AN ANDROID APPLICATION FOR FORT RILEY SOLDIERS FOR INTEGRATED TRAINING AREA MANAGEMENT AND SUSTAINABLE RANGE AWARENESS

by

HEMANTH REDDY PATHI REDDY

B.Tech., PDPM Indian Institute of Information Technology, Design and Manufacturing Jabalpur, 2009

A REPORT

Submitted in partial fulfillment of the requirements for the degree

MASTER OF SCIENCE

Department of Computing and Information Sciences
College of Engineering

KANSAS STATE UNIVERSITY
Manhattan, Kansas

2011

Approved by:

Major Professor
Dr. Daniel Andresen

Abstract

The purpose of this project is to develop an Android applica tion for Fort Riley soldiers so as to support the theme “War Fighters Supporting War Fighters by Caring for Military Training Lands”.

There are four core features in this application:

1. Find a Gully
2. Report a Gully
3. View Current Weather
4. View Current Satellite Image

Features are explained in brief below.

1. Find a gully

In this feature based on the current location of soldier, application will display all gullies near that location using Google Maps API. Soldier can also view the gully details by tapping the gully icon.

2. Report a Gully

In this feature, soldier can report a new gully i.e.; gully which is not already present on the map. This gully will be stored as unverified gully in the database. Once this gully is verified it will be changed to verified gully and it will be plotted on the Google map.

3. View Current Weather

In this feature, soldier can view the current weather conditions of Fort Riley.

4. View Current Satellite Image

In this feature, soldier can view the current satellite image of Fort Riley.

Table of Contents

List of Figures

List of Tables

Acknowledgements

My special thanks to my major professor **Dr. Daniel Andresen** for giving me timely advice, encouragement, and guidance throughout the project.

I would also like to thank **Dr. Gurdip Singh** and **Dr. Torben Amtoft** for graciously accepting to serve on my committee.

I would also like to thank **Dr. Shawn Hutchinson** for giving me an opportunity to work on this project.

Finally, I would like to thank the administrative and technical support staff of the department of CIS for their support throughout my graduate study.

Dedication

I would like to dedicate this project to my father Vishnu Vardhan Reddy, my mother Vijayasree, my uncle Thirupathi Reddy and my aunt Jayasree.

1. CHAPTER 1

1.1. Introduction

The current project involves the development of an Android application for Fort Riley soldiers. The main aim of this project is to help soldiers find gullies on the military training lands and also to report the new gullies found by them. The application also has features to view current weather and current satellite image of Fort Riley.

1.2. Motivation

Gullies are small channels of erosion on landscapes, typically agricultural, that are caused by the concentration of overland flow usually between two opposing slopes. Gully erosion is a serious problem on military training lands resulting in not only soil erosion and environmental degradation, but also increased soldier injuries and equipment damage. Knowledge of gullies beforehand will help prevent these injuries, damages and the easiest way to get this information is through mobile phones and Android phones account for half of U.S.A mobile smart phones.

1.3. Project Description

In this section all features in application are explained in brief.

1.3.1. Find a Gully

In this feature, based on the current location of soldier, application will display all gully locations near the present location. Soldier can view all the details of the gully like name, width, depth and image by tapping on that particular gully icon. Soldier can also view the bigger image of the gully by clicking the 'View Image' button. He can also switch the Google Map view between satellite and street views.

1.3.2. Report a Gully

In this feature, soldier can report a new gully i.e.; gully which is not already present on the map. Soldier is asked to enter details like gully width, gully depth and user id. Latitude and Longitude values are automatically calculated based on the current location of the soldier. Once the submit button is pressed the gully details will be stored in the MySQL database.

1.3.3. View Current Weather

In this feature, soldier can view the current weather conditions of Fort Riley like temperature, humidity and sky condition.

1.3.4. View Current Satellite Image

In this feature, soldier can view the current satellite image of Fort Riley which is regularly updated by the people of GISSAL using ArcGIS. "ArcGIS is a system for working with maps and geographic information. It is used for: creating and using maps; compiling geographic data; analyzing mapped information; sharing and discovering geographic information; using maps and geographic information in a range of applications; and managing geographic information in a database."[21] This ArcGIS Map satellite image is different from Google Map satellite image in the sense on the Google Map we can only add the data to it but we cannot edit it but whereas on ArcGIS Map we can edit the data according to user requirements as these maps are created by user itself.

2. CHAPTER 2

2.1. Requirements Gathering

I have collected all the information required for the project from GISSAL (Geographic Information Systems Spatial Analysis Laboratory). Regular meetings with the people from GISSAL at every phase of the project helped in meeting the requirements of the project. Apart from this, this project required a lot of background research on building applications on Android framework.

All the information regarding gullies is provided in Keyhole Markup Language (KML) file format. KML is an XML notation for expressing geographic annotation and visualization within internet-based, two-dimensional maps and three-dimensional earth browsers [24].

2.2. Requirement Analysis

Below are the software and hardware requirements for developing this application (not for installing the application).

2.2.1. Software Requirements

Operating System: Windows XP or higher / Mac OS X 10.5.8 or later / Linux

Platform: Android SDK Framework

IDE: Eclipse 3.5(Galileo) or higher

Android Emulator: SDK Version 2.2 or Higher

Database: MySQL

Technologies used: Java, XML, PHP, Google Maps API

2.2.2. Hardware Requirements

Processor: P IV or higher

RAM: 256 MB

Space on disk: minimum 250MB

2.3. Feasibility Study

2.3.1. Economic Feasibility

The project is economically feasible as it only requires a mobile phone with Android operating system. The application is free to download once released into Android market. The users should be able to connect to internet through mobile phone and this would be the only cost incurred on the project.

2.3.2. Technical Feasibility

To develop this application, a high speed internet connection, a database server, a web server and software are required. The current project is technically feasible as the application was successfully deployed on Android Emulator.

2.3.3. Behavioral Feasibility

The application is behaviorally feasible since it requires no technical guidance, all the modules are user friendly and execute in a manner they were designed to.

3. CHAPTER 3

3.1. System Design

Requirements gathering followed by careful analysis leads to a systematic Object Oriented Design (OOD). Various activities have been identified and are represented using Unified Modeling Language (UML) diagrams. UML is used to specify, visualize, modify, construct and document the artifacts of an object-oriented software-intensive system under development [13].

3.1.1. Use Case Diagram

In the Unified Modeling Language (UML), the use case diagram is a type of behavioral diagram defined by and created from a use-case analysis. It represents a graphical over view of the functionality of the system in terms of actors, which are persons, organizations or external system that plays a role in one or more interaction with the system. These are drawn as stick figures. The goals of these actors are represented as use cases, which describe a sequence of actions that provide something of measurable value to an actor and any dependencies between those use cases [12].

In this application there is only actor – soldier and below is the use case diagram of this application.

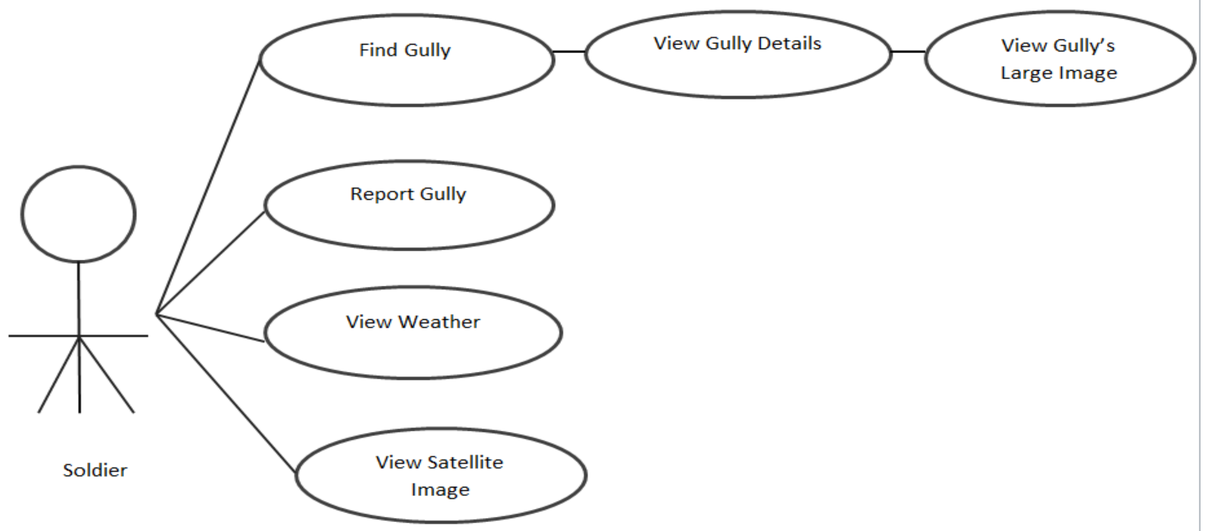


Figure 3.1 Use case Diagram

3.1.2. Class Diagram

In the Unified Modeling Language (UML), class diagram is a type of static structure diagram which describes the structure of a system by showing the classes, attributes and the relationships between the classes of a system. It is the main building block in object oriented modeling. The classes represent both the main objects and or interactions in the application. The class diagram consists of classes which are represented in boxes which contain three parts. The name of the class is contained in upper part, with the attributes of classes in middle part and the bottom part contains the methods or operations that the classes undertake [17]. With detailed modeling, the classes of the conceptual design are split into number of subclasses.

In this application there are 9 classes and below is the class diagram of this application.

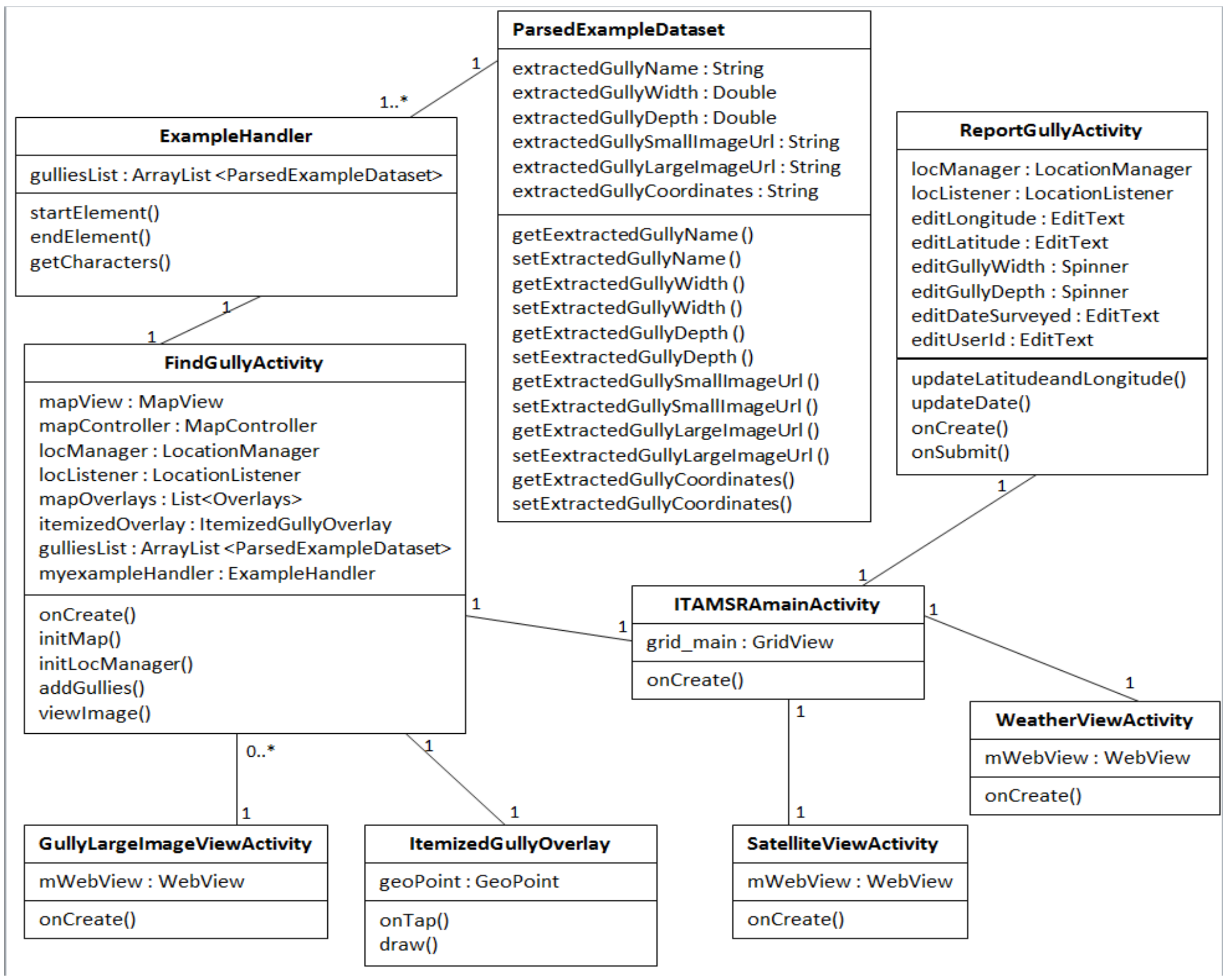

ParsedExampleDataset
extractedGullyName : String
extractedGullyWidth : Double
extractedGullyDepth : Double
extractedGullySmallImageUrl : String
extractedGullyLargeImageUrl : String
extractedGullyCoordinates : String
getEextractedGullyName ()
setExtractedGullyName ()
getExtractedGullyWidth ()
setExtractedGullyWidth ()
getExtractedGullyDepth ()
setEextractedGullyDepth ()
getExtractedGullySmallImageUrl ()
setExtractedGullySmallImageUrl ()
getExtractedGullyLargeImageUrl ()
setEextractedGullyLargeImageUrl ()
getExtractedGullyCoordinates()
setExtractedGullyCoordinates()
ExampleHandler
gulliesList : ArrayList <ParsedExampleDataset>
startElement()
endElement()
getCharacters()
ReportGullyActivity
locManager : LocationManager
locListener : LocationListener
editLongitude : EditText
editLatitude : EditText
editGullyWidth : Spinner
editGullyDepth : Spinner
editDateSurveyed : EditText
editUserId : EditText
updateLatitudeandLongitude()
updateDate()
onCreate()
onSubmit()
FindGullyActivity
mapView : MapView
mapController : MapController
locManager : LocationManager
locListener : LocationListener
mapOverlays : List<Overlays>
itemizedOverlay : ItemizedGullyOverlay
gulliesList : ArrayList <ParsedExampleDataset>
myexampleHandler : ExampleHandler
onCreate()
initMap()
initLocManager()
addGullies()
viewImage()
ITAMSRAmainActivity
grid_main : GridView
onCreate()
WeatherViewActivity
mWebView : WebView
onCreate()
GullyLargeImageViewActivity
mWebView : WebView
onCreate()
ItemizedGullyOverlay
geoPoint : GeoPoint
onTap()
draw()
SatelliteViewActivity
mWebView : WebView
onCreate()
1
1..*
1
1
1
1
1
1
1
1
0..*
1
1
1
1
1

Figure 3.2 Class Diagram

ITAMSRAmainActivity class is the main activity of the application used to create icons with text using GridView, Layout Inflater and to link these icons to other activities (features) of the application.[15] Lines of Code in this class is 152.

FindGullyActivity class is used to show gullies near the current location of soldier on Google Map. MapController, LocationManager and LocationListener class objects have been used to implement this feature. The four classes ItemizedGullyOverlay, ExampleHandler, ParsedExampleDataset and GullyLargeImageViewActivity are also to deal with find a gully feature. Lines of Code in this class is 230.

ItemizedGullyOverlay class is used to add gullies with details onto Google Map. This class extends ItemizedOverlay class which handles sorting north-to-south for drawing, creating span bounds, drawing a marker for each point, and maintaining a focused item. It also matches screen-taps to items, and dispatches Focus-change events to an optional listener.[16] Lines of Code in this class is 155.

ExampleHandler class is used to parse the xml data from KML file using SAX (Simple API for XML) Parser. Lines of Code in this class is 133.

ParsedExampleDataset is used to implement encapsulation of the parsed data using set and get methods. Lines of Code in this class is 147.

GullyLargeImageViewActivity class is used to view large image of a gully. Lines of Code in this class is 48.

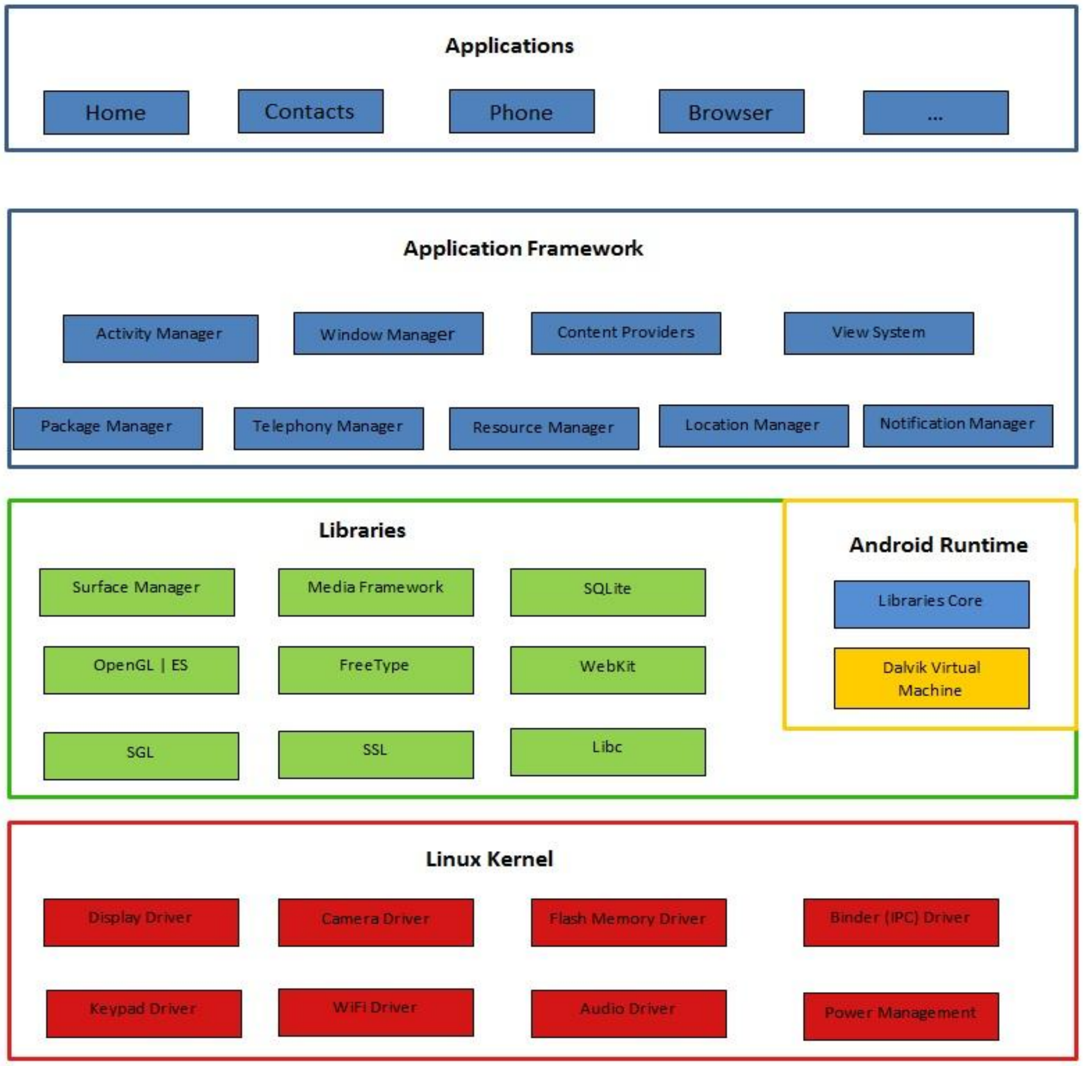
ReportGullyActivity class is used to allow soldier report new gullies by entering the gully details on a form. The architecture used to implement this feature is explained in Section 3.2.2. Lines of Code in this class is 180.

WeatherViewActivity class is used to get weather information from a webpage. WebView is used to access this webpage from an Android device. Lines of Code in this class is 48.

SatelliteViewActivity class is used to get satellite image from webpage. WebView is used to access this webpage from an Android device. Lines of Code in this class is 48.

3.2. System Architecture

3.2.1. Android Architecture



[10]

Figure 3.3 Android Architecture showing the major components of Android OS

3.2.2. Overall Software Architecture

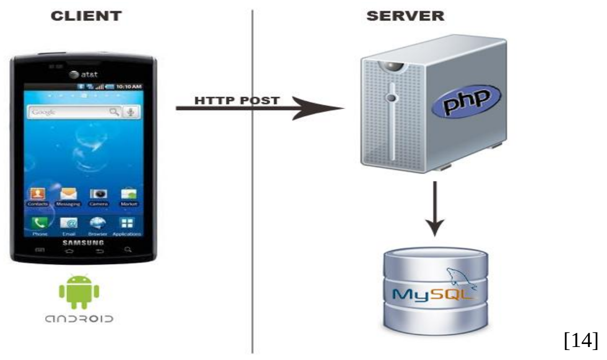


Figure 3.4 Connection between PHP (server) and Android (client) using HTTP and JSON Connection

The architecture shown in above figure is used in report a gully feature where the data from android goes to webserver (PHP) to database server (MySQL). PHP is used here because of the interaction it can offer with the databases and it is easy to deploy on the GISSAL web server and here it sits in middle as shown in figure. On Android, HTTPClient is used to communicate with webserver (PHP). JSON (JavaScript Object Notation) is a lightweight text-based open standard designed for human-readable data interchange and it is used in this application to send data from Android device to PHP Script. SQL insert query is written in PHP script to insert this data into MySQL database server. [14]

4. CHAPTER 4

4.1. Implementation

The Android application is developed by using Eclipse Helios Integrated Development Environment (IDE). Android SDK which includes a variety of custom tools that help us develop mobile applications on the Android platform is used. Android Emulator and ADT plug-in for Eclipse are the most important of these tools.

The GUI is designed using XML and the business logic is written in Java. MySQL 5.0.6.0 database management system has been used to store the gullies information reported by soldiers. Web Service is used to connect to mysql from Android phone by using PHP. HTML is used to create and format webpages. Webpages are used to embed weather and satellite information and there by getting that information on Android using WebView. The Google Maps API is used to plot the gullies locations on the Google Map.

Lines of Code (LOC) written to implement this application is 1921 which include Java, XML, PHP and HTML. Breakdown of the LOC is listed below

Code Type	Number of Lines
Java	1141
XML	732
PHP	17
HTML	31

Table 4.1 Lines of Code

A total of 230 working hours spanning over 4 months have been spent on designing and developing this application and an approximate of additional 5 hours have been spent on testing the application for its correct functionality after the application has been fully developed.

Debugging of the application throughout the development is done using Dalvik Debug Monitor Server (DDMS). DDMS provides port-forwarding services, screen capture on the device, thread and heap information on the device, logcat, process, and radio state information, incoming call and SMS spoofing, location data spoofing etc.[20]. DDMS is also used to verify the location based services implemented in the application.

Implementation of each feature in this application is explained in detail in sub sections.

4.1.1. Report a gully feature implementation

Representational state transfer (REST) architecture is used to implement this feature as we need to connect from Android phone to mysql database server. A RESTful web service is created on web server using PHP framework. On Android application HTTPClient is used to communicate with the REST web server. JSON is used as a data format as it is easier and faster to decode than XML and also maps more nicely to objects.

4.1.2. Find a gully feature implementation

Google Maps API is used to plot the gullies locations on the Google Map. MapView allows us to use embed Google Maps in Android applications. The API provides a number of utilities for manipulating maps and adding content to the map through a variety of services, allowing us to create robust maps applications on Android devices.

Gullies data is available in the form of KML file format. So, SAX Parser is used to parse and extract the gullies data from the KML file. Gullies have been plotted on to the Google Maps using this extracted data.

4.1.3. View Weather and View Satellite Image features implementation

Weather information from weather.com is embedded on webpage using JavaScript and Satellite Image of Fort Riley is regularly updated on a webpage by people of GISSAL using ArcGIS software. In Android application, WebView is used to access these webpages.

4.2. Figures

In this section several screen shots of android emulator have been inserted to depict the working of different features in the application.

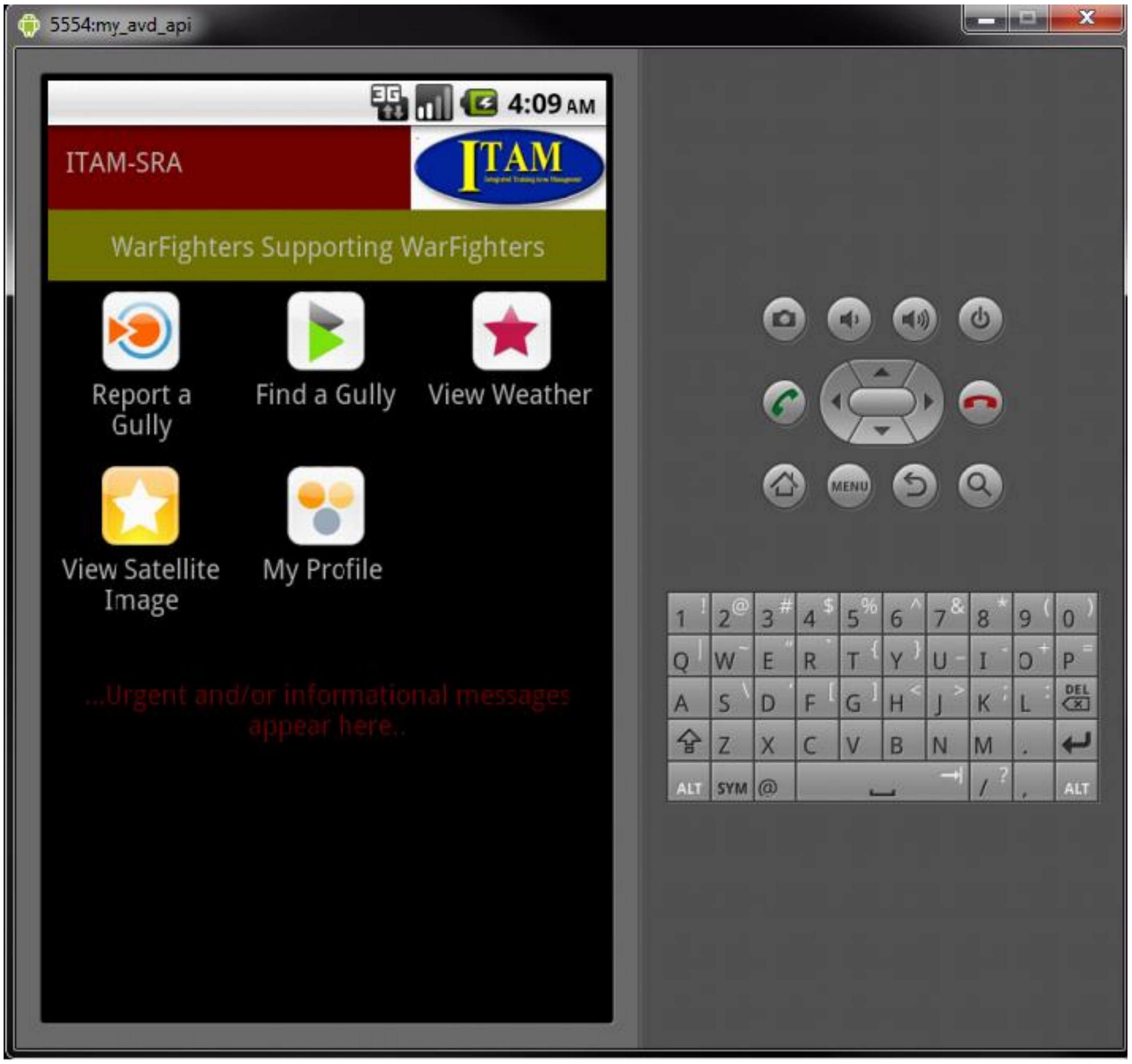
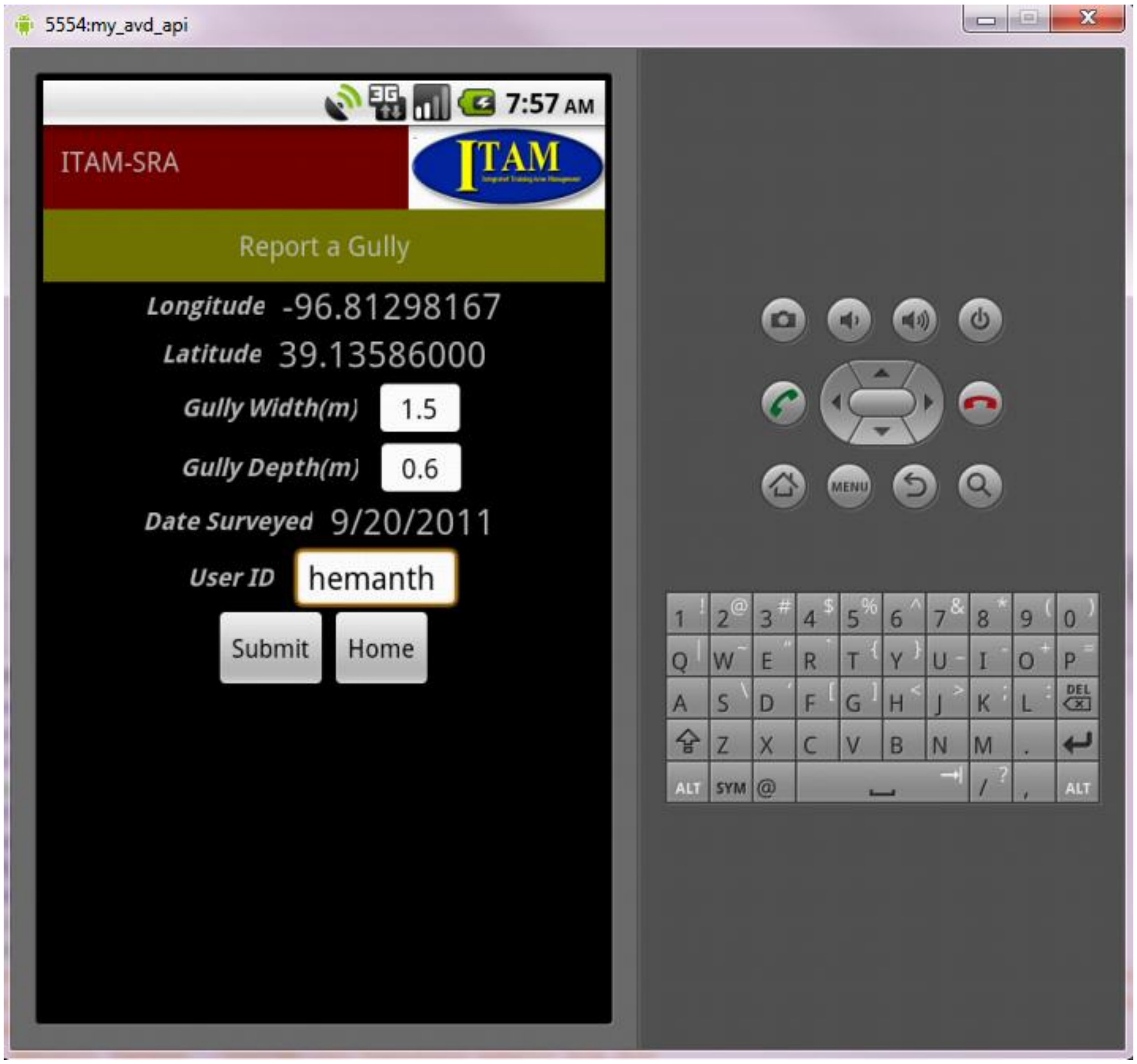


Figure 4.1 Home of ITAMSRA application

Home of the ITAMSRA application with five features is shown. At the bottom, you can also see the urgent messages being displayed.



5554:my_avd_api

7:57 AM

ITAM-SRA

Report a Gully

Longitude -96.81298167

Latitude 39.13586000

Gully Width(m) 1.5

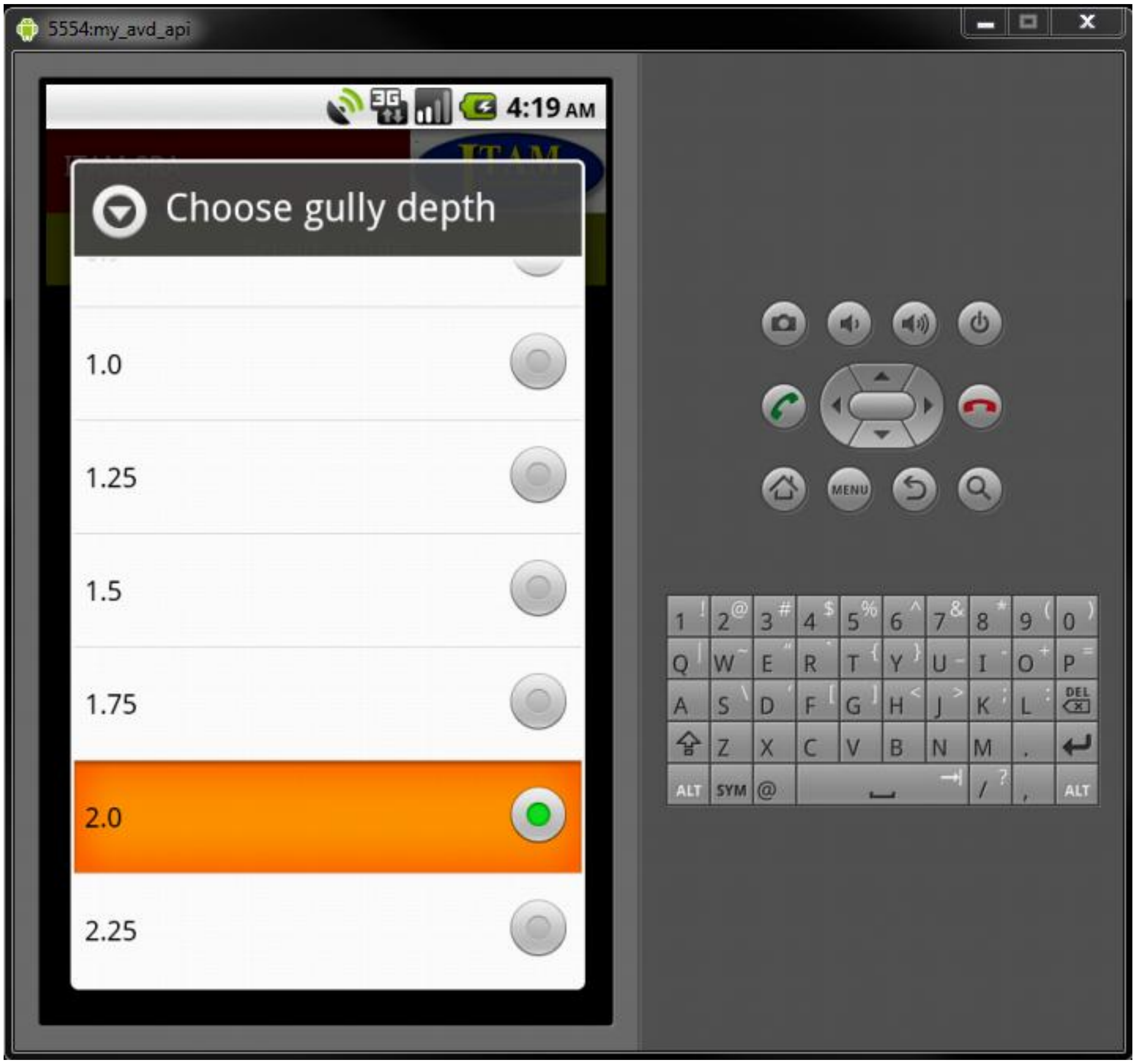
Gully Depth(m) 0.6

Date Surveyed 9/20/2011

User ID hemanth

Figure 4.2 Report a Gully-1

Report a gully feature showing the form and the fields required to be filled by soldier to report a gully. Text in Longitude, Latitude and Date Surveyed fields is non-editable.


Choose gully depth

1.0

1.25

1.5

1.75

2.0 [✓]

2.25

Figure 4.3 Report a Gully-2

Spinner is used to select the gully depth value restricting soldier to select from limited values. Maximum limit is 3.0 meters.

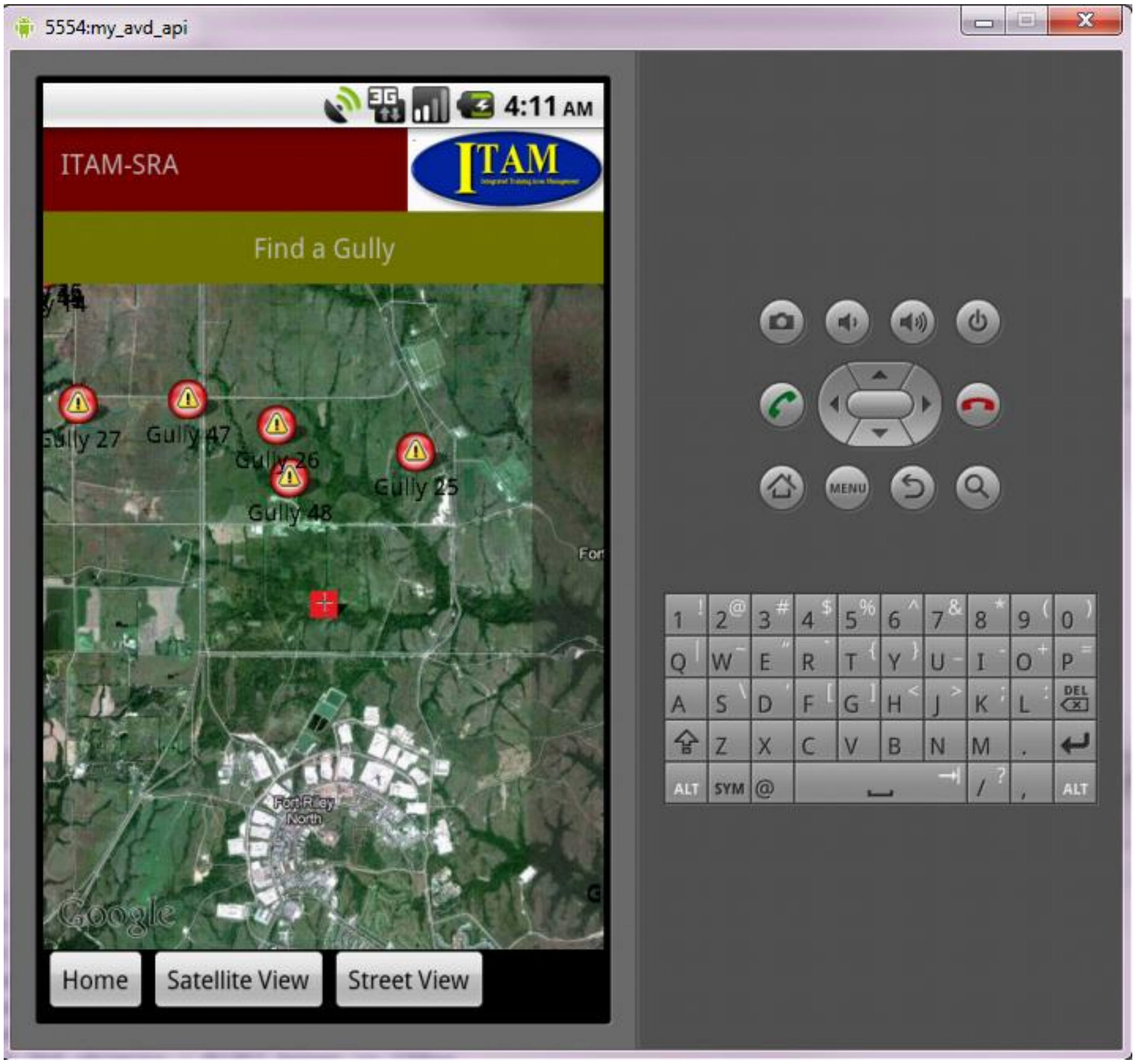


Figure 4.4 Find a Gully in Satellite View

Find a Gully feature displaying all the gullies near the current location in satellite view. Current location is marked with red color crosshair icon.

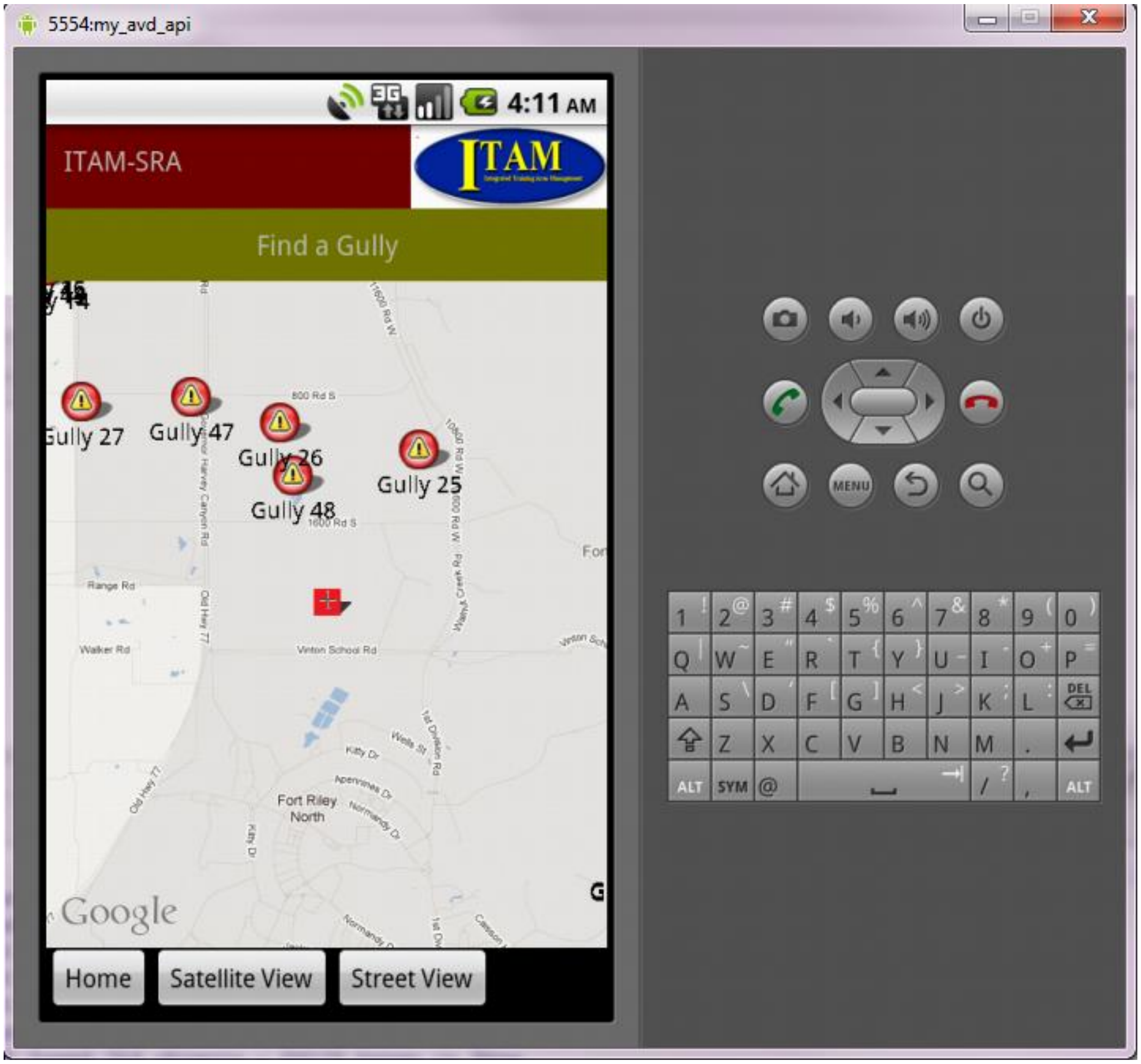


Figure 4.5 Find a Gully in Street View

Find a Gully feature displaying all the gullies near the current location in street view. Current location is marked with red color crosshair icon.

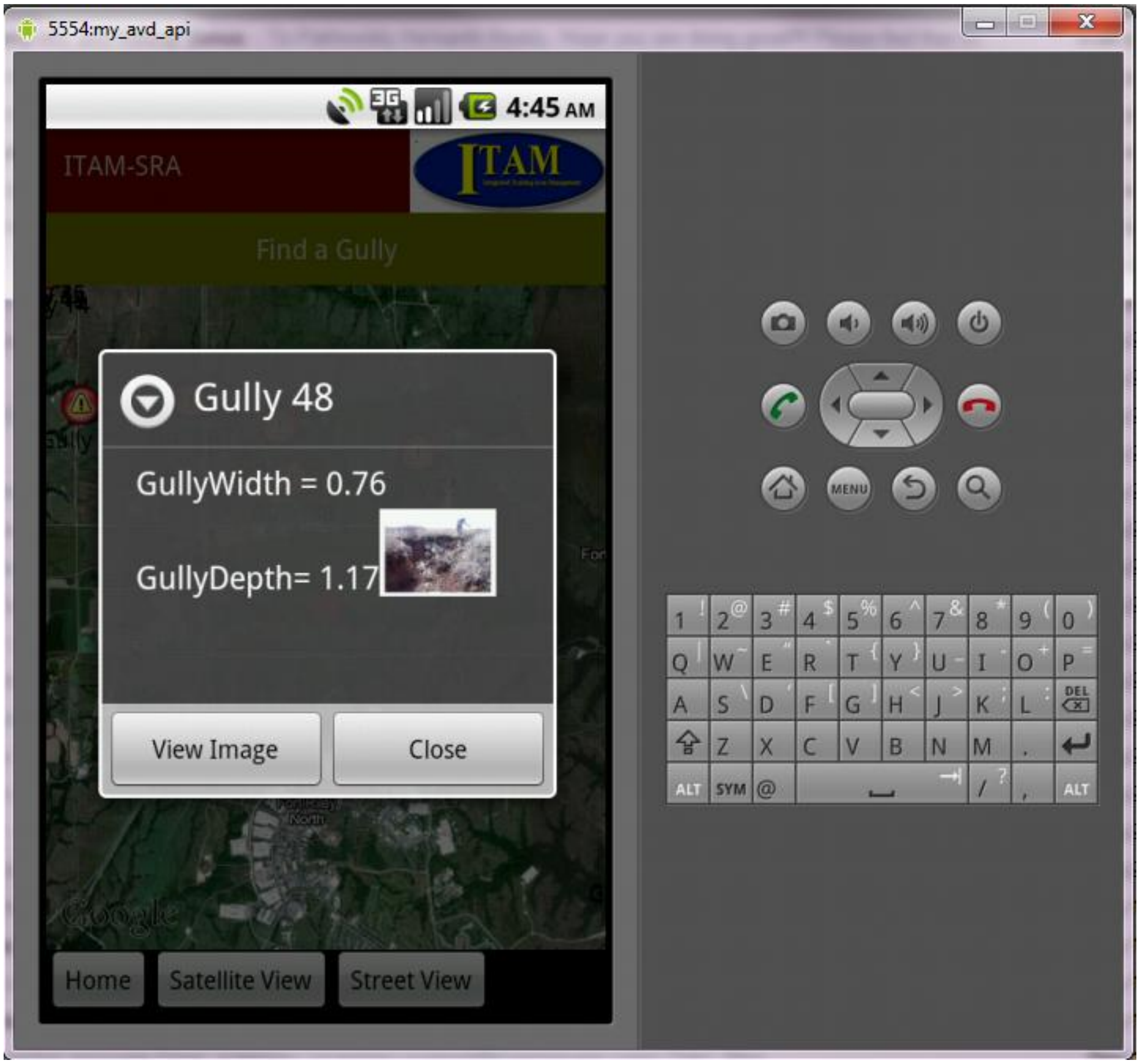


Figure 4.6 View Gully Details

Gully details like width, depth and image can be known by tapping the particular gully as shown in above figure. Large image of gully can be seen by pressing the view image button.

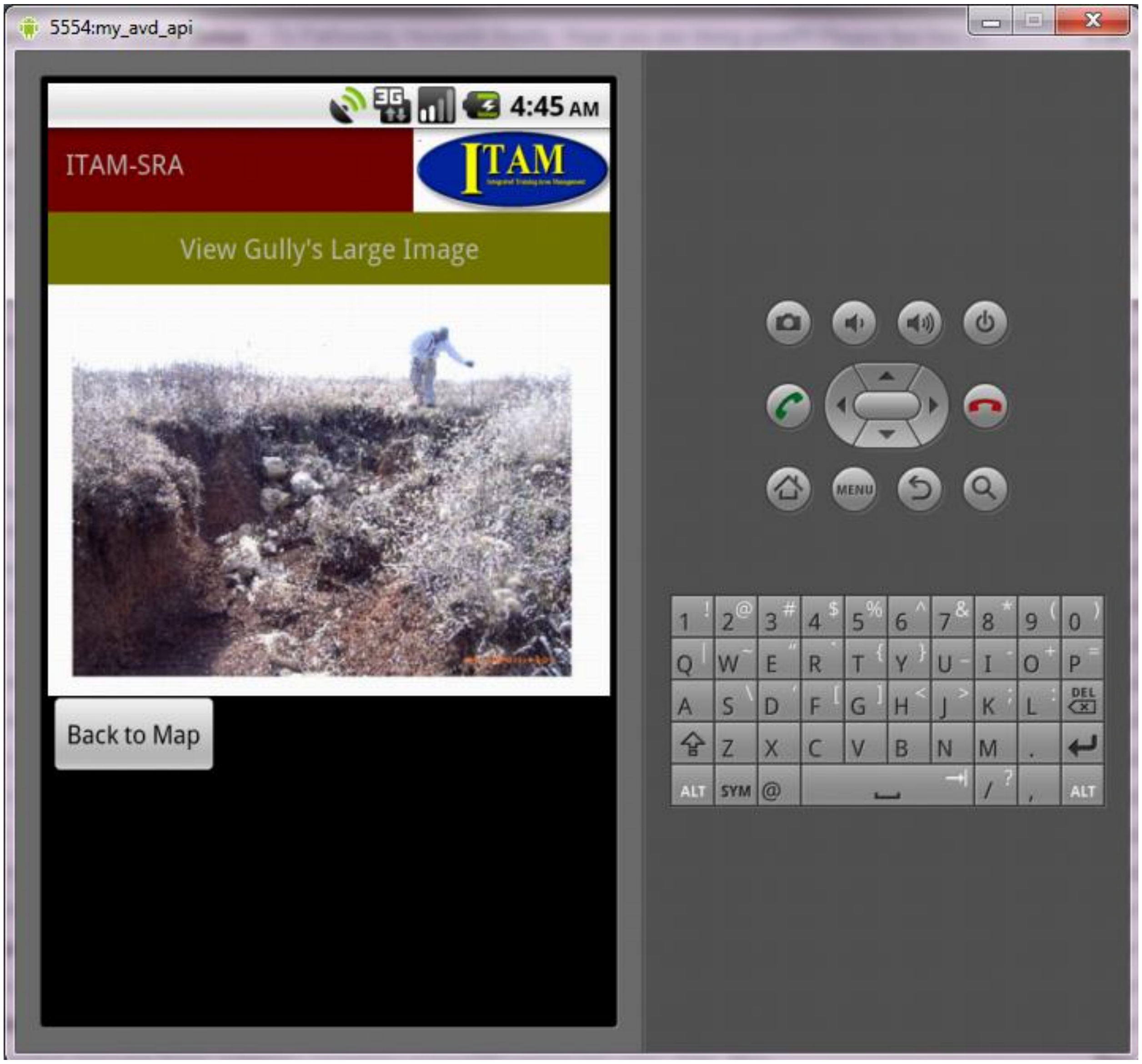


Figure 4.7 View Gully’s Large Image

Large image of the gully can be seen like shown in above figure by pressing the ‘View Image’ button shown in previous figure.

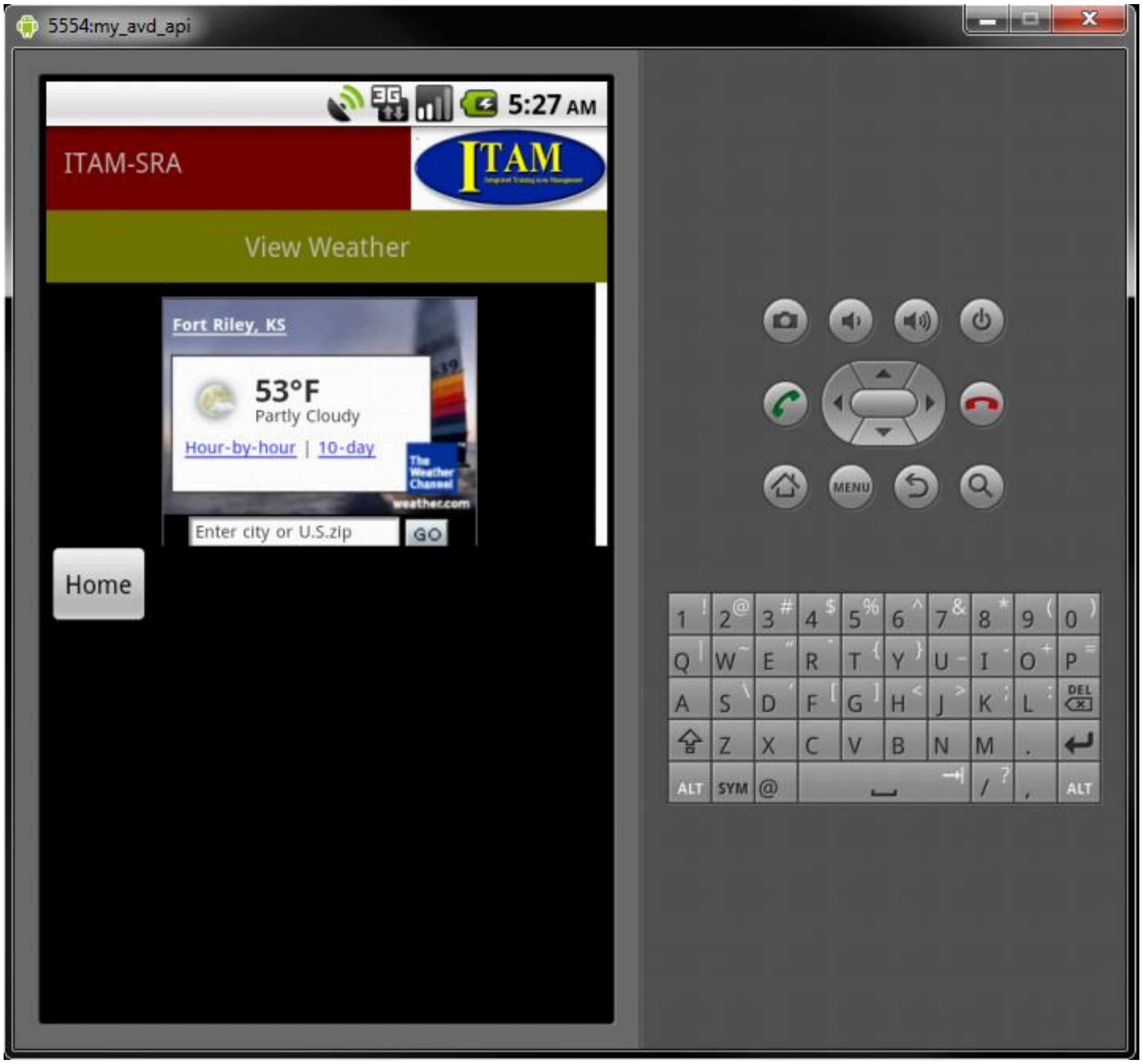


Figure 4.8 View Weather

View Weather feature showing the Fort Riley weather details like temperature and sky condition.

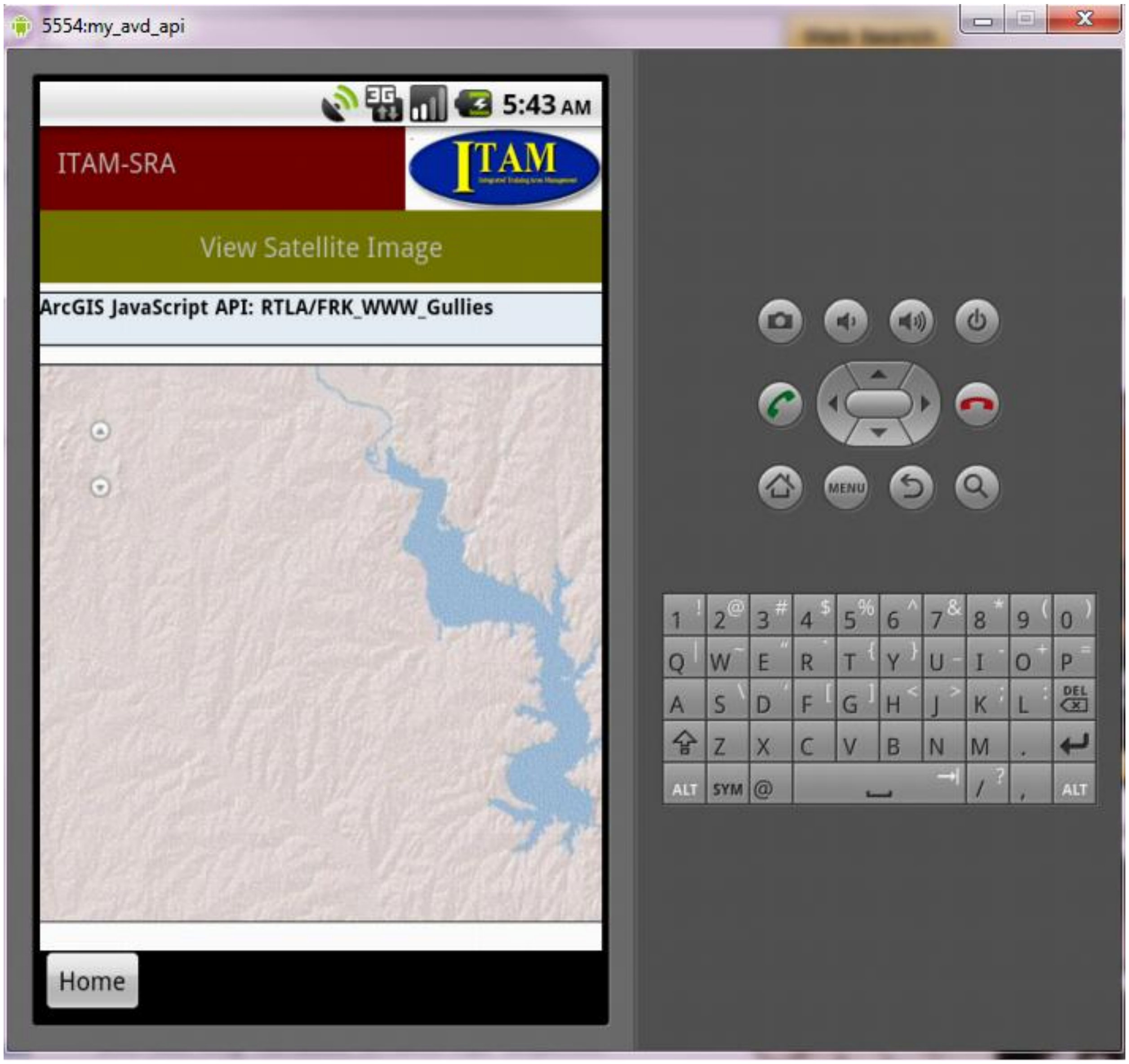


Figure 4.9 View Satellite Image

View Satellite Image feature showing the satellite image of Fort Riley. Satellite image is regularly updated by people of GISSAL using ArcGIS software.

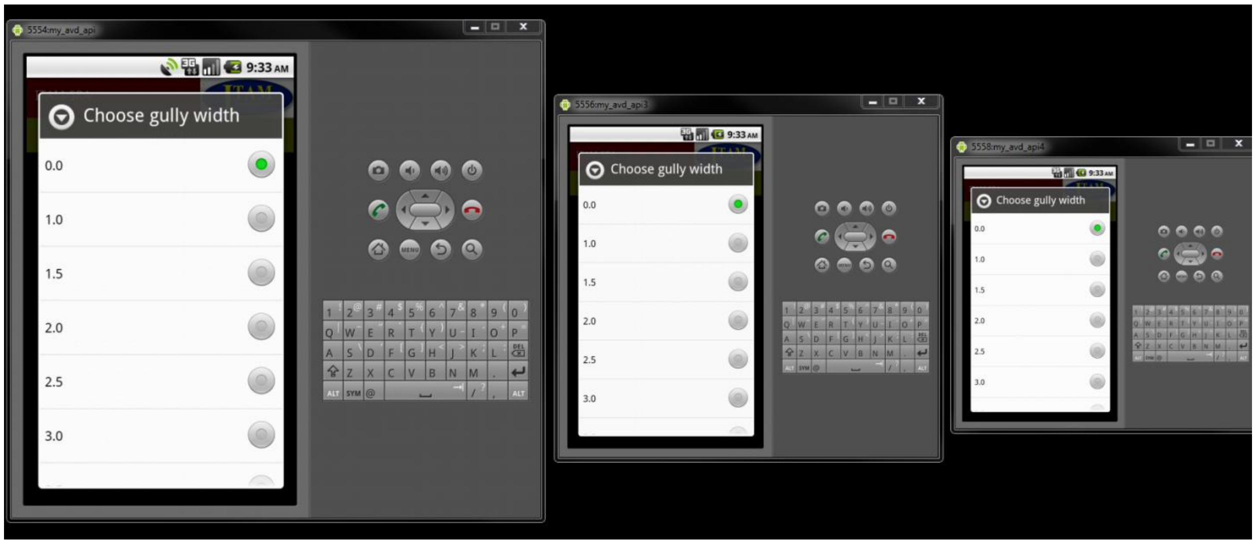


Figure 4.10 Application running on different screen sizes showing report a gully feature

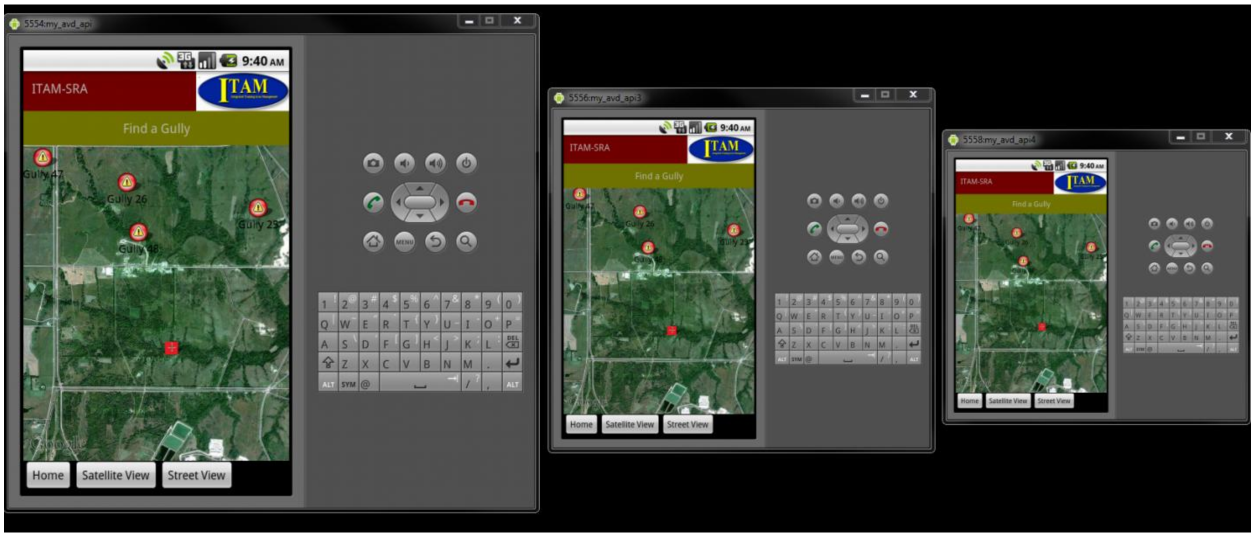


Figure 4.11 Application running on different screen sizes showing find a gully feature

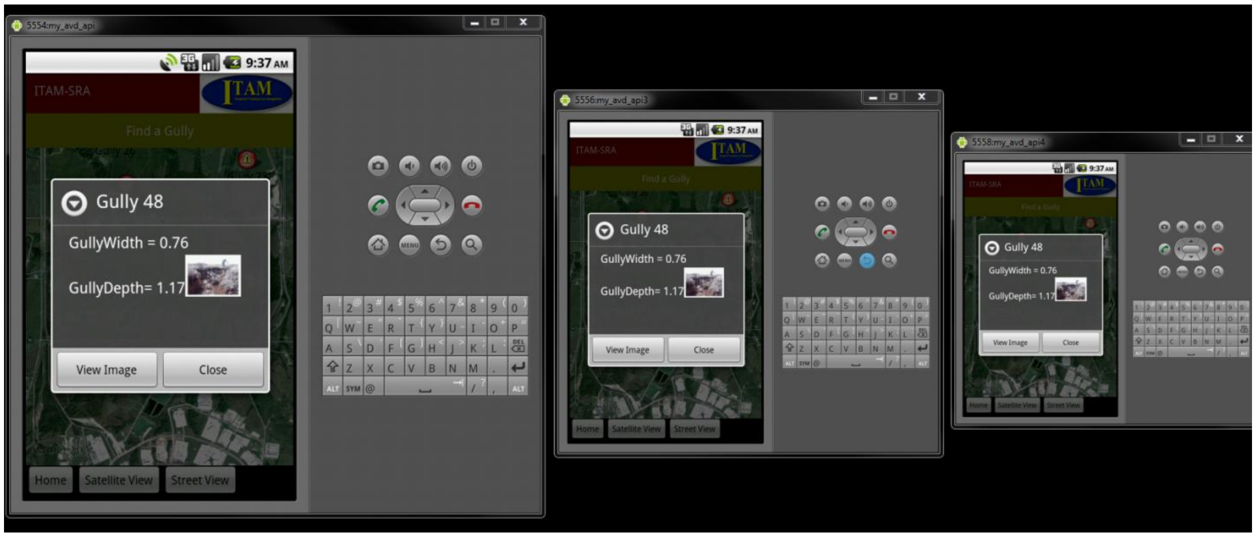


Figure 4.12 Application running on different screen sizes showing view gully details feature

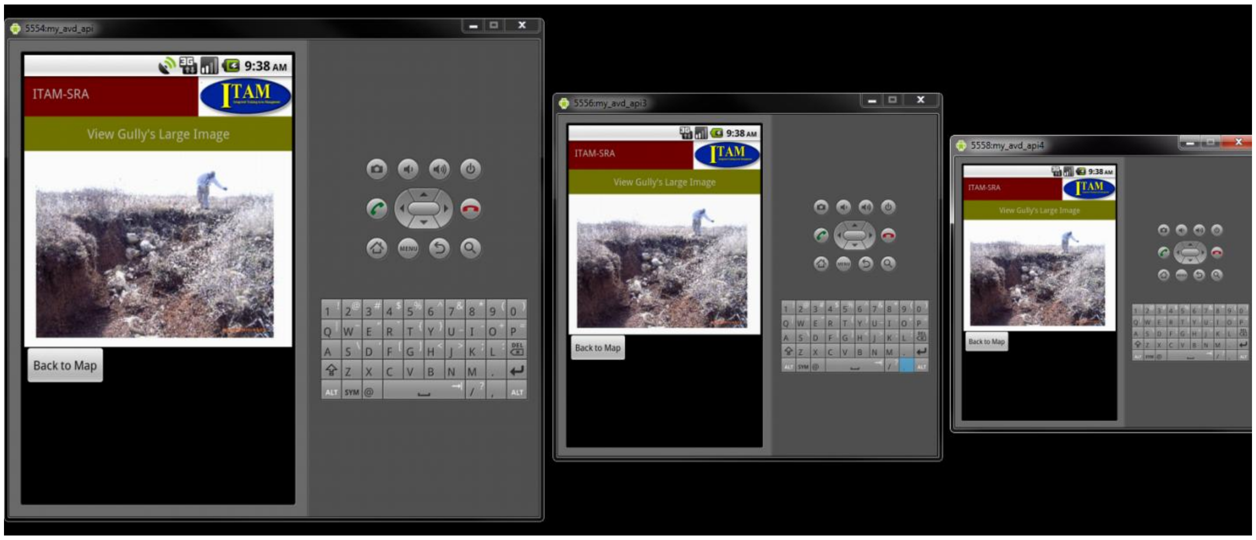


Figure 4.13 Application running on different screen sizes showing view gully's large image feature

The figures 4.10, 4.11, 4.12 and 4.13 show that the application is supported on different screen sizes. Google Map, gully details and gully images are scaled accordingly to support on different screen sizes by scaling the different layout objects used in the application.

5. CHAPTER 5

5.1. Software testing

Software testing is the process of validating and verifying that a software application meets the technical requirements which are involved in its design and development. It is also used to uncover any defects/bugs that exist in the application. It assures the quality of the software. There are many types of testing software viz., manual testing, unit testing, black box testing, performance testing, stress testing, regression testing, white box testing etc.[19] Among these performance testing and load testing are the most important one for an android application and next sections deal with some of these types.

5.2. Manual testing

Manual testing is the process of manually testing software for defects. Functionality of this application is manually tested to ensure the correctness [22]. Few examples of test case for Manual Testing are listed below:

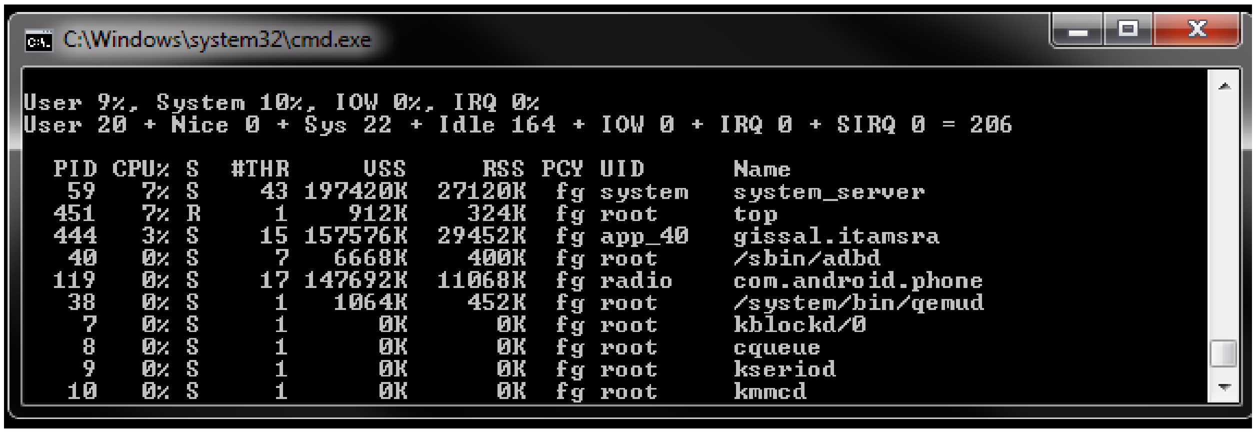
- Verify that gully's data like gully width, gully depth plotted on Google Maps is same as the data in KML file.
- Verify that gully's coordinates on Google Maps is same as the data in KML file using DDMS Emulator Control.
- Verify that data inserted into the database is same as user entered values.
- Verify that gully's image loaded on Google Maps is same as the image on the web server.

5.3. Performance testing

Performance testing is executed to determine how fast a system or sub-system performs under a particular workload. It can also serve to validate and verify other quality attributes of the system such as scalability, reliability and resource usage [23].

5.3.1. CPU Usage of application

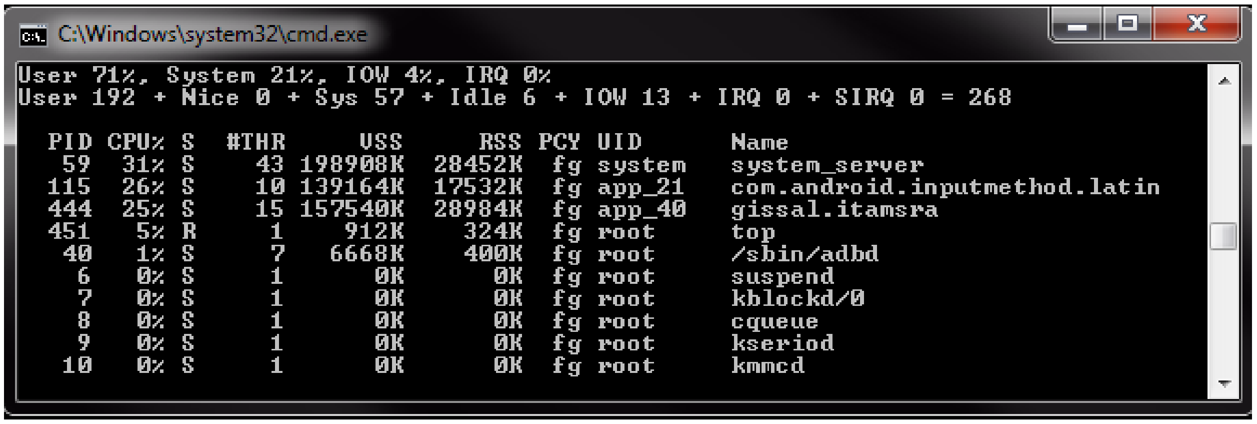
CPU usage of the application (gissal.itamsra) when deployed on Android emulator varied from 3% - 91%, but on an average CPU usage was approximately 20%.



```
User 9%, System 10%, IOW 0%, IRQ 0%
User 20 + Nice 0 + Sys 22 + Idle 164 + IOW 0 + IRQ 0 + SIRQ 0 = 206

  PID CPU% S  #THR     VSS     RSS PCY UID      Name
   59   7% S    43 197420K  27120K  fg system   system_server
  451   7% R     1    912K    324K  fg root     top
  444   3% S    15 157576K  29452K  fg app_40   gissal.itamsra
   40   0% S     7   6668K    400K  fg root     /sbin/adbd
  119   0% S    17 147692K  11068K  fg radio    com.android.phone
   38   0% S     1   1064K    452K  fg root     /system/bin/qemud
    7   0% S     1      0K      0K  fg root     kblockd/0
    8   0% S     1      0K      0K  fg root     cqueue
    9   0% S     1      0K      0K  fg root     kseriod
   10   0% S     1      0K      0K  fg root     kmmcd
```

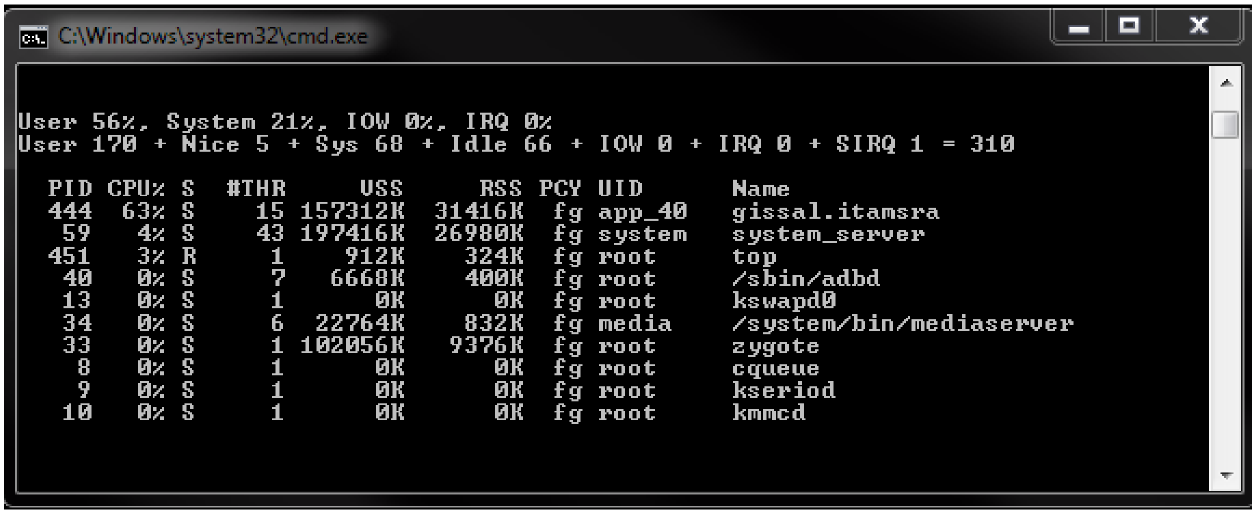
Figure 5.1 CPU usage – 3%



```
User 71%, System 21%, IOW 4%, IRQ 0%
User 192 + Nice 0 + Sys 57 + Idle 6 + IOW 13 + IRQ 0 + SIRQ 0 = 268

  PID CPU% S  #THR     VSS     RSS PCY UID      Name
   59  31% S    43 198908K  28452K  fg system   system_server
  115  26% S    10 139164K  17532K  fg app_21   com.android.inputmethod.latin
  444  25% S    15 157540K  28984K  fg app_40   gissal.itamsra
  451   5% R     1    912K    324K  fg root     top
   40   1% S     7   6668K    400K  fg root     /sbin/adbd
    6   0% S     1      0K      0K  fg root     suspend
    7   0% S     1      0K      0K  fg root     kblockd/0
    8   0% S     1      0K      0K  fg root     cqueue
    9   0% S     1      0K      0K  fg root     kseriod
   10   0% S     1      0K      0K  fg root     kmmcd
```

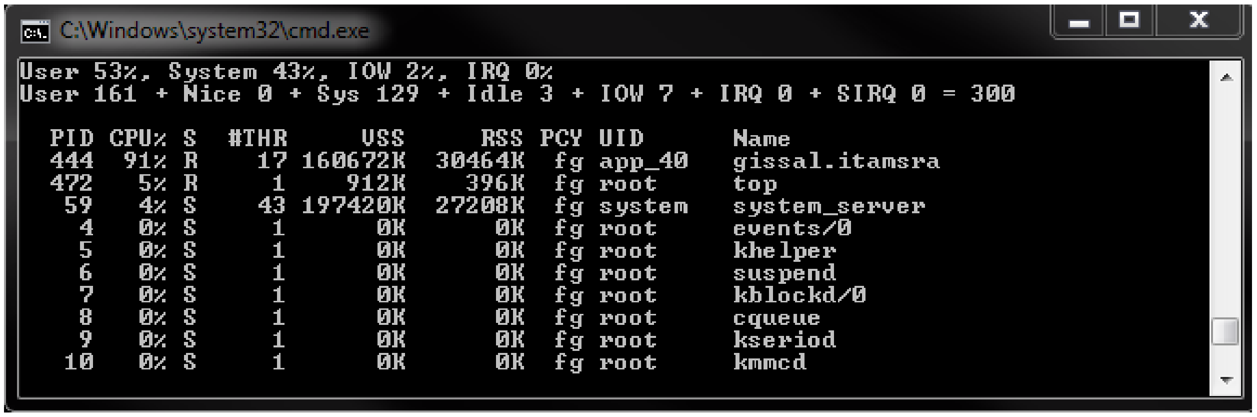
Figure 5.2 CPU usage – 25%



```
User 56%, System 21%, IOW 0%, IRQ 0%
User 170 + Nice 5 + Sys 68 + Idle 66 + IOW 0 + IRQ 0 + SIRQ 1 = 310

  PID CPU% S  #THR     VSS     RSS PCY UID      Name
  444  63% S    15 157312K  31416K  fg app_40   gissal.itamsra
   59   4% S    43 197416K  26980K  fg system   system_server
  451   3% R     1    912K    324K  fg root     top
   40   0% S     7   6668K    400K  fg root     /sbin/adbd
   13   0% S     1      0K      0K  fg root     kswapd0
   34   0% S     6  22764K    832K  fg media    /system/bin/mediaserver
   33   0% S     1 102056K   9376K  fg root     zygote
    8   0% S     1      0K      0K  fg root     cqueue
    9   0% S     1      0K      0K  fg root     kseriod
   10   0% S     1      0K      0K  fg root     kmmcd
```

Figure 5.3 CPU usage – 63%



```
User 53%, System 43%, IOW 2%, IRQ 0%
User 161 + Nice 0 + Sys 129 + Idle 3 + IOW 7 + IRQ 0 + SIRQ 0 = 300

  PID CPU% S  #THR     VSS     RSS PCY UID      Name
  444  91% R    17 160672K  30464K  fg app_40   gissal.itamsra
  472   5% R     1    912K    396K  fg root     top
   59   4% S    43 197420K  27208K  fg system   system_server
    4   0% S     1      0K      0K  fg root     events/0
    5   0% S     1      0K      0K  fg root     khelper
    6   0% S     1      0K      0K  fg root     suspend
    7   0% S     1      0K      0K  fg root     kblockd/0
    8   0% S     1      0K      0K  fg root     cqueue
    9   0% S     1      0K      0K  fg root     kseriod
   10   0% S     1      0K      0K  fg root     kmmcd
```

Figure 5.4 CPU usage – 91%

5.4. Load testing

Load testing is primarily concerned with testing that can continue to operate under specific load, whether that is large quantities of data or a large number of users.

5.4.1. Large quantities of data

Large numbers of artificial gullies are introduced into system to test the performance of 'find a gully' feature. Artificial gullies are added to the KML file. As expected load time of this feature increased with the increase in the no. of artificial gullies because map is loaded only after all gullies with details are mapped onto Google Map.

No. of gullies	Load time(in sec)
49	2.5
98	6
245	30
490	180

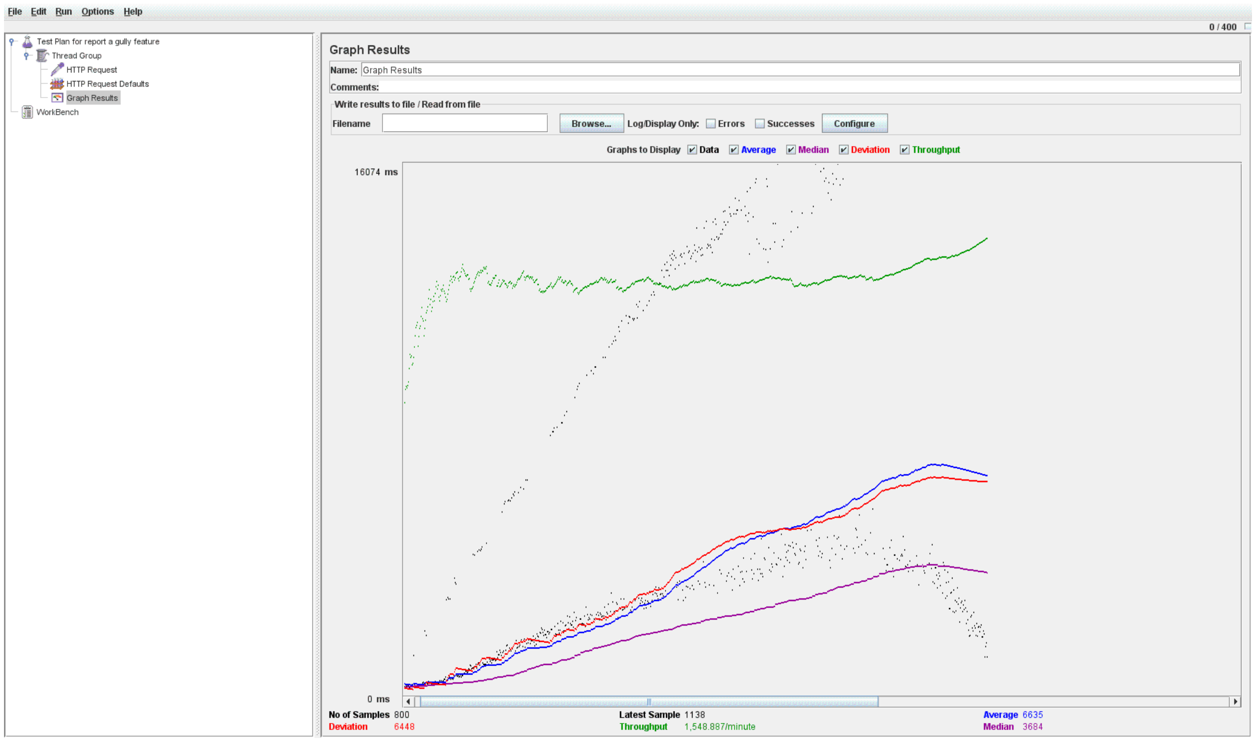
Table 5.1 Load testing (No. of gullies vs. Load time)

5.4.2. Large number of users

Large numbers of users are introduced into system to test the performance of 'report a gully' feature. JMeter is used to create these large numbers of artificial users and to analyze the performance.

Testing the report a gully feature is done by testing the performance of the PHP page used to insert gully details into MySQL database. In order to test the page, I have used a total of 400 threads, simulating 400 simultaneous users, looping 2 times, with a ramp period of 10 seconds, with a total of 800 requests.

The graph below explains all the characteristics of this PHP page along with all the statistics.



Test Plan for report a gully feature
Thread Group
HTTP Request
HTTP Request Defaults
Graph Results
WorkBench

Graph Results
Name: Graph Results
Comments:
Write results to file / Read from file
Filename
Log/Display Only: ☐ Errors ☐ Successes

```
No of Samples 800        Latest Sample 1138                   Average 6635
Deviation     6448       Throughput    1,548.887/minute       Median  3684
```

Figure 5.5 JMeter Graph Results

It is clear from the above graph that this PHP page performs consistently under heavy load generated by many simultaneous users. The throughput was around 1550, which indicates that the website can handle 1550 requests per minute. Connection to the MySQL database server for each request is one hindrance for the less number of requests per minute and more average time.

5.5. Wireless bandwidth requirement analysis

The largest data that need to be downloaded in this application from the web server is gully image. Smaller image of gully is displayed when soldier taps on the gully icon and the bigger image of gully is displayed when soldier clicks the view image button.

Size of smaller image of gullies varied from 8KB-12KB and size of bigger image of gullies varied from 80KB-100KB. So, 25KB/sec bandwidth is required for an acceptable performance with maximum load time of 4 sec for individual features.

Load time of different sub-features in report a gully feature is calculated for different bandwidths by varying network speed and network latency in emulator launch parameters and these times are listed below in the table [18].

NETWORK SPEED	NETWORK LATENCY	ACTUAL SPEED (in real world) (KB/sec)	LOAD TIME		
			Find a gully	View gully details	View gully image
GPRS	GPRS	2.5	15.48	4.55	15.65
UMTS	UMTS	15	7.86	4.17	3.47
HSPDA	NONE	50	7.25	3.88	3.30

Table 5.2 Load time vs. Network Speed of different sub-features in find a gully feature.

From the above table we can conclude that bandwidth is not a major issue for the better performance of this application unless it is very low as in the case of GPRS.

6. CHAPTER 6

6.1. Results and Challenges

The current application is developed using Android framework and can be used by all soldiers at Fort Riley military base on their Android mobile phones. It can be also used by the gully assessment team to easily find the gully locations.

CHALLENGES:

- Understanding the client requirements was one of the crucial tasks of the whole project.
- Graphic User Interface (GUI) design was a difficult task as there are many types of Android devices with varying screen size and resolutions unlike iPhone.
- Implementing Global Positioning System (GPS) and other services of Google Maps API on Android was a challenging task.
- Learning different technologies and frameworks with little guidance.

7. CHAPTER 7

7.1. Conclusions

The application has been designed successfully to meet all the user requirements. All soldiers at Fort Riley military base with Android phones can download and use this application once it is released into Android market.

7.2. Future Enhancements

The application can further be modified in the following ways:

- In report gully feature, camera feature can be implemented allowing soldier to take picture of the gully while reporting it.
- Similar to find gully and report gully features, additional features like find stream-crossing damage and report stream-crossing damage can be implemented.
- Additional feature ‘my profile’ can be implemented allowing soldiers to create, edit and login to their profiles.
- Instead of extracting gullies data from the kml file, data can be extracted directly from the mysql database by properly designing database.

8. CHAPTER 8

8.1. References

[1] Location based services on Android

http://blogs.itemis.de/frey/2009/04/04/location-based-services-on-android-part-1/

[2] Code snippets for Android development

http://stackoverflow.com/

[3] Android Development Guide

http://developer.android.com

[4] Create Icon with Text Using Grid View and Layout Inflater

http://oudomvilla.wordpress.com/2010/08/23/create-icon-with-text-using-gridview-and-layout-inflater/

[5] Icons used in the project (used only for project purpose).

http://www.fasticon.com/

[6] Google Projects for Android: Google APIs

http://code.google.com/android/add-ons/google-apis/maps-overview.html

[7] Parsing XML using SAX Parser

http://www.anddev.org/novice-tutorials-f8/parsing-xml-from-the-net-using-the-saxparser-t353.html

[8] Android development system requirements

http://developer.android.com/sdk/requirements.html

[9] Android Architecture from Android developer site

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

[10] Android architecture from Wikipedia

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

[11] Android architecture from HowStuffWorks

http://electronics.howstuffworks.com/google-phone2.htm

[12] Use Case Diagram

http://en.wikipedia.org/wiki/Use_case_diagram

[13] Unified Modeling Language

http://en.wikipedia.org/wiki/Unified_Modeling_Language

[14] Connection between PHP (server) and Android (client) Using HTTP and JSON Connection
http://fahmirahman.wordpress.com/2011/04/21/connection-between-php-server-and-android-client-using-http-and-json/

[15] Create Icon with Text Using GridView and Layout Inflater
http://oudomvilla.wordpress.com/2010/08/23/create-icon-with-text-using-gridview-and-layout-inflater/

[16] Itemized Overlay
http://code.google.com/android/add-ons/google-apis/reference/com/google/android/maps/ItemizedOverlay.html

[17] Class Diagram
http://en.wikipedia.org/wiki/Class_diagram

[18] Performance and Cost of GSM/GPRS/EDGE Solutions
http://www.lysko.com/fromPSTNtoWCDMA.htm

[19] Software Testing
http://en.wikipedia.org/wiki/Software_testing

[20] DDMS
http://developer.android.com/guide/developing/debugging/ddms.html

[21] ArcGIS
http://en.wikipedia.org/wiki/ArcGIS

[22] Manual Testing
http://en.wikipedia.org/wiki/Manual_testing

[23] Performance Testing
http://en.wikipedia.org/wiki/Software_performance_testing

[24] KML
http://en.wikipedia.org/wiki/Keyhole_Markup_Language