired location where we want to travel and even shows the places around us where we commonly go in everyday life which makes it easy to travel otherwise can be to locate the device and its movements in case the device is stolen .

* *Phone Services*

Android devices are similar to other phones which are typically used to make calls, send SMS and can be used for multimedia applications to download music anytime anywhere and games and everything else what we expect from a modern telephonic technology.

**1.4 Why Android ?**

Zero start-up cost to begin development with The tool required to develop any android applications is free of cost and Google charge very small fee deal out application in the market.

* *Freedom to innovate*

Android OS is an open source platform which is based on Linux kernel and other open source libraries. Moreover are free to build applications which runs on android devices and even free to extend platform as well.

* *Freedom to collaborate*

Android developers are encouraged to share code with others and they don’t even have to sign an NDA to do this.

* *Multi-platform*

Support Android OS are supported on several different hardware devices including various phones and tablets.

* *Multi-carrier*

Support Android powered smart phones are offered by most of the carrier services.

CHAPTER - 2

SYSTEM REQUIREMENTS AND SPECIFICATIONS

1. **SYSTEM REQUIREMENTS AND SPECIFICATIONS**

**2.1 Purpose of this Document**

The idea of whole document is to represent bounded physical representation of body of information what we designed have the capacity to communicate or not. This document

contains diagrammatic, symbolic and sensory-representational information of the whole

project. This document produces an artifact by collecting and representing information.

This document helps the user to understand about the project in a lucid manner. The documentation gives an overview of functional requirements and data requirements.

This document explains hardware and software Requirements, user view of product use, general constraints and guidelines of a project.

Documentation is nothing but a method of communicating what the whole project is about. Objective, realistic and complete are the important factors kept in mind while writing satisfactory documentation. So adequacy of the project is not based on length, format, complexity or volume. No standardized documentation is described to write for projects. The whole process of implementation, operation of system and development is based on a proper documentation. Documentation helps in different ways like maintaining the project and also reference to the user like at any point any help required about the project, it can be understood by having a look at the documentation. Documentation can also be done by embedding proper comments in the executable code of the modules. Parenthesis, spaces, blank lines along with suitable loops around the comments block were used to improve the readability.

**2.2 General Description**

* **Users and their Characteristics**

Owner of the mobile phone will install our application, start our application enter his alternate number his name and password and then commit it into database. When he loses his mobile he waits for a SMS from MAT containing location or postal address etc.

. Suppose in a scenario wherein the thief changes the SIM of mobile phone as soon as possible and reboots the system. On reboot our system is triggered then intimation is sent to relative.

* **Product Perspective**

MAT would be widely used as people are concerned about their mobile phones. Moreover the software needs very less memory and the package can be downloaded over internet. A little modification along with web interface would make this a tracking application.

**2.3 Overview of Functional Requirements**

Input requirement is a Mobile with GPS and GPRS facility. It should be a Google Android OS mobile. The mobile will have software installed in it. On SIM change it shall alert the alternate number via SMS. And when requested it would send GPS co-ordinates as well as postal address if requested. Everything would happen in stealth mode without notice to the user.

**2.4 Overview of Data Requirements**

MAT has to be fed with IMSI number , alternate phone number and password so that

whenever the phone is rebooted it checks with the one present in database and the one present. If it’s different from the actual present then number of steps involved is discussed below. After fetching GPS co-ordinates from satellite, our system would check if the mobile is connected to internet. If yes then mobile would send those GPS co-ordinates to GOOGLE MAPS and fetch the postal address from Google maps. This postal address would be sent to owners other no. via SMS . There is no ATS system in market which would send complete postal address on being stolen. All other systems would send only GPS co-ordinates.

**2.5 General Constraints, Assumptions Dependencies, Guidelines**

There is no such constraint for development of this project. But still there is constraint which is the connection or interfacing the network between the PC terminal and mobile phone .The major aspect here is as we are dependent on Google map. The entire project will be executed by using GPRS and GPS mechanism as a request for all the operation claimed for the functional aspect of this project. (Elliott D. Kaplan, 2006)

Certain situation is being considered for the project to be in working condition:

When registering a SIM, it should be present in the mobile.

When registering a new SIM, one SMS would be sent before we register a new SIM.

To get the postal address the thief has to have internet enabled on his SIM.

Assumption is that the thief would change the SIM card without formatting the phone, if he formats the phone then our software will not be available and thus can’t help to do so.

We assume that thief’s SIM card has enough currency for sending an SMS.

**2.6 User View of Product**

Once installed the software, user don’t have to reopen the software every time and check the details. Only if the SIM is changed then the software would do the activities in stealth mode without the user notice. Only interface with the system is during registration of SIM in database. Since our system is based on GOOGLE Android operating system our system would send the complete address (postal address) as to where the mobile is which makes it easy to trace the lost mobile. So once the software is installed and details of the user are entered the software would have no contact with the user and it does its work in background without disturbing the user which makes it user friendly.

**Hardware and Software Requirements**

***Hardware Requirements***

* Smart phones with Qualcomm processor, GPS and GPRS support.

***Software Requirements***

* Google Android OS 1.5
* SQLite database

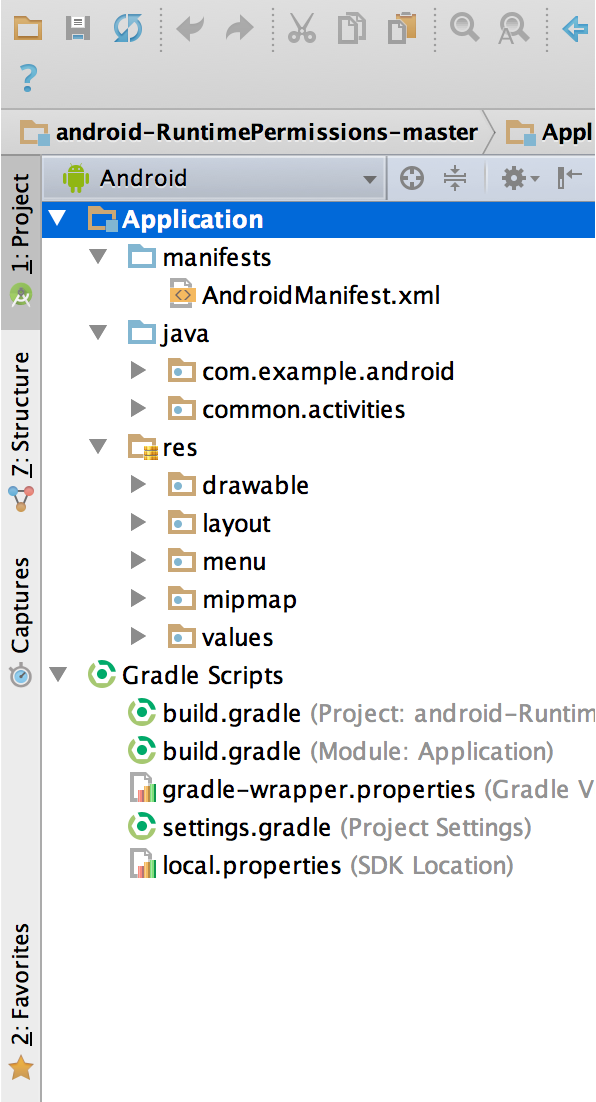
**2.7 TOOLS USED**

**ANDROID STUDIO**

Android Studio is the official Integrated Development Environment (IDE) for Android app development, based on [IntelliJ IDEA](https://www.jetbrains.com/idea/). On top of IntelliJ's powerful code editor and developer tools, Android Studio offers even more features that enhance your productivity when building Android apps, such as:

* A flexible Gradle-based build system
* A fast and feature-rich emulator
* A unified environment where you can develop for all Android devices
* Instant Run to push changes to your running app without building a new APK
* Extensive testing tools and frameworks
* Lint tools to catch performance, usability, version compatibility, and other problems
* C++ and NDK support
* Built-in support for [Google Cloud Platform](http://developers.google.com/cloud/devtools/android_studio_templates/), making it easy to integrate Google Cloud Messaging and App Engine

**Project Structure in Android studio**



**Figure 2.1** The project files in Android view.

Each project in Android Studio contains one or more modules with source code files and resource files. Types of modules include:

* Android app modules
* Library modules
* Google App Engine modules

By default, Android Studio displays project files in the Android project view, as shown in figure 1. This view is organized by modules to provide quick access to project's key source files.

All the build files are visible at the top level under Gradle Scripts and each app module contains the following folders:

* **manifests**: Contains the AndroidManifest.xml file.
* **java**: Contains the Java source code files, including JUnit test code.
* **res**: Contains all non-code resources, such as XML layouts, UI strings, and bitmap images.

CHAPTER – 3

DESIGN

1. **SYSTEM DESIGN DESCRIPTION**

**3.1 Preliminary Design**

The Mobile Anti theft project contains 3 modules.

* Database Module

This module is used to fetch the real data from user and store into database using SQLite. In this module first user interface where user provides SIM IMSI number and alternate number then click submit button. Next it will store information in the database. The users can fetch the information from database to bind user interface control.

 User can enter name, alternate number and IMSI number.

 User can change the alternate number and IMSI number.

* Receiver Module

The module helps us to trigger the application at boot up and get periodic location updates after theft.

* Core Module

This module starts automatically in stealth mode because of the receiver module and checks the SIM IMSI Number with the database.

If SIM IMSI Number does not match with the database, it sends a SMS to alternate number stating that SIM has changed along with the location details in stealth mode otherwise the application terminates normally.

**Design Considerations:**

* **Input**

 IMSI (International Mobile Subscriber Identity) number of the SIM.

 Alternate number of the Owner.

 Name of the Owner

* **Output**

 Text message stating that SIM has changed and the location details.

 GPS Location of phone.

 Postal Address of the phone.

**3.2 Class Diagram**

The class diagram is the main building block of [object-oriented](https://en.wikipedia.org/wiki/Object-oriented_programming) modeling. A class diagram in the [Unified Modeling Language](https://en.wikipedia.org/wiki/Unified_Modeling_Language) (UML) is a type of static structure diagram that describes the structure of a system by showing the system's [classes](https://en.wikipedia.org/wiki/Class_(computer_science)), their attributes, operations (or methods)and the relationships among objects.

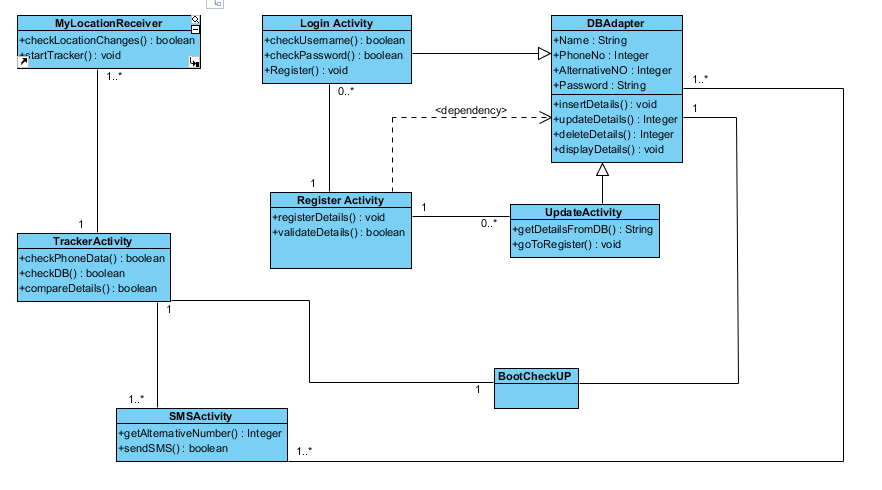
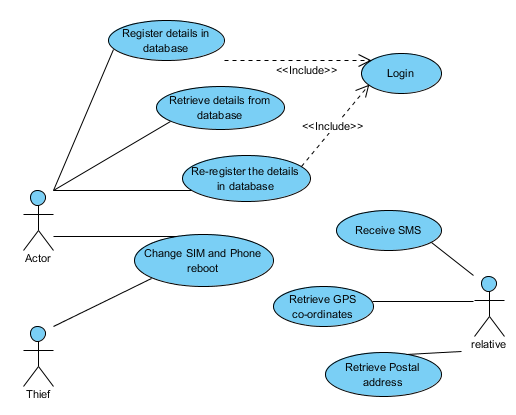


Fig:Class diagram for MAT Application

**3.3 Use Case Diagram**

Use case diagram here shows the roles of all the people involved in this project. Here in our project we consider 3 cases they are owner, relative and thief.

From the above use case diagram we know that the owner can register the details in the database when software is installed and at any point can retrieve the details and if necessary can change it accordingly and relative receives SMS, GPS co-ordinates and postal address



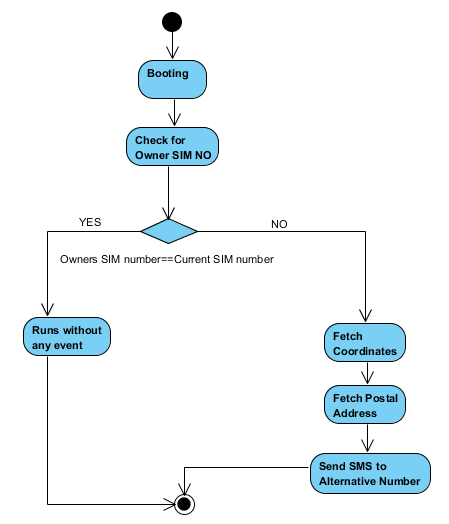
*Fig: Use case diagram for MAT App*

whenever the change of SIM is done and thief doesn’t play a role in this but he is related with changing SIM and rebooting as soon it’s done, relative of the owner will get the details accordingly.

**3.4 Activity Diagram:**

**In the Background**

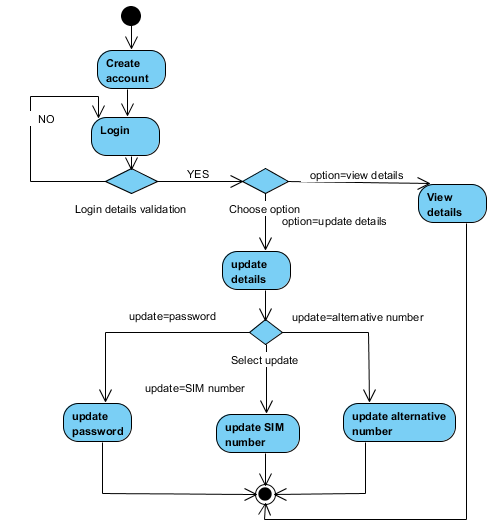
The actual process running in the background is not known to the user. He just enters the details requires and saves the application. Whenever the system is rebooted the process checks for the saved IMSI and if it’s same as the stored then it doesn’t start any events. In case if the IMSI stored is not same as current IMSI the application will act and series of events will occur without the knowledge or without giving any hint it sends SMS to the alternative number stored by the owner.



*Fig:Activity diagram for Stealth Activity*

**In the Foreground**

Foreground is nothing but what user interface of this application includes. Whenever the user invokes the application it asks for login and if the user is new then it asks to create account. After creating account user can provide all the details and never open the application again and if necessary then he can access it using the login details and only the owner can change the details if necessary. And the application will run on every boot without the user’s knowledge.



*Fig:Activity Diagram for Foreground Activity*

CHAPTER – 4

IMPLEMENTATION

1. **IMPLEMENTATION**

Implementation means to carry out or to put into effect. The implementation phase of the system of any software is to obtain source code from the design specifications. Implementation becomes necessary to obtain source code and related internal documentation required to understand the system and be demonstrated easily. Documentation and code is written in a way so that testing, debugging and modification become easy.

Post-implementation is nothing but an evaluation in which we can verify that objectives of the project is met completely and to check if actual cost of the project does not go beyond initial evaluation.

It can be even stated as review about the problems which need to be converted for the success of the project. Once implementation and conversion is done, review is conducted to check whether the system meets all the expectations and is changed if any improvements are required. Post implementation also measures performance of the system against pre-determined necessities and also checks if any modification or re-design necessary.

**4.1 Foreground/Database Module Implementation**

Here in this module we develop a database using SQLlite database which has one relation to store name, password, IMSI, relatives’ phone number. We connect to SQLlite database through java and commit values into database. Again when the phone is rebooted this database is opened and queried for IMSI number if it matches then we close it else we fetch relatives phone number and send an SMS to that number. This database is implemented with the help of a DBAdapter class whose methods are as follows:

**Database Module (Database Adapter)**

* **open ()**

**public synchronized SQLiteDatabase getWritableDatabase()**

This is user defined function which helps in opening connection to the database. It

contains the method getWritableDatabase().

It is used to create or/and open database which is used for writing and reading. Once successfully database is opened it will be cached and can be called whenever any write operation to be done on database. When no longer we need it, can be closed by call close ( ).

Errors like full disk or bad permission may occur and can cause fail of operation but if problem is fixed future attempts can be made to call open ().

* **close ()**

**public synchronized void close()**

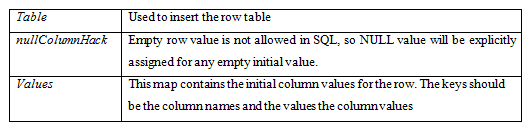
It is used to close the database which is already open so that resources are released.

* **createNote(name,password,imsi,phoneNo.)**

**public long insert (String table, String nullCoumnHack, ContentValues values)**

This is user defined function which is used to create an entry in the database.

***Parameters***



*Table 1: parameters during creating an entry in database* (Android Developers, 2007)

**Returns**

ID of the new row inserted or -1 incase error occurs.

* **deleteNote(id)**

**public int delete (String table, String whereClause, String[] whereArgs)**

This is user defined function is used to delete an entry in the database. Convenient

method in database for deleting rows.

**Parameters**



*Table 2: parameters during deleting an entry in database*

**Returns**

“Number of rows affected if a whereClause is passed in, 0 otherwise. To remove all rows and get a count pass "1" as the whereClause.”

* **fetchNote(id)**

**public Cursor query (String table, String[] columns, String selection, String[] selectionArgs, String groupBy,String having, String orderBy)**

User defined function which is used to fetch an entry from the database. Queries the table given and returns the cursor on result set.



*Table 3: parameters used for fetching an entry in database*

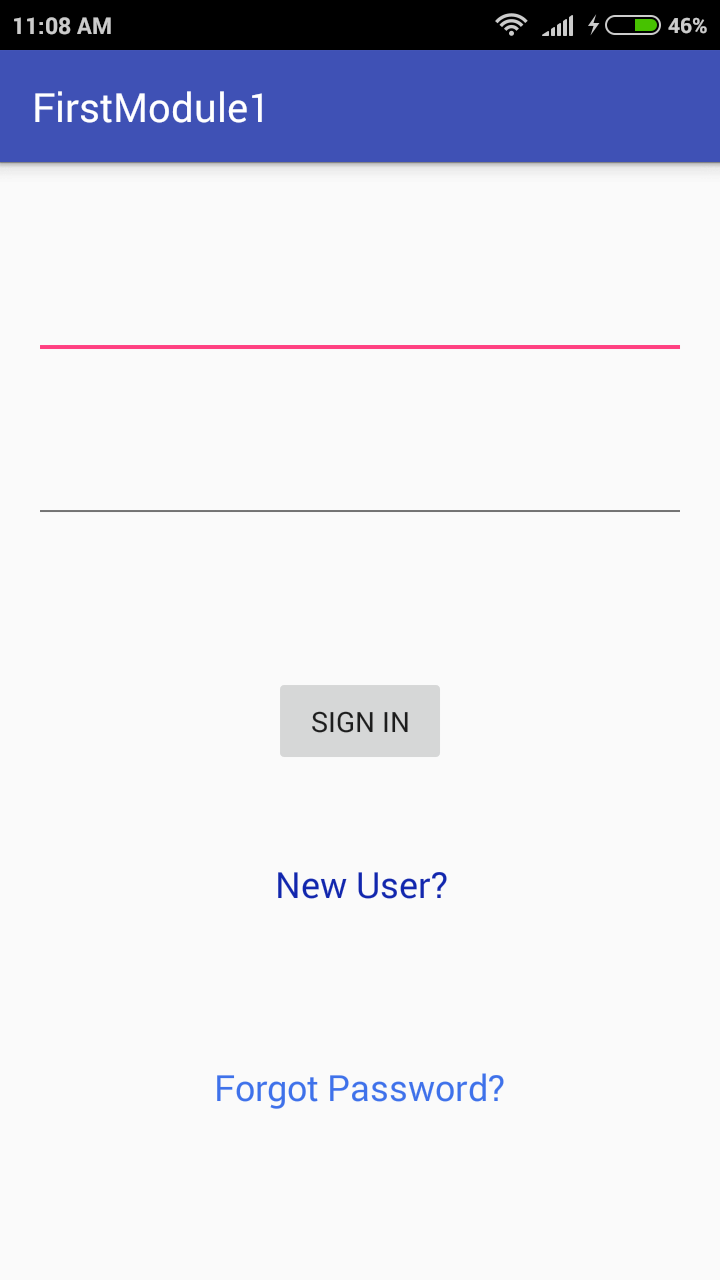
**Returns:**

Cursor object

In addition to this, theforeground module i.e., the database module of the application consists of three activities, namely the Log in activity, the Registration activity and the Update activity.

**LoginActivity** :

The UI of this activity can be created by means of xml elements provided by Android. It may look as follows:



It is to be noted that all three activities mentioned above make use of SQLiteDatabase through DBAdapter class,created earlier.

The java code for the Login activity may look as follows:

**import** android.content.Intent;  
**import** android.database.Cursor;  
**import** android.support.v7.app.AppCompatActivity;  
**import** android.os.Bundle;  
**import** android.telephony.SmsManager;  
**import** android.view.View;  
**import** android.widget.EditText;  
**import** android.widget.Toast;  
  
**public class** LoginActivity **extends** AppCompatActivity {  
 DBAdapter **db**;  
 EditText **name**,**pass**;  
 @Override  
 **protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***activity\_login***

);  
 **db**=**new** DBAdapter(**this**);  
 }  
**public void** Signed(View v)  
 {  
 **name**=(EditText)findViewById(R.id.***editText***);  
 **pass**=(EditText)findViewById(R.id.***editText2***);  
 **if**(**name**.getText().toString().equals(**""**)||**pass**.getText().toString().equals(**""**))  
 {  
 Toast.*makeText*(**this**,**"Please enter password"**, Toast.***LENGTH\_LONG***).show();  
 }  
 **else** {  
 **db**.open();  
 *//Toast.makeText(this,db.getSingleEntry(name.getText().toString()),Toast.LENGTH\_LONG).show();* **if** (**pass**.getText().toString().equals(**db**.getSingleEntry(**name**.getText().toString()))) {  
 **db**.close();  
 Intent i = **new** Intent(**"com.example.saideep.firstmodule1.UpdateActivity"**);  
 startActivity(i);  
 }

**else** {  
 Toast.*makeText*(**this**, **"UserName or Password Invalid"**, Toast.***LENGTH\_LONG***).show();  
 **db**.close();  
 }  
 }  
 }  
 **public void** getRegistered(View v)  
 {  
 **db**.open();  
  
 Cursor c=**db**

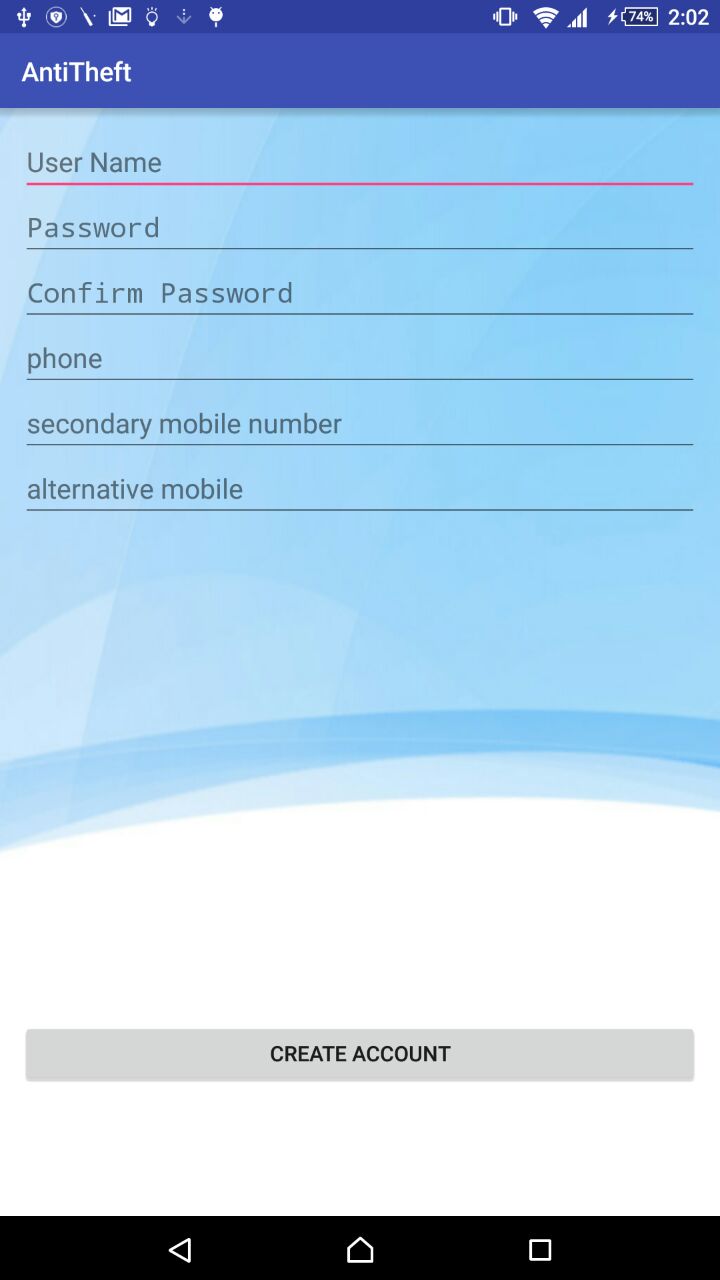
.getAllData();  
 c.moveToFirst();  
 **if**(c.getCount()<1)  
 {  
 **db**.close();  
 Toast.*makeText*(**this**,**"No Account Exists"**, Toast.***LENGTH\_LONG***).show();  
 Intent i=**new** Intent(**"com.example.saideep.firstmodule1.RegisterActivity"**);  
 startActivity(i);  
  
 }  
 **else** {  
 Toast.*makeText*(**this**,**"Permission Denied"**, Toast.***LENGTH\_LONG***).show();  
 **db**.close();  
 }  
 }  
 **public void** forgot(View v)  
 {  
 **db**.open();

Cursor c=**db**.getAllData();  
 c.moveToFirst();  
 **if**(c.getCount()<1)  
 {  
 **db**.close();  
 Toast.*makeText*(**this**,**"No Account Exists"**, Toast.***LENGTH\_LONG***).show();  
 }  
 **else** {  
 Toast.*makeText*(**this**, **"Password will be texted to the registered phone number!!"**, Toast.***LENGTH\_LONG***).show();  
  
 SmsManager sms = SmsManager.*getDefault*();  
 sms.sendTextMessage(c.getString(3), **null**, **"password is"** + c.getString(2) + **""**, **null**,**null**);  
 **db**.close();

}  
 }  
}

**Registration Activity:**

The UI of this activity can be created by means of xml elements provided by Android. It may look as follows:



The above activity makes sure that only one record of user is allowed by the app,by removing any earlier record in case of updation and creates a new record of user data in the app database.

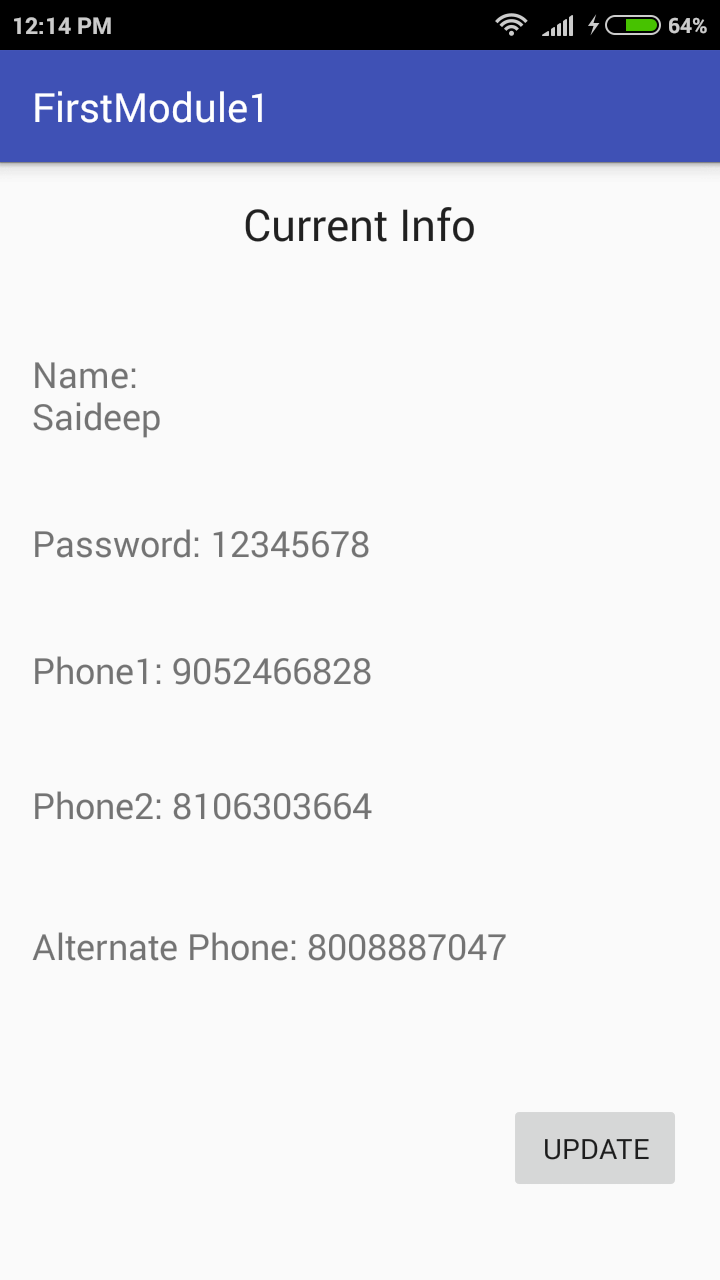
The Code Snippet for the above activity is as follows:

import android.content.Intent;  
import android.support.v7.app.AppCompatActivity;  
import android.os.Bundle;  
import android.view.View;  
import android.widget.EditText;  
import android.widget.Toast;  
  
public class RegisterActivity extends AppCompatActivity {  
 DBAdapter db;  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*activity\_register*);  
 db=new DBAdapter(this);  
 }  
 public void registerUser(View v)  
 {  
 EditText etn=(EditText)findViewById(R.id.*editText1*);  
 String name=etn.getText().toString();  
 EditText etp=(EditText)findViewById(R.id.*editText2*);  
 String pass=etp.getText().toString();  
 EditText etl1=(EditText)findViewById(R.id.*editText3*);  
 String phone1=etl1.getText().toString();  
 EditText etl2=(EditText)findViewById(R.id.*editText4*);  
 String phone2=etl2.getText().toString();  
 EditText etm=(EditText)findViewById(R.id.*editText5*);  
 String recphone=etm.getText().toString();  
 db.open();

long id=db.insertData(name,pass,phone1,phone2,recphone);  
 db.close();  
 if(id==-1)  
 Toast.*makeText*(this, "Insufficient Data", Toast.*LENGTH\_LONG*).show();  
 else  
 Toast.*makeText*(this, "Info Added", Toast.*LENGTH\_LONG*).show();  
 etn.setText("");  
 etp.setText("");  
 etl1.setText("");  
 etl2.setText("");  
 etm.setText("");  
 Intent i=new Intent(this, LoginActivity.class);  
 i.addFlags(Intent.*FLAG\_ACTIVITY\_CLEAR\_TOP*);startActivity(i);  
 }  
}

**UpdateActivity:**

The UI of this activity can be created by means of xml elements provided by Android. It may look as follows:



The Update Activity displays the current details of the user and provides him an option to re-register by means of the update button.

The code for the update activity is as follows:

**import** android.content.Intent;  
**import** android.database.Cursor;  
**import** android.support.v7.app.AppCompatActivity;  
**import** android.os.Bundle;  
**import** android.view.View;  
**import** android.widget.TextView;  
  
**public class** UpdateActivity **extends** AppCompatActivity {  
 DBAdapter **db**,**db1**;  
 @Override  
 **protected void** onCreate(Bundle savedInstanceState) {  
 **db**=**new** DBAdapter(**this**);  
 **db1**=**new** DBAdapter(**this**);  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***activity\_update***);

**db**.open();  
 Cursor c=**db**.getAllData();  
 c.moveToFirst();  
 CharSequence a[]=**new** CharSequence[]{c.getString(1),c.getString(2),c.getString(3),c.getString(4),c.getString(5)};  
 **db**.close();  
 TextView name=(TextView)findViewById(R.id.***textView8***);  
 TextView pass=(TextView)findViewById(R.id.***textView9***);  
 TextView line1=(TextView)findViewById(R.id.***textView10***);  
 TextView line2=(TextView)findViewById(R.id.***textView12***);  
 TextView phone=(TextView)findViewById(R.id.***textView13***

);  
 name.setText(**"Name: "**+a[0]);  
 pass.setText(**"Password: "**+a[1]);  
 line1.setText(**"Phone1: "**+a[2]);  
 line2.setText(**"Phone2: "**+a[3]);  
 phone.setText(**"Alternate Phone: "**+a[4]);  
 }  
 **public void** update(View v)  
 {  
 **db1**.open();  
 **db1**.deleteData(1);  
 **db1**.close();  
 Intent i=**new** Intent(**"com.example.saideep.firstmodule1.RegisterActivity"**);  
 startActivity(i);  
 }  
  
}

**4.2 Receiver Module:**

This is a relatively small module of the application however, it caters the basic needs of the core module. This module contains the BootCheckup and LocationChange BroadcastReceivers which are responsible for firing the core functionalities of the Anti-Theft Mobile App**.**

**BootCheckUp Receiver:**

The code for BootCheckUp Receiver is as follows:

**import** android.content.BroadcastReceiver;  
**import** android.content.Context;  
**import** android.content.Intent;  
**import** android.widget.Toast;  
**public class** BootCheckUp **extends** BroadcastReceiver {  
 @Override  
 **public void** onReceive(Context context, Intent intent) {  
  
  
 *//---starts the main activity of our app---* Intent i = **new** Intent(context,TrackerActivity.**class**);  
 i.addFlags(Intent.***FLAG\_ACTIVITY\_NEW\_TASK***);  
 context.startActivity(i);  
 }  
}

It starts the stealth tracker activity at the time of device boot up.

**LocationChange Receiver:**

The LocationChange BroadcastReceiver checks for location changes of the mobile device and fires the SMS Activity of the Core Module accordingly.

It makes use of the Location Manager framework of Android for LOaction details.

import android.content.BroadcastReceiver;  
import android.content.Context;  
import android.content.Intent;  
import android.location.Address;  
import android.location.Location;  
import android.location.LocationManager;  
  
import android.widget.Toast;  
public class MyLocationReceiver extends BroadcastReceiver {  
  
 @Override  
 public void onReceive(Context context, Intent intent) {  
 String locationKey = LocationManager.*KEY\_LOCATION\_CHANGED*;  
 String providerEnabledKey = LocationManager.*KEY\_PROVIDER\_ENABLED*;

if (intent.hasExtra(providerEnabledKey)) {  
 if (!intent.getBooleanExtra(providerEnabledKey, true)) {  
 Toast.*makeText*(context,  
 "Provider disabled",  
 Toast.*LENGTH\_SHORT*).show();  
 } else {  
 Toast.*makeText*(context,  
 "Provider enabled",  
 Toast.*LENGTH\_SHORT*).show();  
 }  
 }  
 if (intent.hasExtra(locationKey)) {  
 Location loc = (Location)intent.getExtras().get(locationKey);  
 Toast.*makeText*(context,  
 "Location changed : Lat: " + loc.getLatitude() + " Lng: " + loc.getLongitude(), Toast.*LENGTH\_SHORT*).show();  
  
 Intent i=new Intent("com.example.saideep.mlocationtracker.SmsActivity");  
 i.putExtra("latitude",loc.getLatitude());  
 i.putExtra("longitude",loc.getLongitude());  
 i.addFlags(Intent.*FLAG\_ACTIVITY\_NEW\_TASK*);  
 context.startActivity(i);  
  
 }  
 }  
  
}

**4.3 Core Module:**

This module is concerned with base functionality of the app which includes the running of the app in the background in stealth mode and sending of SMS containing location details. But the prerequisites of this module are the Broadcast Receivers to fire boot up and location change events, this module includes activities which check the information from the database and help us compare it with the current phone state and trigger the necesssary functionalities.

The following are the activities included in this module are Tracker Activity and SMS Activity.

**TrackerActivity:**

The Tracker Activity runs in stealth mode and it starts running at boot up.As it works in stealth mode no focus is made on its UI and in fact, the activity UI is made invisible and transparent. The code for Tracker Activity is as follows:

import android.support.v7.app.AppCompatActivity;  
import android.app.PendingIntent;  
import android.content.ComponentName;  
import android.content.Context;  
import android.content.Intent;  
import android.content.pm.PackageManager;  
import android.database.Cursor;  
import android.location.Criteria;  
import android.location.LocationManager;  
import android.os.Bundle;  
import android.telephony.TelephonyManager;  
import android.util.Log;  
import android.view.WindowManager;  
import android.widget.Toast;  
import java.util.List;

public class TrackerActivity extends AppCompatActivity {  
 LocationManager lm;  
 PendingIntent pendingIntent;  
 DBAdapter db;  
  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*activity\_tracker*);  
 getWindow().addFlags(WindowManager.LayoutParams.*FLAG\_NOT\_TOUCHABLE*);  
 PackageManager p = getPackageManager();  
 ComponentName componentName = new ComponentName(this, com.example.saideep.mlocationtracker.TrackerActivity.class

);

p.setComponentEnabledSetting(componentName,PackageManager.*COMPONENT\_ENABLED\_STATE\_DISABLED\_UNTIL\_USED*, PackageManager.*DONT\_KILL\_APP*);

db=new DBAdapter(this);  
  
TelephonyManager tm = (TelephonyManager) getSystemService(Context.*TELEPHONY\_SERVICE*);

String simID = tm.getSimSerialNumber();  
 if (simID != null)  
 Toast.*makeText*(this, "SIM card ID: " + simID, Toast.*LENGTH\_LONG*).show();  
 String telNumber = tm.getLine1Number();  
 if (telNumber != null)  
 Toast.*makeText*(this, "Phone number: " + telNumber, Toast.*LENGTH\_LONG*).show();  
  
 db.open();  
 Cursor cursor = db.getAllData();  
 cursor.moveToFirst();  
 if (!(telNumber.equals(cursor.getString(3)) || telNumber.equals(cursor.getString(4)))) {  
 db.close();  
 *//---use the LocationManager class to obtain locations data---* lm = (LocationManager) getSystemService(Context.*LOCATION\_SERVICE*);  
 *//---print out all the location providers---* List<String> locationProviders = lm.getAllProviders();  
 for (String provider : locationProviders) {  
 Log.*d*("LocationProviders", provider);  
 }  
 *//---set the criteria for best location provider---* Criteria c = new Criteria();  
 c.setAccuracy(Criteria.*ACCURACY\_FINE*);  
 *//---OR---  
 //c.setAccuracy(Criteria.ACCURACY\_COARSE);* c.setAltitudeRequired(false);  
 c.setBearingRequired(false);  
 c.setCostAllowed(true);  
 c.setPowerRequirement(Criteria.*POWER\_HIGH*);  
  
 *//---get the best location provider---* String bestProvider = lm.getBestProvider(c, true);  
 Toast.*makeText*(this, "Best provider is " + bestProvider, Toast.*LENGTH\_SHORT*).show();  
  
 Intent i = new Intent(this, MyLocationReceiver.class);  
 pendingIntent = PendingIntent.*getBroadcast*(this, 0, i, PendingIntent.*FLAG\_UPDATE\_CURRENT*);  
 *//---request for location updates using GPS OR Network---* try {  
 if (bestProvider.equals("network"))  
 lm.requestLocationUpdates(LocationManager.*NETWORK\_PROVIDER*, 60000, 100, pendingIntent);  
 else  
 lm.requestLocationUpdates(LocationManager.*GPS\_PROVIDER*, 60000, 100, pendingIntent);  
 } catch (SecurityException s) {  
 s.printStackTrace();  
 }  
  
 }  
 else  
 {  
 db.close();  
 }  
 }  
}

The above Activity takes care of the communication of the database with the app and reading the phone information through Telephony Manager of Android. In addition, it also chooses the best location provider sends the location details to the SMS Activity if the current SIM number does not match with the one stored in the app database.

**SMS Activity:**

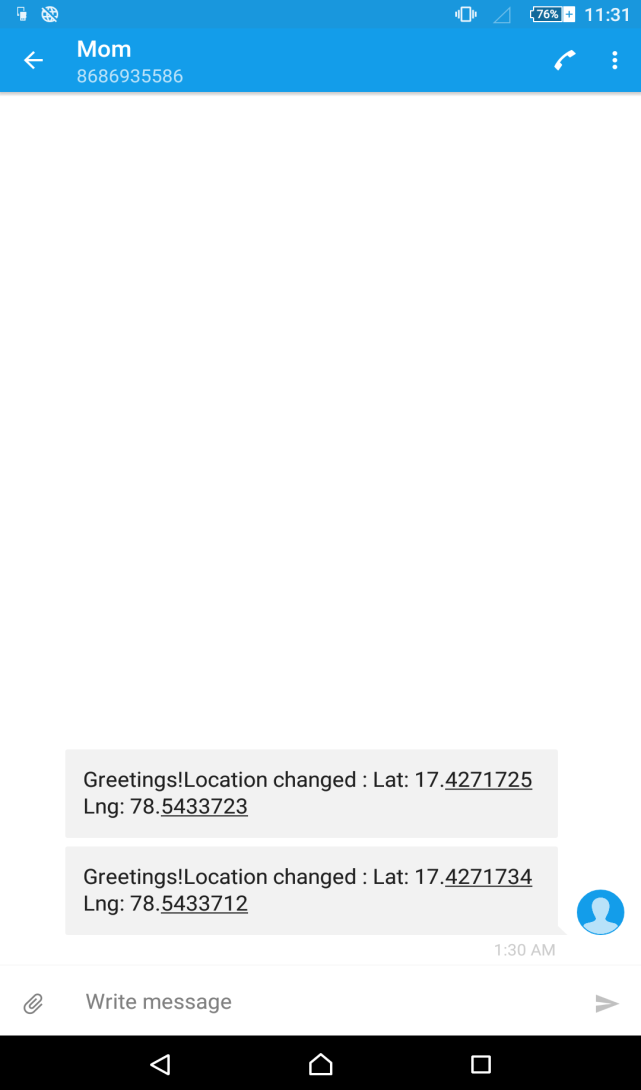
This activity takes care of the necessary measures get the location details and sends it to the alternate mobile number stored in the app database.

The code for SMS Activity is as follows:

import android.content.Context;  
import android.database.Cursor;  
import android.location.Address;  
import android.location.Geocoder;  
import android.support.v7.app.AppCompatActivity;  
import android.os.Bundle;  
import android.telephony.SmsManager;  
import android.widget.Toast;  
  
import java.io.IOException;  
import java.util.List;  
import java.util.Locale;  
  
public class SmsActivity extends AppCompatActivity {  
 DBAdapter db;  
 String altphone;  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*activity\_sms*);  
 db = new DBAdapter(this);  
 db.open();  
 Cursor cursor=db.getAllData();  
 cursor.moveToFirst();  
 altphone=cursor.getString(5);  
 db.close();  
 addGet(this,getIntent().getDoubleExtra("latitude",0.0), getIntent().getDoubleExtra("longitude", 0.0));  
 }  
 public void addGet(Context context, double latitude, double longitude) {  
  
 Geocoder geocoder = new Geocoder(context, Locale.*getDefault*

());  
  
 try  
 {  
 List<Address> addresses = geocoder.getFromLocation(latitude, longitude, 1);  
 Toast.*makeText*(context,"Addresses"+addresses , Toast.*LENGTH\_SHORT*).show();  
  
  
 SmsManager sms = SmsManager.*getDefault*();  
 sms.sendTextMessage(altphone, null,""+addresses,null, null);  
  
 }  
 catch (IOException e)  
 {  
 e.printStackTrace();  
  
 }  
 }  
}

Finally the SMS received by the alternate mobile number may look as:



CHAPTER – 5

TESTING

1. **TESTING**

**5.1 Testing Process**

After designing phase there is the coding phase. In this phase, every module identified and specified in the design document is independently Coded and Unit is tested .

Unit testing (or module testing) is the testing of different units or modules of a system. In this phase, the physical design of the system is converted into the logical programming language.



*Fig 6: Levels of testing*

**5.2 Testing Objectives**

The coding for the Drug Accessor is done in java, jsp, struts and Daffodil server

before starting of the coding.

We have tried to follow some coding standards and Guidelines.

The coding standards are: -

Naming standards for the java Classes and variables etc.

Screen design standards.

Validation and checks that need to be implemented.

The Guidelines are: -

Code should be well documented.

Coding style should be simple.

Length of function should be short.

**5.3 Levels of Testing**

**5.3.1 Unit Testing**

In this, the programs that made up the system were tested. This is also called as program testing. This level of testing focuses on the modules, independently of one another. Unit testing is conducted to check if individual modules are working correctly. In unit testing, code testing strategy is adopted which initially examines the logic of the program and all the syntax errors will be checked for, during the development process itself. In this test case every instruction or module is checked in the program i.e. during testing every path is checked and even the loops. In this case we take random values and check if the program executes properly. Unit testing includes a detailed explanation of test cases, management of test cases and testing criteria.

***User Input***

In User interface the data entry is done through GUI and tested. Each element is tested for valid range and invalid range of data.

***Error Handling***

In this system we have tried to handle all the errors that are occurred while running the GUI forms. The common errors we saw are reading the empty record and displaying a compiler message, etc.We have handled errors using exception handling. Many errors have been handled using try catch blocks provided by java. Most of the errors occurred while committing data into database these have been handled with utmost care.

* + 1. **Integration Testing**

In this process of testing all the modules which are developed by different people are integrated using an integration plan. Integration plan is nothing but the plan which is used to combine all the small modules together to recognize the full system. In this methodology the modules are checked after every partly integrated system is combined.

Integration testing mainly checks for interfaces between units.

Module dependency graph is the most significant factor which monitors the integration plan. The way of different modules calling each other in an order explains the module dependency graph. Structured chart is also a form of module dependency graph. So we can develop integration plan based on structure chart using any of the following approaches.

Big-bang approach.

Top-down approach.

Bottom-up approach.

Mixed approach.

**5.3.3 Bottom up Integration Testing**

In this methodology, every subsystem is tested distinctly and after integration the complete system is tested. Subsystem consists of several modules that communicate which each other using well defined interfaces.The key purpose of testing every subsystem is to check the interface between various modules; here both the data and control interface is tested.

The primary benefit of bottom-up integration testing is that numerous separate subsystems are tested concurrently and the disadvantage of this approach is the complexity involved because of the large number of small subsystems.

In Main module, individual programs are tested initially then after obtaining successful results we moved on further for integration. After integrating certain parts it’s tested again until we obtained the good results later on the whole system is combined and tested for system testing.

* + 1. **System Testing**

System testing is the process in which the overall system is checked if the interactions between components are working properly. It is also used to find inconsistencies between specification and implementations. It also involves reliability, performance, load and security testing.

In this stage the system is checked experimentally so that all the user requirements are fulfilled.

The testing of system takes places in different levels to ensure that system is free from failure. System testing is usually done by the test team who ensures the complete working of the system. System testing is usually performed on system test machines in simulated environments like VMware. System is provided to the user for entry validations and the values are provided to ensure the correctness at each and every stage. The values are provided to user to make sure he doesn’t enter unrelated data. These tests are done to make sure it produces correct outputs. All the output was checked and was found to be correct. After checking for all the tests, sessions are conducted for feedback and sent finally for acceptance test. Finally system gets accepted and is checked with live data.

There are actually 3 different kinds of system testing they are

Alpha testing- This type of testing is actually carried out by a test team related

with organization

Beta testing- This type of testing is carried out by a selected group of

customers.

Acceptance testing- This test is carried out by the customer to decide whether

to accept the system delivery or not

.

The Test Cases used to verify this application are:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test ID** | **Test Type** | **Test Description** | **Execution Steps** | **Expected Output** | **Actual Output** | **Comments** |
| ATM\_1.1 | User Interface and Experience | Verify if the application is installed successfully. | 1)Open the playstore /install the .apk on your device.  2)Click the ATM icon in the App Drawer. | The Application should start successfully. | Application is installed. | This is to ensure that the application is installed successfully on the device. |
| ATM\_1.2 | User Registration | Verify if the Application allows a new user to register successfully. | 1)Open the ATM app and click on new user.  2)Fill the details in the registration form. | The app should redirect to the registration page and store user details in the app database. | Successful user registration. | This is to ensure that the user details are available with the app. |
| ATM\_1.3 | User login authentication. | Verify if the user has an account and if he/she does then authenticate the user based on password. | 1)App should if an account already exists.  2)If an account exists then the app should authenticate the user. | The app should prompt “No account” if there is no user account else it should authenticate user based on login name and password | Successful authentication of the user. | This is to ensure that only a single valid user account exists whose access is restricted through password. |
| ATM\_1.4 | Update and check user details. | Verify if the Application allows to login to his account to check and update his/her details. | 1)User should sign in to his account using his login name and password.  2)Display user details and an option to update. | The app should allow a user to sign in to his account and should be able to verify and update their details. | The app should display the user details on successful login and update user details if needed. | This is needed in case user decides to update his/her details. |
| ATM\_1.5 | Forgot Password. | Verify if the application allows a user to reset the password. | 1)App should provide a forgot password option to reset the password. | The user should be given a choice to reset his password successfully. | The user should be able to reset his/her password. | This is needed if the user forgot his password. |
| ATM\_1.6 | Stealth mode after device boot up. | Verify if the app starts after boot up in stealth mode. | 1)App should start in stealth mode after device boot up (verify this with a toast). | The app should pop a toast to show that it has started without any clickable foreground UI in stealth mode. | A Toast is to be prompted saying “App Started!!!” | This is needed to verify if the app has started after device bootup. |
| ATM\_1.7 | Sending Location Details in the stealth mode. | Verify if the app sends location details through SMS if the current SIM number does not match with the one stored in user database. | 1)App should compare current SIM number with the one stored in the app database and send an SMS containing Location details to the registered alternate number.  2)If the SIM number matches correctly then the app should simply terminate in background. | The app should send an SMS containing location details if current SIM number does not match with the number stored in the app database. | An SMS with location details is sent to the alternate number in the app database without the user’s knowledge. | This is the ,main functionality through we can track the device location. |
| ATM\_1.8 | Sending Location details based location changes. | Verify if the app keeps sending current location details whenever the device location changes after theft. | 1)App runs in background and sends location details through SMS every time the location details. | The app should keep sending SMS to the alternate number for every major location change. | The app keeps sending SMS containing location details of the device. | This is important to verify the current location of the mobile device. |

1. **CONCLUSION AND FUTURE WORK**

The Current Version of Anti Theft Mobile Application can only send SMS and details about SIM and GPS co-ordinates when theft is detected. It would also retrieve the postal address of the stolen phone and send it as an SMS to the alternate phone number.

Future versions of the application can provide additional functionalities which may include the use of the Mobile device’s camera to take photographs of the perpetrator in stealth mode and sending it to the alternate phone number along with the location details.

1. **BIBLIOGRAPHY**

BOOKS REFERRED:

[1] Beginning Android 4 Application Development, Wei-Meng Lee, Wrox Publications, 2012.

[2] Android Wireless Application Development, Shane Conder, Lauren Darcy, Second Edition, Addison-Wesley, 2012

WEBSITES REFERRED:

[1] Android. (2010, 11 03). Retrieved 3 24, 2012, from New Android Market for Phones:

<http://www.android.com/>

[2] Android Developers. (2007, 12 17). *what is Andoid.* Retrieved 03 29, 2012, from

<http://developer.android.com/guide/basics/what-is-android.html>

[3] StackOverflow: <http://stackoverflow.com/questions/15711499/get-latitude-and-longitude-with-geocoder-and-android-google-maps-api-v2>