SMART VIDEO SURVEILLANCE SYSTEM AND IMAGE CAPTURING USING ANDROID DEVICE WITH GCM ALERT

A PROJECT REPORT

Submitted by

K.ANANDRAJ (50810205004)

S.MUTHAMIZHAN (50810205036)

S.PRAVEEN RAJ (50810205048)

In partial fulfillment for the award of the degree

of

BACHELOR OF TECHNOLOGY

in

INFORMATION TECHNOLOGY

ARUNAI COLLEGE OF ENGINEERING

TIRUVANNAMALAI :: 606 603

ANNA UNIVERSITY:: CHENNAI 600 025

APRIL 2014

ANNA UNIVERSITY : CHENNAI 600 025

BONAFIDE CERTIFICATE

Certified that this project report **" SMART VIDEO SURVEILLANCE SYSTEM AND IMAGE CAPTURING USING ANDROID DEVICE WITH GCM ALERT "** is the bonafide work of **K.ANANDRAJ (50810205004), S.MUTHAMIZHAN (50810205036) , S.PRAVEEN RAJ (50810205048)** who carried out the project work under my supervision.

SIGNATURE
Mr.N.ANANDA KUMAR,M.E.,
HOD/IT
Dept of Information Technology,
Arunai College of Engineering,
Velu Nagar, Mathur,
Tiruvannamalai-606 603.

SIGNATURE
Ms.A.SATHIYAKALA, M.Tech.
Supervisor
Dept of Information Technology,
Arunai College of Engineering,
Velu Nagar, Mathur,
Tiruvannamalai-606 603.

Submitted for VIVA-VOCE held on ……. at Arunai College of Engineering, Tiruvannamalai.

INTERNAL EXAMINER **EXTERNAL EXAMINER**

ACKNOWLEDGEMENT

We are extremely thankful to our Management for having included us to annex the golden opportunity of studying this course in this college. We wish to thank our Chairman **Mr.E.V.Velu, M.A., M.L.A.,** Vice Chairman **Er.E.V.Kumaran,M.E.,** and Managing Director **Dr.E.V.Kamban,M.D.,** for their innovative suggestion and encouragement.

We wish to thank our Secretary **Mr.R.Selvaraj** and Principal **Mr.D.Thandapani, Ph.D.,** for being a source of inspiration. We extend our heartfelt and genuine gratitude to our Dean **Dr.S.Selvakumar Raja, Ph.D.,** and our Head of Department **Mr.N.Ananda Kumar, M.E.,** and Guide **Ms.A.Sathiyakala, M.Tech.** for their valuable suggestion throughout this project.

We express our heartiest gratitude to all faculty members of **IT Department** and other Staff of our college for providing their valuable suggestion at different stage of the project.

Finally, We express our thanks to our **Parents** and **Friends** for their prayers, enthusiasm and endless support without which this project wouldn't have been completed.

TABLE OF CONTENTS

ABSTRACT

Video surveillance systems are becoming increasingly important for crime investigation and the number of cameras installed in public space is increasing. However, many cameras installed at fixed positions are required to observe a wide and complex area. In order to efficiently observe such a wide area at lower cost, mobile robots are an attractive option. According to the result of moving object detection research on video sequences, the movement of the people is tracked using video surveillance.

Usual background subtraction approaches often poorly perform on a long traffic video due to gradual changes of illumination and background shadow position. We present a fast and robust background subtraction algorithm based on unified spatio-temporal background and foreground modeling. The correlation between neighboring pixels provides high levels of detection accuracy in the dynamic background scene. Our Bayesian fusion method, which establishes the traffic scene model, combines both background and foreground models and considers prior probabilities to adapt changes of background in each frame. The moving object is identified using the Cauchy distribution model. The Cauchy distribution model will compare the current frame with the previous frame. The threshold value is calculated to find the moving image. Using threshold value the detected pixel is identified. Hence the movement of the object is identified accurately. After motion detection it will send GCM alert to the android mobile application.

LIST OF FIGURES

INTRODUCTION

CHAPTER 1
INTRODUCTION

Video Surveillance systems have increase the user needs of dynamism in order to allow the different users (operators and administrators) to monitor the system selecting different QoS depending on the system status and to access live and recorded video from different localizations, for example, from their mobile devices. More concretely, in IP surveillance systems some resources involved are limited or expensive so dynamic reconfiguration could become competitive advantage for system integrator and designers able to offer flexible applications adaptable to users' needs.

Advances in programming paradigms have allowed increasing the dynamism and flexibility of distributed environments. Concretely, Service-Oriented approaches provide means of developing decoupled applications in heterogeneous networks by defining the concept of service.

A service, in the SOA context, is an entity that receives and sends messages through well-defined interfaces, allowing building more complex applications that increase the value of the system. This concept can be applied to Qos-aware (Quality of Service) systems, in order to ease the configuration and reconfiguration of applications. Besides, Android is a software stack for mobile devices that includes an operating system, middleware and applications that can be suitable for the development of the end-user surveillance application.

1.1 OBJECTIVE

To design a surveillance based service system that provide security for particular place and alert the user by sending alert message.

1.2 SCOPE OF THE PROJECT

The surveillance based service provide a secure the particular place from the unauthorized person and alert the user by sending message if unauthorized person enters.

1.3 DOMAIN INTRODUCTION

Surveillance is the monitoring of the behavior, activities, or other changing information, usually of people for the purpose of influencing, managing, directing, or protecting. Surveillance is therefore an ambiguous practice, sometimes creating positive effects, at other times negative. It is sometimes done in a surreptitious manner. It most usually refers to observation of individuals or groups by government organizations, but disease surveillance, for example, is monitoring the progress of a disease in a community.

The word *surveillance* is the French word for "watching over"; "sur" means "from above" and "veiller" means "to watch". The inverse (reciprocal) of surveillance is surveillance ("two watch from below"). The word *surveillance* may be applied to observation from a distance by means of electronic equipment (such as CCTV cameras), or interception of electronically transmitted information (such as Internet traffic or phone calls). It may also refer to simple, relatively no- or low-technology methods such as human intelligence agents and postal interception.

LITERATURE SURVEY

CHAPTER 2
LITERATURE SURVEY

2.1 ViBe: A universal background subtraction algorithm for video sequences *Olivier Barnich and Marc Van Droogenbroeck*,Member,IEEE

This project presents a technique for motion detection that incorporates several innovative mechanisms. For example, our proposed technique stores, for each pixel, a set of values taken in the past at the same location or in the neighborhood. It then compares this set to the current pixel value in order to determine whether that pixel belongs to the background, and adapts the model by choosing randomly which values to substitute from the background model. This approach differs from those based on the classical belief that the oldest values should be replaced first. Finally, when the pixel is found to be part of the background, its value is propagated into the background model of a neighboring Pixel. We describe our method in full details (including pseudo code and the parameter values used) and compare it to other background subtraction techniques. Efficiency figures show that our method outperforms recent and proven state-of-the-art methods in terms of both computation speed and detection rate. We also analyze the performance of a downscaled version of our algorithm to the absolute minimum of one comparison and one byte of memory per pixel. It appears that even such a simplified version of our algorithm performs better than mainstream techniques.

2.2 Evaluation of Background Subtraction Algorithms with Post-processing ***Donovan H. Parks and Sidney S. Fels, University of British Columbia Electrical and Computer Engineering, UBC, Vancouver, Canada***

Processing a video stream to segment foreground objects from the background is a critical first step in many computer vision applications. Background subtraction (BGS) is a commonly used technique for achieving this segmentation. The popularity of BGS largely comes from its computational efficiency, which allows applications such as human computer interaction, video surveillance, and traffic monitoring to meet their real-time goals. Numerous BGS algorithms and a number of post processing techniques that aim to improve the results of these algorithms have been proposed. In this project, we evaluate several popular, state-of-the-art BGS algorithms and examine how post-processing techniques affect their performance. Our experimental results demonstrate that post-processing techniques can significantly improve the foreground segmentation masks produced by a BGS algorithm. We provide recommendations for achieving robust foreground segmentation based on the lessons learned performing this comparative study.

2.3 ***Estévez-Ayres, P. Basanta-Val, M. García-Valls, J. A. Fisteus and L. Almeida,*** **"QoS-aware Real-Time Composition Algorithms for Service-Based Applications", IEEE Trans. on Industrial Informatics, vol 5 (3), pp. 278-288, Aug. 2009.**

This project presents a model for quality-of-service (QoS)-aware service composition in distributed systems with real-time and fault-tolerance requirements. This model can be applied in application domains like, for example, remote monitoring, control and surveillance.

Classic approaches to real-time systems do not provide the flexibility and fault-tolerance required in new emerging environments that need to combine a high degree of dynamism with temporal predictability. Our approach addresses these new challenges by combining concepts from the service oriented paradigm and distributed real-time systems. We propose a concrete system model based on a holistic time-triggered-based approach for design and configuration. Based on this model, we propose two algorithms for the composition of QoS-aware service-based applications with temporal requirements: an exhaustive algorithm that computes the optimal service combination in terms of a figure of merit, suitable for offline composition; and an improved algorithm based on heuristics and partial figures of merit, suitable for online composition. Experimental results show that the latter reduces dramatically the number of combinations explored with a minimal degradation in the quality of the solution, making it feasible for online execution in dynamic environments

2.4 Dynamic Web Service Composition Problems and Solution -A Survey

Since the web has evolved as a service provider in all areas, there are few problems which are to be handled. Some challenges faced by web services are related to security, quality of service and composition. Among all the challenges, web service composition turns out to be an area of major research, because it supports business-business or enterprise application integration. With the emergence of semantic web the scope for semantic based web services composition increases as it provides better results compared to the traditional method of discovering candidate services for composition. Along with the semantics the nature of composition also needs to be dynamic as the web services and its parameters are changing frequently.

2.5 No-Heap Remote Objects for Distributed Real-Time Java

This project presents an approach to provide real-time support for Java's Remote Method Invocation (RMI) and its integration with the RTSJ memory model in order to leave out garbage collection. A new construct for remote objects, named *No-heap Remote object* (*NhRo*), is introduced. The usage of a NhRo guarantees that memory required to perform a remote invocation (at the server side) does not use heap-memory. Thus, the aim is to avoid garbage collection in the remote invocation process, improving predictability and memory isolation of distributed Java-based real-time applications. The project presents the bare model and the main programming patterns which are associated with the NhRo model. Sun RMI implementation has been modified to integrate the NhRo model in both static and dynamic environments.

2.6 Android Mobile Operating System for i.MX Applications Processor Platforms

Android is a free, open source and fully customizable mobile platform based on the Linux kernel. Android offers a full vertical software stack: an operating system, middleware and key applications. It also contains a rich set of APIs that allows third-party developers to develop great applications. Free scale now supports Android with a board support package (BSP) that is ready to be adapted to select i.MX platforms. The i.MX51 multimedia applications processor running Android is an excellent platform for building a high-performance, low-power and cost-effective mobile device that successfully passes the Android Compatibility Test Suite (CTS). The reference hardware, images, source patches and documentation are available now for the i.MX51 Evaluation Kit (EVK) at www.freescale.com/imxandroid.Freescale enables our customers with integrated hardware/software solutions to realize faster time to market, and the Android platform provides a compelling and innovative end user experience to support this effort.

2.7 Data Distribution Service for Real-Time Systems Specification

This specification describes two levels of interfaces:

•A lower DCPS (Data-Centric Publish-Subscribe) level that is targeted towards the efficient delivery of the proper information to the proper recipients.

• An optional higher DLRL (Data Local Reconstruction Layer) level, which allows for a simple integration of the Service into the application layer. The expected application domains require DCPS to be high-performance and predictable as well as efficient in its use of resources. To meet these requirements it is important that the interfaces are designed in such a way that they,

• Allow the middleware to pre-allocate resources so that dynamic resource allocation can be reduced to the minimum.

• Avoid properties that may require the use of unbounded or hard-to-predict resources, minimize the need to make copies of the data.

Even at the DCPS level, typed interfaces (i.e., interfaces that take into account the actual data types) are preferred to the extent possible. Typed interfaces offer the following advantages:

• They are simpler to use: the programmer directly manipulates constructs that naturally represent the data.

• They are safer to use. Verifications can be performed at compile time.

• They can be more efficient: the execution code can rely on the knowledge of the exact data type it has in advance, to e.g., pre-allocate resources.

It should be noted that the decision to use typed interfaces implies the need for a generation tool to translate type descriptions into appropriate interfaces and implementations that fill the gap between the typed interfaces and the generic middleware. QoS (Quality of Service) is a general concept that is used to specify the behavior of a service.

Programming service behavior by means of QoS settings offers the advantage that the application developer only indicates ‘what’ is wanted rather than ‘how’ this QoS should be achieved. Generally speaking, QoS is comprised of several QoS policies. Each QoS policy is then an independent description that associates a name with a value. Describing QoS by means of a list of independent QoS policies gives rise to more flexibility. This specification is designed to allow a clear separation between the publish and the subscribe sides, so that an application process that only participates as a publisher can embed just what strictly relates to publication. Similarly, an application process that participates only as a subscriber can embed only what strictly relates to subscription.

2.8 Using RTSP with Firewalls, Proxies, and Other Intermediary Network Devices

This project provides information to developers and implementers about the incorporation of Real Time Streaming Protocol (RTSP) into firewalls, proxies, and other intermediary devices. The project highlights the specific protocol messages that must be examined to reallocate or de-allocate ports, and the pertinent RTSP message syntax.

SYSTEM ANALYSIS

CHAPTER 3

SYSTEM ANALYSIS

3.1 EXISTING SYSTEM

- In the existing system, the moving object is identified only by background subtraction algorithms which are not exactly doing that detection works.
- Image can be stored in the server and it can be retrieve after some time.
- Image Retrieval from the remote place is not done in the existing system.

3.2 DISADVANTAGES

- There is no accuracy in the captured image.
- The moving object cannot be detected correctly.
- No alert about the motion detection to the user.
- Image cannot be retrieve at the time of motion detection.

3.3 PROPOSED SYSTEM

- In the Proposed system, the moving object is identified using the image subtraction method.
- The background image is subtracted from the foreground image. From that the moving object is identified. Here user can detect the exact image of the moving object.

- Another advantage of this system is when an unknown image is captured by the system it will alert the user automatically by sending an SMS to user's mobile.
- User will be using Android Mobile for the Retrieval of Images from the remote place to know whether those images are important and can be ignored.

3.4 ADVANTAGES

- High accuracy in image capturing
- Send an alert to user's mobile whenever a new object is detected
- Image can be stored in the server and can be view at the time of motion detection
- User can view the image, or video clips via his Android mobile itself
- Programming paradigms have allowed increasing the dynamism and flexibility of distributed environments

SYSTEM REQUIREMENTS SPECIFICATION

CHAPTER 4

SYSTEM REQUIREMENTS SPECIFICATION

The purpose of the Software Requirement Specification is to produce the specification of the analysis task and also to establish complete information about the requirement, behavior and other constraints such as functional performance and so on. The goal of Software Requirement Specification is to completely specify the technical requirements for the software product in a concise and unambiguous manner.

4.1 Software Requirements

- Operating system : Windows XP
- Technology Used : Android 2.2
- IDE : Eclipse 3.4 (min)
- Emulators : Micro emulator 5055
- Plug-in : ADT plug-in
- Tools used : Android SDK1.2,GoogleAPIv8 or minv7

4.2 Hardware Requirements

- Processor : Pentium P4
- Motherboard : Genuine Intel
- RAM : Min 1 GB
- Hard Disk : 80 GB

4.3 LANGUAGE SPECIFICATION

This project can be implemented only in JAVA because Android supports only JAVA for user applications.

JAVA

Java is Platform Independent. Java is an object-oriented programming language developed initially by James Gosling and colleagues at Sun Microsystems. It implements a strong security model, which prevents compiled Java programs from illicitly accessing resources on the system where they execute or on the network. Popular World-Wide Web browsers, as well as some World-Wide Web servers and other systems implement Java interpreters. These are used to display interactive user interfaces, and to script behavior on these systems.

SWING

To create a Java program with a graphical user interface (GUI), Swing is essential. The Swing toolkit includes a rich set of components for building GUIs and adding interactivity to Java applications. Swing includes all the components you would expect from a modern toolkit: table controls, list controls, tree controls, buttons, and labels.

Swing is far from a simple component toolkit, however. It includes rich undo support, a highly customizable text package, integrated internationalization and accessibility support. Swing is part of the Java Foundation Classes (JFC). The JFC also include other features important to a GUI program, such as the ability to add rich graphics functionality and the ability to create a program that can work in different languages and by users with different input devices.

Swing GUI Components

The Swing toolkit includes a rich array of components: from basic components, such as buttons and check boxes, to rich and complex components, such as tables and text. Even deceptively simple components, such as text fields, offer sophisticated functionality, such as formatted text input or password field 12 behavior. There are file browsers and dialogs to suit most needs, and if not, customization is possible. If none of Swing's provided components are exactly what you need, you can leverage the basic Swing component functionality to create your own.

MYSQL Server

Microsoft SQL Server is an application used to create computer databases for the Microsoft Windows family of server operating systems. Microsoft SQL Server provides an environment used to generate databases that can be accessed from workstations, the Internet, or other media such as a personal digital assistant (PDA).

MySQL works on many different system platforms, including AIX, BSDi, FreeBSD, HP-UX, eComStation, i5/OS, IRIX, Linux, Mac OS X, Microsoft Windows, NetBSD, Novell NetWare, OpenBSD, OpenSolaris, OS/2 Warp, QNX, Solaris, Symbian, SunOS, SCO OpenServer, SCO UnixWare, Sanos and Tru64. A port of MySQL to OpenVMS also exists. MySQL is primarily an RDBMS and ships with no GUI tools to administer MySQL databases or manage data contained within the databases. Users may use the included command line tools,[citation needed] or download MySQL front-ends from various parties that have developed desktop software and web applications to manage MySQL databases, build database structures, and work with data records.

SYSTEM DESIGN SPECIFICATION

CHAPTER 5

SYSTEM DESIGN SPECIFICATION

5.1 Overall Architecture Diagram

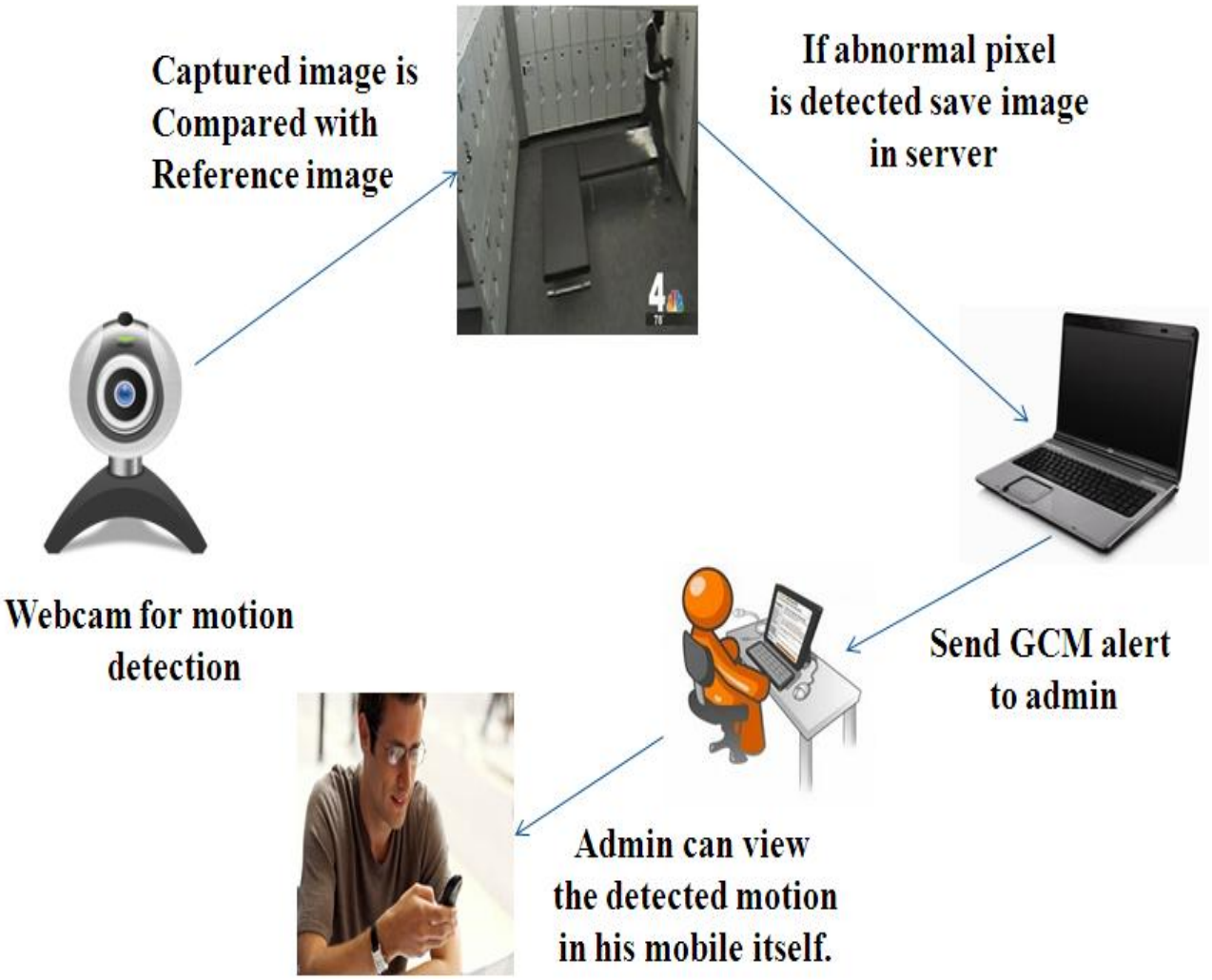


Fig.5.1 Over all Architecture Diagram

5.2 DATAFLOW DIAGRAM

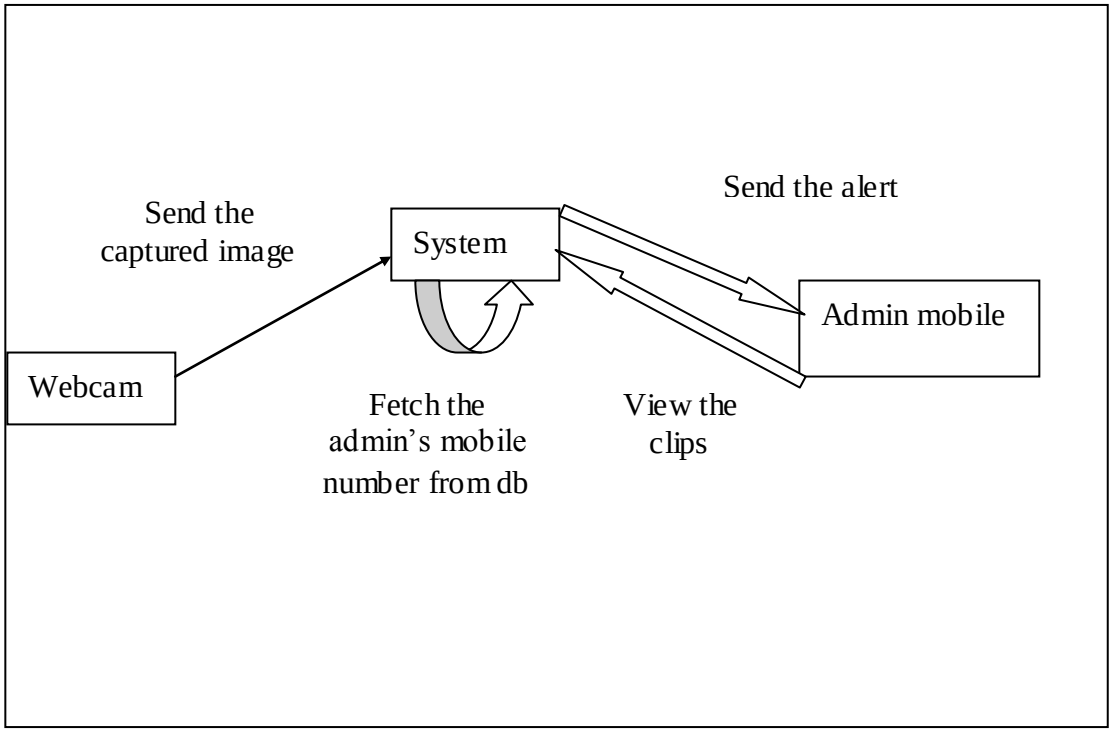


Fig.5.2 Dataflow Diagram

5.3 USECASE DIAGRAM

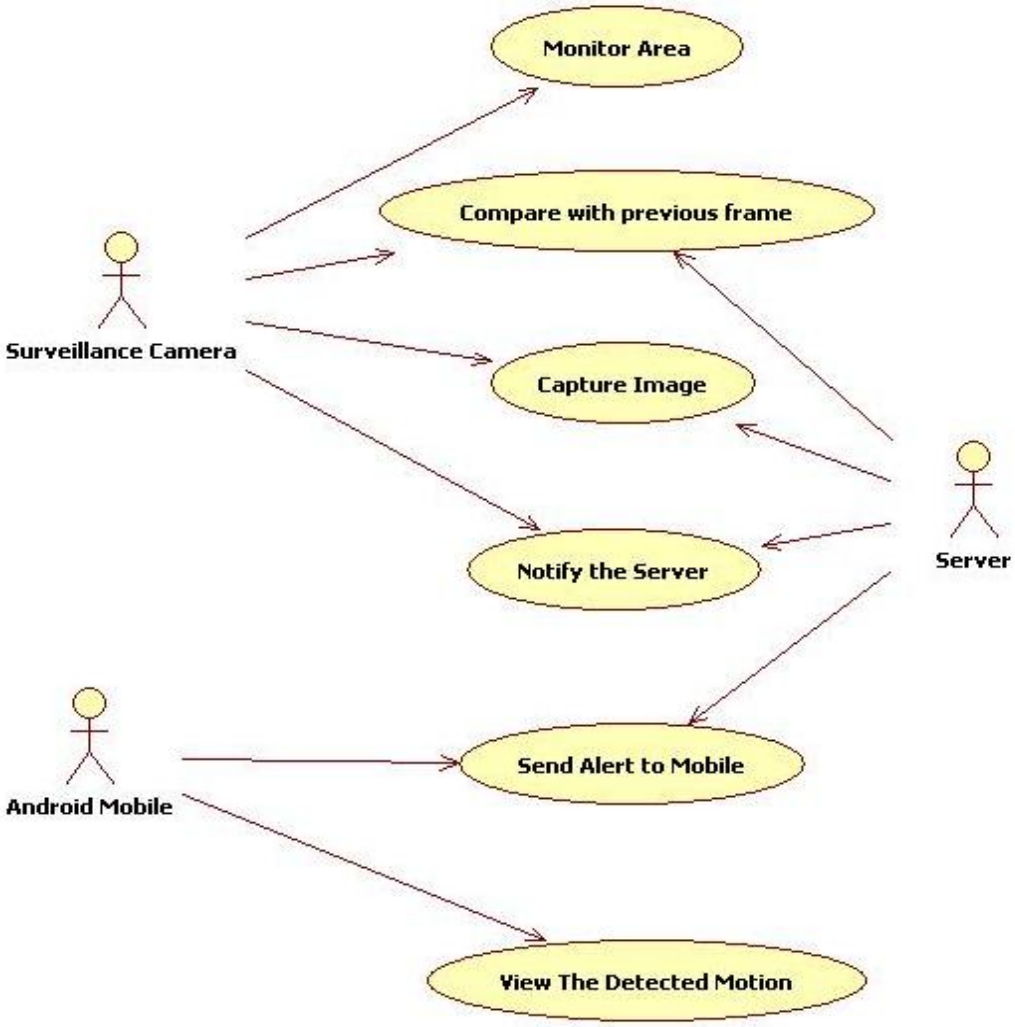


Fig.5.3 Use case Diagram

5.4 SEQUENCE DIAGRAM

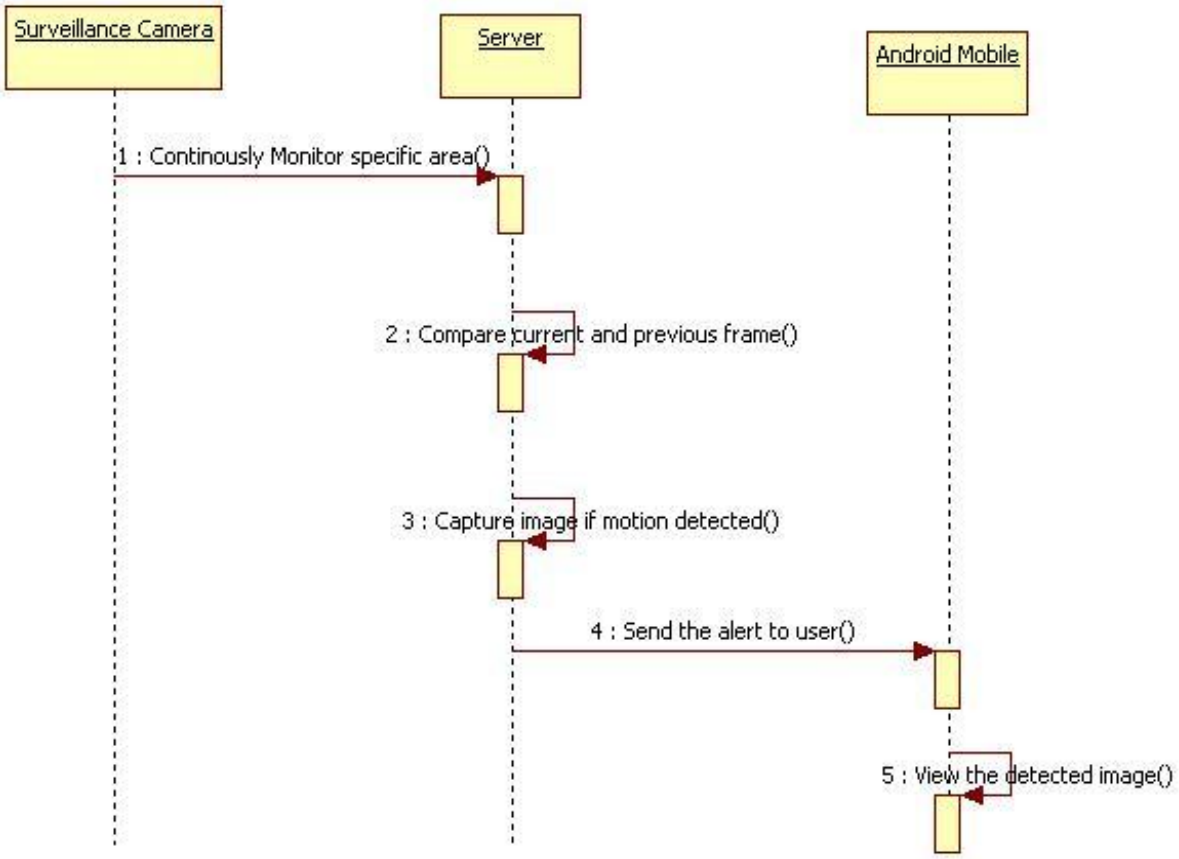


Fig.5.4 Sequence Diagram

5.5 COLLABORATION DIAGRAM

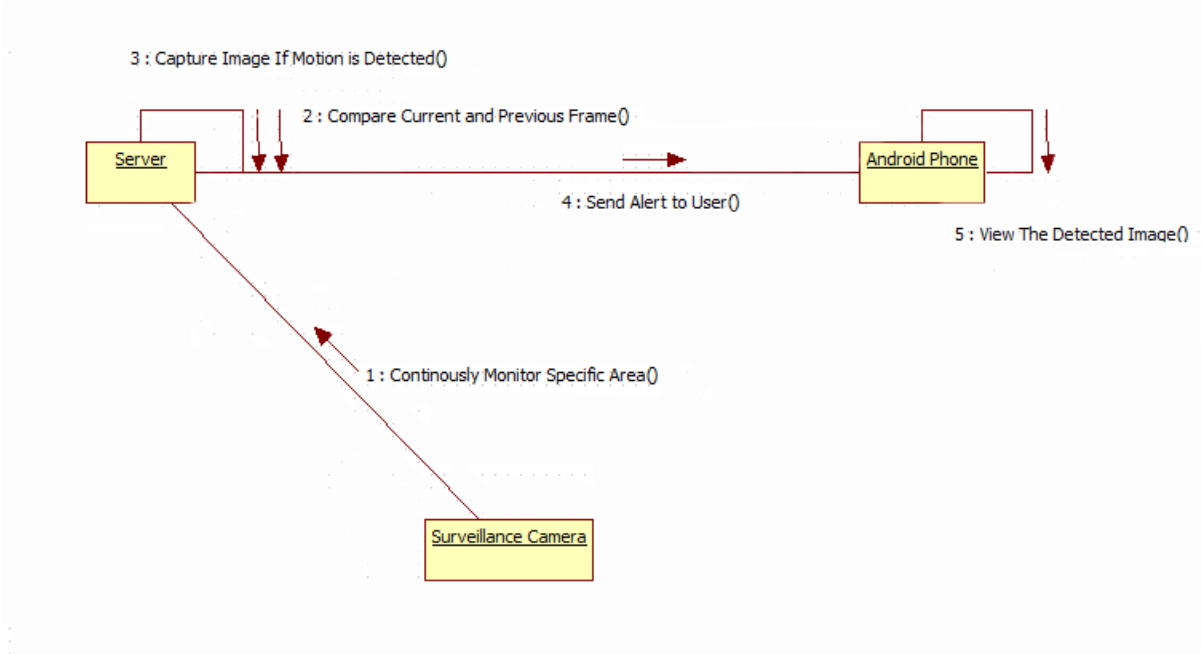


Fig.5.5 Collaboration Diagram

5.6 ACTIVITY DIAGRAM:

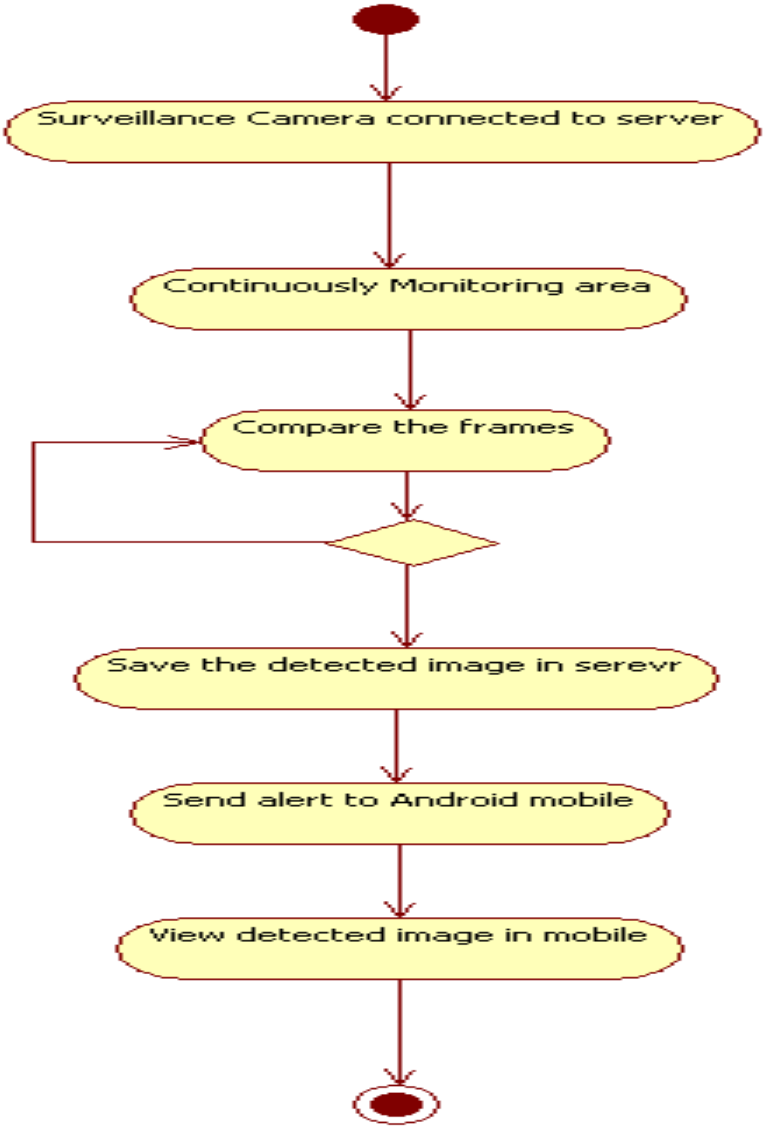


Fig.5.6 Activity Diagram

5.7 CLASS DIAGRAM

-

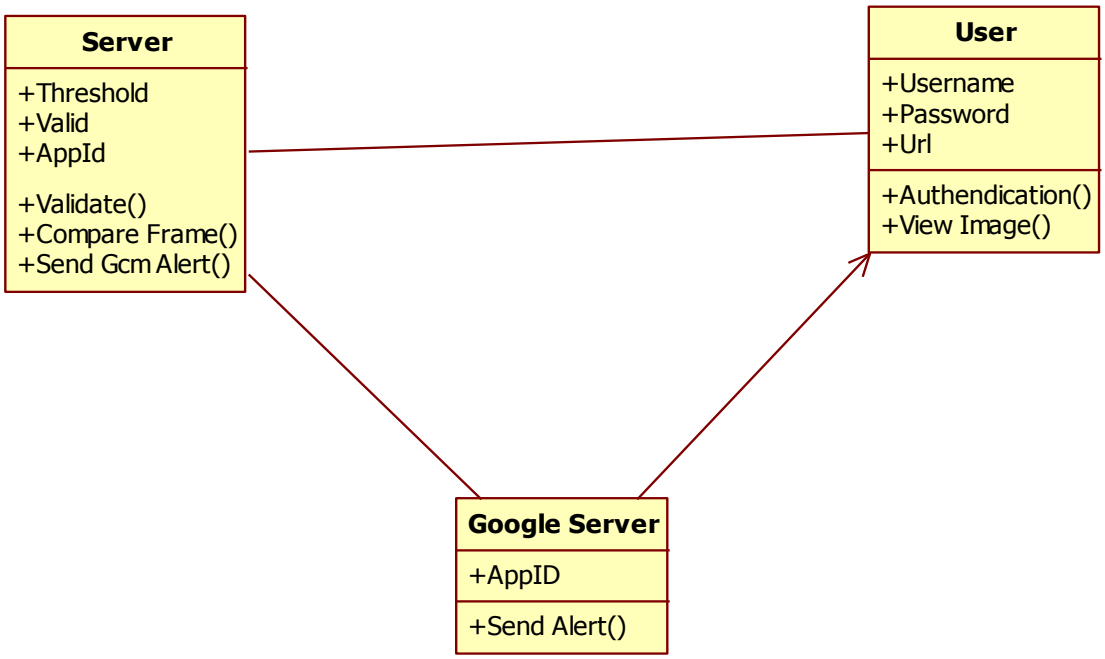


Fig.5.7 Class Diagram

SYSTEM IMPLEMENTATION

CHAPTER 6

SYSTEM IMPLEMENTATION

6.1 MODULE DESCRIPTION

- **User Registration for Application**
- **Viewing the Detected Image**
- **Detecting Motion using Cauchy Distribution Model**
- **Sending GCM Alert**

6.1.1 USER AUTHENTICATION FOR APPLICATION

User authentication is a means of identifying the user and verifying that the user is allowed to access some restricted service .The main aim of this modules is to authenticate the user to application to view the motion detected image. This modules include username and password for authentication to application The validation is based on web service in server.

6.1.2 VIEWING THE DETECTED IMAGE

Android application will receive the notification (GCM) based on project id which is registered in Google account. Application id will unique for each application After receiving the GCM alert from the server to the application and the user needs to authenticate for the application The image can be viewed using the URL which is received from the GCM alert.

6.1.3 DETECTING IMAGE USING CAUCHY DISTRIBUTION MODEL

The Main aim of this module is to detect the motion in the particular area. The motion detection is done using Cauchy distribution model and Absolute Differential Estimation .Absolute Differential Estimation is used to compare the background frame and incoming video frame if any changes occur in incoming video frame .Cauchy distribution Model is used to detect the pixel of moving object in the detected incoming video frame.

6.1.4 SENDING GCM ALERT

Whenever motion detected that image is saved on the server and the server will notify the Google server. The Google server will send a GCM Alert to the android application user mobile who are all registered for that application. Google Cloud Messaging for Android (GCM) is a service that allows you to send data from your server to your users' Android-powered device. This could be a lightweight message telling your app there is new data to be fetched from the server (for instance, a movie uploaded by a friend), or it could be a message containing up to 4kb of payload data (so apps like instant messaging can consume the message directly).

SYSTEM TESTING

CHAPTER 7

SYSTEM TESTING

7.1 PENETRATION TESTING

Setting up the Test Environment

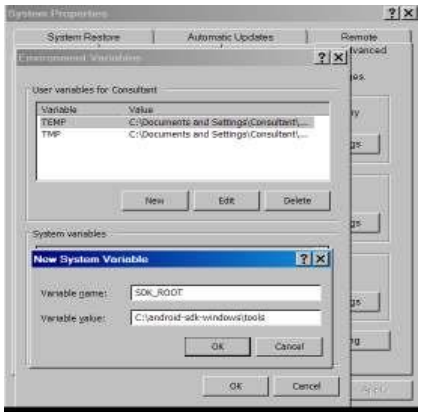
There are several ways to test mobile applications e.g.:

1. Using a regular web application penetration testing chain (browser, proxy).
2. Using WinWAP with a proxy2.
3. Using a phone emulator with a proxy3.
4. Using a phone to test and proxy outgoing phone data to a PC.

This project will focus on using a phone emulator with a proxy as it is the easiest and cheapest option out there for testing mobile applications. For some platforms, this can be difficult but for Android applications, use of an emulator is easy and effective.

Requirements:

- Computer running a Microsoft Windows operating system
- Eclipse 3.5
- Android SDK 2.2
- Fiddler



Installing the Android SDK

The first step before any testing can commence is to download and install the Android SDK4. For the purposes of this project, Microsoft Windows is for testing. the computer needs to have Java 5 or 6 and Eclipse in order to install the SDK. The installation process is very easy on Microsoft Windows and is self explanatory - simply run setup.exe. Next, add the SDK_ROOT to system variables pointing to the /tools folder and add %SDK_ROOT% to the PATH variable as shown below.

Figure 1: System variables to set to avoid specifying the whole path when running Android SDK commands 4

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

Starting the Emulator

The Android emulator comes packaged with the SDK. It is a QEMU-based device-emulation tool that can be used to designing, debugging, and testing applications in an actual Android run-time environment. Before starting the emulator you need to create an Android Virtual Device (AVD). Navigate to Eclipse > Window menu > Android SDK and AVD Manager > Virtual Devices and create a new AVD with the default settings.

To start the emulator, enter the following command: emulator –avd test avd. It will look at more advanced options that you can specify with this command later in the project. It will launch the emulator as shown in the screenshot below. Next, download any Android application or create one of your own using the "App Inventor" to test with the emulator and other tools mentioned in this project.

```
C:\android-sdk-windows\tools>adb install C:\SyncClientBinary.
1144 KB/s (0 bytes in 750906.000s)
        pkg: /data/local/tmp/SyncClientBinary.apk
Success

C:\android-sdk-windows\tools>_
```

How to Install and, Uninstall Android Applications on the Emulator

User need to obtain an application's ".apk" (Android Package) file in order for user to perform penetration testing. Use the Android Debug Bridge (ADB) that comes with the SDK to install the files into the emulator.

Open a command prompt and enter the following command to install any Android Package file adb install <path of the .apk file>. If get an error message during the installation, try the following commands:

- adb kill-server
- adb start-server

If the install fails due to size constraints, restart the emulator by executing the following command emulator –partition-size 256 –memory 512 –avd testavd

```
C:\Documents and Settings\Consultant>emulator -partition-size 256 -memory 512 -a
vd testavd
```

```
C:\android-sdk-windows\tools>adb shell
# cd data
cd data
# cd app
cd app
# ls
ls
com.funambol.android-1.apk
# rm com.funambol.android-1.apk
rm com.funambol.android-1.apk
rm failed for com.funambol.android-1.apk, No such file or directory
# rm com.funambol.android-1.apk
rm com.funambol.android-1.apk
```

User can uninstall the application either using the command prompt or the emulator. To use the command prompt open the “adb shell”, navigate to the “app” folder and use the rm command to delete the “.apk” file as shown below.

```
C:\android-sdk-windows\tools>emulator -avd testavd -http-proxy http://localhost:
8888
```

Setting up a Proxy Tool

If the application is using HTTP(s), or is a website that are testing on the Android browser, the next step is to setup a proxy tool such as Fiddler or Paros. There are 4 main ways of setting up such a proxy:

1. Specify the proxy details when starting the emulator using the command below. This command is to use a proxy listening on port 8888 (the default configuration for Fiddler). If using any other proxy port (e.g. port 8080 for Paros) then change the port number. emulator –avd testavd –http-proxy http://localhost:8888

2. The second option is to specify the proxy details in the emulator APN settings as shown below.

Navigate to Home > Menu > Wireless & Networks > Mobile Networks > Access Point Names. Update the following settings:

- **Name**: Internet
- **APN**: Internet
- **Proxy**: IP address of your computer e.g. 192.168.1.3
- **Username**: <Not Set>
- **Password**: <Not Set>

Setting up a proxy tool using the APN settings of the emulator

```
C:\Documents and Settings\Consultant>adb shell
# export HTTP_PROXY=http://localhost:8888
export HTTP_PROXY=http://localhost:8888
#
```

3. The third option is to specify it using the adb shell using the export command to set an environment variable, for example:

export HTTP_PROXY=http://localhost:8888

4. The final alternative is by changing the proxy settings in the settings database from where the android web browser reads. The settings database uses SQLite.

Familiarity with basic SQL commands is recommended if plan to use this method. Change the hostname and port information appropriately is illustrated in the command below leaving everything else as is.

> adb shell # sqlite3

/data/data/com.google.android.providers.settings/databases/settings.db

sqlite> INSERT INTO

systeVALUES(99,'http_proxy','localhost:8888');

sqlite>.exit

Once you have used one of these options your proxy should start seeing requests and responses. The figure below shows Fiddler intercepting HTTP requests sent by the emulator browser. Having a web proxy intercepting requests is a key piece of the puzzle.

Fiddler intercepting requests sent by the emulator browser

Android Application Penetration Testing Toolkit

The Android SDK comes with several utilities that, although not designed specifically for security testing, could come in handy for penetration testing. In addition, there are several tools out there such as the Manifest explorer, Intent Sniffer and Intent Fuzzer that you could use as part of your toolkit as well.

Before we look at specific tools it helps to point out a useful tip when testing web applications on the Android platform - leverage the hidden debug menu. In order to get access to this menu follow the steps below:

Navigate to the Android browser in the emulator

Enter about:debug in the address bar and click

Go to Menu

More

Settings

Scroll down to the bottom to see the now enabled debug menu

The "UAString" setting lets change the User Agent string of the browser when in this menu.

Similarly, there are other settings that user can put to good use during penetration testing.

Debug menu

Android Debug Bridge (ADB)

We have already seen this tool in action when installing Android applications. It is part of the Android SDK. It has its own shell, which allows you to execute Linux commands such as ls -l. The Android Developer's

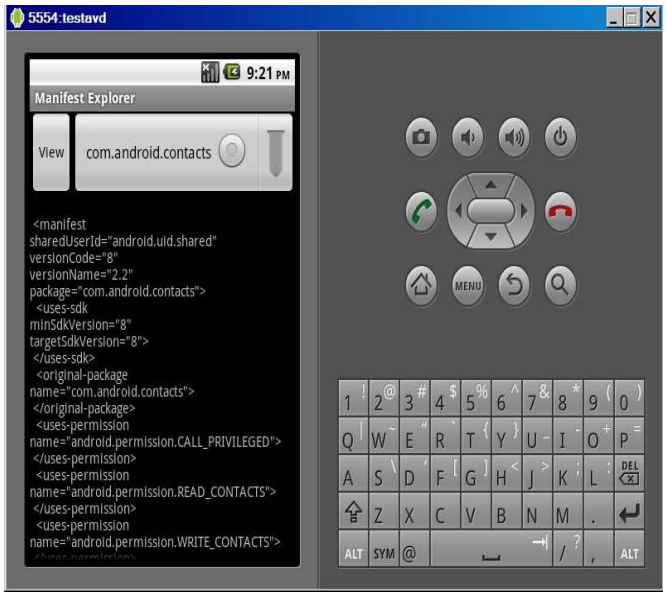
Guide5 lists the full range of ADB shell commands but we highlight a few below.

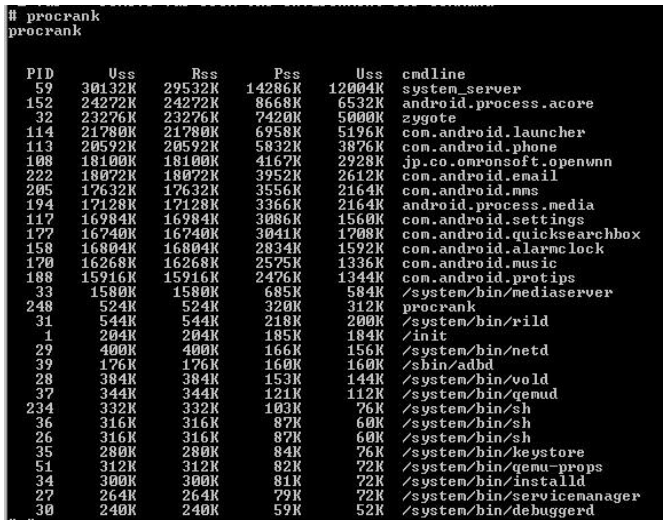
ADB could be used to locate all the emulators and Android devices connected to the computer using the command below: adb devices

In such cases the command found one instance of the emulator running. If multiple instances are running you can use the –s option in order to run commands against a specific device or emulator. adb -s emulator-5554 install Foobar.apk

Another important command provided by the ADB is to pull/push files to and from the emulator/device instance's data file. This could be useful if wants to download files from the emulator/device to your computer and review or process them. We will examine this functionality in more detail when we discuss the decompilation process.

The dumpsys or dumpstate commands can be used to dump system data to the screen or a file as shown below. This file could contain important security related information. Alternatively you could use the Dalvik Debug Monitor Service (DDMS) for this purpose.





Manifest Explorer

Every application running on Android has an AndroidManifest.xml file. This file is very important from a security perspective as it defines the permissions an application requests. The Manifest Explorer tool6 is a utility that allows to review this XML file with ease. When testing it is important to verify that the application follows the principle of "least privilege" and does not use permissions that are not required for it to function.

7.2 MAINTENANCE

The objectives of this maintenance work are to make sure that the system gets into work all time without any bug. Provision must be for environmental changes which may affect the computer or software system. This is called the maintenance of the system. Nowadays there is the rapid change in the software world. Due to this rapid change, the system should be capable of adapting these changes. In this project the process can be added without affecting other parts of the system. Maintenance plays a vital role.

The system will able to accept any modification after its implementation. This system has been designed to favour all new changes. Doing this will not affect the system's performance or its accuracy. This is the final step in system life cycle. Here we implement the tested error-free system into real-life environment and make necessary changes, which runs in an online fashion.

As a rule, system testing takes, as its input, all of the "integrated" software components that have successfully passed integration testing and also the software system itself integrated with any applicable hardware system(s). The purpose of integration testing is to detect any inconsistencies between the software units that are integrated together (called assemblages) or between any of the assemblages and the hardware. System testing is a more limited type of testing; it seeks to detect defects both within the "inter-assemblages" and also within the system as a whole.

CONCLUSION

CHAPTER 8
CONCLUSION

This project introduced an approach for an effective video surveillance in the current system, this overcomes the traditional surveying where human intervention is needed and has to watch keenly for keeping track of the entire system. But now with this project a unique technique introduced which is a major advantage to the old system. Here usage of Android Smartphone's essential, in order to effectively capture the image. This project also focus a unique feature in which it send a GCM alert at once an motion is detected in surveillance area to the Android mobile. The alert contains a URL of detected image that are stored in the server. Also we are in intent to dedicate this project to many important surveillance areas so that many unwanted things can be prevented.

FUTURE ENHANCEMENT

CHAPTER 9
FUTURE ENHANCEMENT

Though this project has many added advantage, in future it will be upgrade into the next level that is not only by just viewing the captured image and also view the entire clip that happened in the surveillance area. All this will be done just at the spontaneous moment, within a seconds of the action been happened at the site.

APPENDICES

APPENDICES

APPENDIX1-SAMPLE CODE (SERVER SIDE)

MESSAGE SENDING:

```
import java.io.IOException;
public class MessageSender {
Connection conn = ConnectionManager.getConnection();
public void sendPostAlert(String filename) {
ArrayList<String> list_gcm_id = new ArrayList<String>();
try {
Statement st = conn.createStatement();
ResultSet rs = st.executeQuery("SELECT * FROM usertable");
while (rs.next()) {
list_gcm_id.add(rs.getString("gcm_id"));}
} catch (SQLException e) {
e.printStackTrace();}
System.out.println(list_gcm_id+"\n");
if (list_gcm_id.size() == 1) {
Sender sender = new Sender(
"AIzaSyB36xhJBNHfDKwoq8_qAVkJ8Wg9uLoBE2c");
Message message = new Message.Builder().collapseKey("1")
.timeToLive(3).delayWhileIdle(true).addData("filename", filename).build();
try {
Result result = sender.send(message, list_gcm_id.get(0), 3);
System.out.println(result.toString());
}
```

```
catch (IOException e) {
e.printStackTrace();
}
} else if (list_gcm_id.size() != 0 && list_gcm_id.size() > 1) {
Sender sender = new
Sender("AIzaSyB36xhJBNHfDKwoq8_qAVkJ8Wg9uLoBE2c");
Message message = new Message.Builder().collapseKey("1")
.timeToLive(3).delayWhileIdle(true)
.addData("filename", filename).build();
try {
MulticastResult multicastResult = sender.sendNoRetry(message,
list_gcm_id);
System.out.println(multicastResult.toString());
} catch (IOException e) {
e.printStackTrace();
}
}
}
```

MOTION DETECTION EFFECT:

```
import java.awt.Dimension;
public class MotionDetectionEffect implements Effect {
public static Player player = null;
public Buffer buf;
public BufferToImage btoi;
public Image img;
public int OPTIMIZATION = 0;
boolean tag=true;
public int THRESHOLD_MAX = 10000;
public int THRESHOLD_INC = 1000;
public boolean debug = false;
public int THRESHOLD_INIT = 5000;
public int blob_threshold = THRESHOLD_INIT;
private final static int INITIAL_SQUARE_SIZE = 5;
public final static Format[] supportedFormat = new Format[] {
private Format inputFormat;
private Format outputFormat;
private Format[] inputFormats;
private Format[] outputFormats;
private int[] bwPixels;
private byte[] bwData;
private boolean visualize = true;
private boolean serverActive = true;
private boolean updateRequested;
private int avg_ref_intensity;
private int avg_img_intensity;
```

```
private RGBFormat vfIn = null;
private int[] threshs = { 50, 100, 150, 200 };
private int det_thresh = threshs[1];
private int[] colors = { 0x00FF0000, 0x00FF9900, 0x00FFFF00,
0x00FFFFFF };
private int[] newImageSquares = null;
private int[] oldImageSquares = null;
private int[] changedSquares = null;
private int numberOfSquaresWide;
private int numberOfSquaresHigh;
private int numberOfSquares;
private int sqSide = INITIAL_SQUARE_SIZE;
private int sqArea = 0;
private int sqWidthLeftover = 0;
private int sqHeightLeftover = 0;
private int pixelSpace = 0;
private int imageWidth = 0;
private int imageHeight = 0;
private int imageArea = 0;
public MotionDetectionEffect() {
inputFormats = new Format[] { new RGBFormat(null,
Format.NOT_SPECIFIED,
Format.byteArray, Format.NOT_SPECIFIED, 24, 3, 2, 1, 3,
Format.NOT_SPECIFIED, Format.TRUE, Format.NOT_SPECIFIED) };
outputFormats = new Format[] { new RGBFormat(null,
Format.NOT_SPECIFIED, Format.byteArray, Format.NOT_SPECIFIED,
```

```
24, 3, 2, 1, 3, Format.NOT_SPECIFIED,
Format.TRUE,Format.NOT_SPECIFIED)
};}
public Format[] getSupportedInputFormats() {
return inputFormats;}
public Format[] getSupportedOutputFormats(Format input) {
if (input == null) {
return outputFormats;}
if (matches(input, inputFormats) != null) {
return new Format[] { outputFormats[0].intersects(input) };
} else {
return new Format[0];}}
public Format setInputFormat(Format input) {
inputFormat = input;
return input;}
public Format setOutputFormat(Format output) {
if (output == null || matches(output, outputFormats) == null)
return null;
RGBFormat incoming = (RGBFormat) output;
Dimension size = incoming.getSize();
int maxDataLength = incoming.getMaxDataLength();
int lineStride = incoming.getLineStride();
float frameRate = incoming.getFrameRate();
int flipped = incoming.getFlipped();
int endian = incoming.getEndian();
if (size == null)
return null;
```

```
if (maxDataLength < size.width * size.height * 3)
maxDataLength = size.width * size.height * 3;
if (lineStride < size.width * 3)
lineStride = size.width * 3;
if (flipped != Format.FALSE)
flipped = Format.FALSE;
outputFormat = outputFormats[0].intersects(new RGBFormat(size,
maxDataLength, null, frameRate, Format.NOT_SPECIFIED,
Format.NOT_SPECIFIED, Format.NOT_SPECIFIED,
Format.NOT_SPECIFIED, Format.NOT_SPECIFIED, lineStride,
Format.NOT_SPECIFIED, Format.NOT_SPECIFIED));
return outputFormat;}
public synchronized int process(Buffer inBuffer, Buffer outBuffer) {
int outputDataLength = ((VideoFormat)
outputFormat).getMaxDataLength();
validateByteArraySize(outBuffer, outputDataLength);
outBuffer.setLength(outputDataLength);
outBuffer.setFormat(outputFormat);
outBuffer.setFlags(inBuffer.getFlags());
byte[] inData = (byte[]) inBuffer.getData();
byte[] outData = (byte[]) outBuffer.getData();
int[] sqAvg = null;
int[] refsqAvg = null;
vfIn = (RGBFormat) inBuffer.getFormat();
Dimension sizeIn = vfIn.getSize();
int pixStrideIn = vfIn.getPixelStride();
int lineStrideIn = vfIn.getLineStride();
imageWidth = (vfIn.getLineStride()) / 3;
imageHeight = ((vfIn.getMaxDataLength()) / 3) / imageWidth;
```

```
imageArea = imageWidth * imageHeight;
int r, g, b = 0;
if (oldImageSquares == null) {
changeSqSize(INITIAL_SQUARE_SIZE);
updateRequested = true;}
System.arraycopy(inData, 0, outData, 0, outData.length);
bwPixels = new int[outputDataLength / 3];
for (int ip = 0; ip < outputDataLength; ip += 3) {
int bw = 0;
r = (int) inData[ip] & 0xFF;
g = (int) inData[ip + 1] & 0xFF;
b = (int) inData[ip + 2] & 0xFF;
bw = (int) ((r + b + g) / (double) 3);
bwPixels[ip / 3] = bw;}
if (updateRequested) {
updateRequested = false;
updateSquares();
return BUFFER_PROCESSED_OK;
} else {updateSquares();
oldNewChange();
int c = 0;
for (int i = 0; i < changedSquares.length; i++) {
if (changedSquares[i] > det_thresh) {
c++; }}
if (c > 40 && serverActive && tag) {
tag = false;
Timer t = new Timer();
```

```
t.schedule(new TimerTask(){
public void run(){
tag =true;}
},1*1000);
Date date = new Date();
DateFormat formatter = new SimpleDateFormat("MMMddHHmmss");
String s = formatter.format(date.getTime()) + ".jpg";
String filename=s;
s = "C:\\Program Files (x86)\\Apache Software Foundation\\Tomcat
7.0\\webapps\\LoginRegistration\\images\\" + s;
System.out.println(s);
try {
buf = outBuffer;
btoi = new BufferToImage((VideoFormat) buf.getFormat());
img = btoi.createImage(buf);
MessageSender send=new MessageSender();
send.sendPostAlert(filename);
saveJPG(img, s);
} catch (Exception ex) {
ex.printStackTrace();}
System.out.println("Motion detected (motion at " + c + "areas");}
if (visualize) {
for (int i = 1; i <= numberOfSquares; i++) {
if ((changedSquares[i - 1] > threshs[0])) {
if (((i % numberOfSquaresWide) != 0)&& (numberOfSquares - i) >
numberOfSquaresWide)
int begin = ((((i % numberOfSquaresWide) - 1) * sqSide) + ((i /
numberOfSquaresWide)* imageWidth * sqSide)) * 3
```

```
if (changedSquares[i - 1] > threshs[3]) {
b = (byte) (colors[3] & 0xFF);
g = (byte) ((colors[3] >> 8) & 0xFF);
r = (byte) ((colors[3] >> 16) & 0xFF);
} else if (changedSquares[i - 1] > threshs[2]) {
b = (byte) (colors[2] & 0xFF);
g = (byte) ((colors[2] >> 8) & 0xFF);
r = (byte) ((colors[2] >> 16) & 0xFF);
} else if (changedSquares[i - 1] > threshs[1])
b = (byte) (colors[1] & 0xFF);
g = (byte) ((colors[1] >> 8) & 0xFF);
r = (byte) ((colors[1] >> 16) & 0xFF);
} else {
b = (byte) (colors[0] & 0xFF);
g = (byte) ((colors[0] >> 8) & 0xFF);
r = (byte) ((colors[0] >> 16) & 0xFF);}
for (int k = begin; k < (begin + (sqSide* imageWidth * 3));
 k = k+ (imageWidth * 3)) {
for (int j = k; j < (k + (sqSide * 3)); j = j + 3) {
try {
outData[j] = (byte) b;
outData[j + 1] = (byte) g;
outData[j + 2] = (byte) r;
} catch (ArrayIndexOutOfBoundsException e) {
System.out.println("Nullpointer: j = "+ j + ". Outdata.length = "+
outData.length); System.exit(1);}}
}}}}}}
return BUFFER_PROCESSED_OK;
}
```

```
public static void saveJPG(Image img, String sav) {
BufferedImage bi = new BufferedImage(img.getWidth(null),
img.getHeight(null), 1);
Graphics2D g2 = bi.createGraphics();
g2.drawImage(img, null, null);
FileOutputStream out = null;
try { out = new FileOutputStream(sav);
} catch (FileNotFoundException io) {
System.out.println("File Not Found");}
JPEGImageEncoder encoder = JPEGCodec.createJPEGEncoder(out);
JPEGEncodeParam param = encoder.getDefaultJPEGEncodeParam(bi);
param.setQuality(0.5F, false);
encoder.setJPEGEncodeParam(param);
try {
encoder.encode(bi);
out.close();
} catch (IOException io) {
System.out.println("IOException")}}
public String getName() {
return "Motion Detection Codec"; }
public void open() {
}
public void close() {
}
public void reset()
{}
public Object getControl(String controlType) {
System.out.println(controlType);
return null; }
```

```
public Object[] getControls() {
return null;}
public Format matches(Format in, Format outs[]) {
for (int i = 0; i < outs.length; i++) {
if (in.matches(outs[i]))
return outs[i];}
return null;}
byte[] validateByteArraySize(Buffer buffer, int newSize) {
Object objectArray = buffer.getData();
byte[] typedArray;
if (objectArray instanceof byte[]) {
typedArray = (byte[]) objectArray;
if (typedArray.length >= newSize) {
return typedArray;}
byte[] tempArray = new byte[newSize];
System.arraycopy(typedArray, 0, tempArray, 0, typedArray.length);
typedArray = tempArray;
} else {
typedArray = new byte[newSize];}
buffer.setData(typedArray);
return typedArray;}
private void setPixelSpace(int newSpace) {
pixelSpace = newSpace;}
private void changeSqSize(int newSide) {
sqSide = newSide;
sqArea = newSide * newSide;
int wid = (imageWidth / sqSide);
int hei = (imageHeight / sqSide);
```

```
sqWidthLeftover = imageWidth % sqSide;
sqHeightLeftover = imageHeight % sqSide;
if (sqWidthLeftover > 0) {
wid++;}
if (sqHeightLeftover > 0) {
hei++;}
numberOfSquaresWide = wid;
numberOfSquaresHigh = hei;
numberOfSquares = wid * hei;
newImageSquares = new int[numberOfSquares];
oldImageSquares = new int[numberOfSquares];
changedSquares = new int[numberOfSquares]; }
private int averageInSquare(int startX, int startY, int sqWidth,
int sqHeight) {
int average = 0;
for (int i = 0; i < sqHeight; i = i + 1 + pixelSpace) {
for (int j = 0; j < sqWidth; j = j + 1 + pixelSpace) {
average += bwPixels[(((startY + i) * imageWidth) + (startX + j))];}}
average = average / (sqWidth * sqHeight);
return average;}
private void updateSquares() {
System.arraycopy(newImageSquares, 0, oldImageSquares, 0,
newImageSquares.length);
int sqCount = 0;
for (int j = 0; j < (imageHeight); j = j + sqSide) {
for (int i = 0; i < (imageWidth); i = i + sqSide) {
if (i <= (imageWidth - sqSide) && j <= (imageHeight - sqSide)) {
newImageSquares[sqCount] = averageInSquare(i, j, sqSide,sqSide);
```

```
} else if (i > (imageWidth - sqSide)&& j <= (imageHeight - sqSide)) {
newImageSquares[sqCount] = averageInSquare(i, j,sqWidthLeftover,
sqSide);
} else if (i <= (imageWidth - sqSide)&& j > (imageHeight - sqSide)) {
newImageSquares[sqCount] = averageInSquare(i, j,
sqSide,sqHeightLeftover);
} else if (i > (imageWidth - sqSide)&& j > (imageHeight - sqSide)) {
newImageSquares[sqCount] = averageInSquare(i, j,sqWidthLeftover,
sqHeightLeftover); }
sqCount++;}
}
}
private void oldNewChange() {
for (int i = 0; i <= (numberOfSquares - 1); i++) {
int difference = Math.abs((newImageSquares[i])- (oldImageSquares[i]));
changedSquares[i] = difference;                    }
}
public synchronized void updateModel(boolean visualize,
boolean serverActive, boolean simplified, int[] threshs,
int[] colors, int sqSide, int det_thresh) {
this.visualize = visualize;
this.serverActive = serverActive;
if (sqSide != this.sqSide)
changeSqSize(sqSide);
if (!simplified) {
System.out.println((colors == null) + " " + (this.colors == null));
System.arraycopy(colors, 0, this.colors, 0, colors.length);
System.arraycopy(threshs, 0, this.threshs, 0, colors.length);
this.det_thresh = det_thresh;
```

```
System.out.println("New det_threhsh: " + this.det_thresh);}
updateRequested = true;}
public boolean isVisual() {
return visualize;
}
public int[] getThreshholds()
{
return threshs;}
public boolean isServerActive()
{
return serverActive;
}
public int[] getColors()
{
return colors;
}
public int getSqSide()
{
return sqSide;
}
}
```

ANDROID MOBILE SIDE

DISPLAY IMAGE:

```
import java.io.IOException;
public class DisplayImage extends Activity {
Button image_btn;
EditText inputUrl;
ImageView image_view;
Bitmap b = null;
public void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
setContentView(R.layout.imageex);
inputUrl =
((EditText)findViewById(R.id.imageUrl));inputUrl.setText(DisplayImage.th
is.getResources().getString(R.string.ipaddress)+getSharedPreferences("filen
ame", Context.MODE_PRIVATE).getString("filename", ""));
inputUrl.setSingleLine();
inputUrl.setTextSize(18);
image_view = (ImageView)findViewById(R.id.imageView1);
image_btn = (Button) findViewById(R.id.getImageButton);
image_btn.setOnClickListener(new OnClickListener() {
public void onClick(View v) {
if (TextUtils.isEmpty(inputUrl.getText().toString())) {}
else {
new Thread(new Runnable() {
public void run() {
try {b = BitmapFactory.decodeStream((InputStream)new
```

```
URL(inputUrl.getText().toString()).getContent());
} catch (MalformedURLException e) {
e.printStackTrace();
} catch (IOException e) {
e.printStackTrace();
}handler.sendEmptyMessage(1);
}}).start();
}}
});}
Handler handler = new Handler() {
public void handleMessage(Message msg) {
image_view.setImageBitmap(b);}};
private Drawable ImageOperations(Context ctx, String url) {
try {
InputStream is = (InputStream) this.fetch(url);
Drawable d = Drawable.createFromStream(is, "src");
return d;
} catch (MalformedURLException e) {
e.printStackTrace();
return null;
} catch (IOException e) {
e.printStackTrace();
return null;}}
public Object fetch(String address) throws
MalformedURLException,IOException {
URL url = new URL(address);
Object content = url.getContent();
return content;}
}
```

GCM INTENT SERVER:

```
import org.apache.http.HttpResponse;
import org.apache.http.client.HttpClient;
import org.apache.http.client.methods.HttpPost;
import android.annotation.SuppressLint;
import android.app.Notification;
import android.app.NotificationManager;
import android.app.PendingIntent;
import android.content.Context;
import android.content.Intent;
import android.support.v4.app.NotificationCompat;
import android.util.Log;
import com.google.android.gcm.GCMBaseIntentService;
public class GCMIntentService extends GCMBaseIntentService {
HttpClient httpClient;
HttpPost httpPost;
HttpResponse httpResponse;
private Thread thread;
public GCMIntentService() {
super(R.string.registrationno+"");
}
protected void onError(Context arg0, String arg1) {
Log.d("Registration error", "Registration error" + arg1);}
 ("NewApi")
protected void onMessage(Context arg0, Intent arg1) {
Log.d("Gcm message", "" + arg1.getStringExtra("filename"));
arg0.getSharedPreferences("filename", Context.MODE_PRIVATE).edit()
.putString("filename", arg1.getStringExtra("filename")).commit();
```

```
NotificationManager manager = (NotificationManager) arg0
.getSystemService(NOTIFICATION_SERVICE);
Intent notificationIntent = new
Intent(getApplicationContext(),Welcome.class);
PendingIntent pendingIntent
=PendingIntent.getActivity(getApplicationContext(), 0,
notificationIntent,Intent.FLAG_ACTIVITY_NEW_TASK);
NotificationCompat.Builder builder = new NotificationCompat.Builder(
arg0).setWhen(System.currentTimeMillis())
.setContentText("Motion has been detected. Click to view")
.setContentTitle("Video Survilliance")
.setSmallIcon(R.drawable.alert).setAutoCancel(true)
.setDefaults(Notification.DEFAULT_SOUND).setTicker("Motion
detected").setContentIntent(pendingIntent);
Notification notification = builder.build();
manager.notify((int) System.currentTimeMillis(), notification);}
protected void onRegistered(Context arg0, final String arg1) {
Log.d("Registration received", "Registration received " + arg1);
Gcmpreference.saveString(GCMIntentService.this,
Gcmpreference.GCM_ID, arg1);
Log.i("reg",""+Gcmpreference.getString(GCMIntentService.this,
Gcmpreference.GCM_ID, ""));
protected void onUnregistered(Context arg0, String arg1) {
}
}
```

GCM PREFERENCE:

```
import android.content.Context;
import android.content.SharedPreferences;
import android.content.SharedPreferences.Editor;
public class Gcmpreference {
public static final String GCM_ID = "GCM_ID";
private Context context;
public static final String PREFERENCE_NAME = "VIDEO_SURV";
public static final int MODE = Context.MODE_PRIVATE;
public Gcmpreference(Context context) {
this.context = context;
}
public static void saveString(Context context, String key, String value) {
getEditor(context).putString(key, value).commit();
}
public static String getString(Context context, String key, String defValue) {
return getPreference(context).getString(key, defValue);}
public static SharedPreferences getPreference(Context context) {
return context.getSharedPreferences(Gcmpreference.PREFERENCE_NAME,
Gcmpreference.MODE);}
public static Editor getEditor(Context context) {
return getPreference(context).edit();
}
}
```

SIGNUP:

```
import java.io.BufferedReader;
public class Signup extends Activity {
Button signup_back_btn, signup_submit;
EditText signup_username, signup_password, signup_reenter_password,
email_edit, phone_edit;
boolean checkUser = false;
private Context context;
ProgressDialog pd;
HttpClient httpClient;
HttpPost httpPost;
HttpResponse httpResponse;
int len;
int i;
int num=0;
public void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
setContentView(R.layout.signup);
createWidgetId();
signup_back_btn.setOnClickListener(new OnClickListener() {
public void onClick(View v) {
finish();}});
signup_submit.setOnClickListener(new OnClickListener() {
@SuppressLint("NewApi")
public void onClick(View v) {
if (signup_username.getText().toString().isEmpty()) {
Toast.makeText(Signup.this, "Please Enter your username",
```

```
Toast.LENGTH_LONG).show();}
else if (signup_password.getText().toString().isEmpty()) {
Toast.makeText(Signup.this, "Please Enter your
password",Toast.LENGTH_LONG).show();
} else if (signup_reenter_password.getText().toString().isEmpty()) {
Toast.makeText(Signup.this,"Please Re-Enter your password",
Toast.LENGTH_LONG).show();                    }
else if (email_edit.getText().toString().isEmpty()) {
Toast.makeText(Signup.this,"Please Enter your email-
address",Toast.LENGTH_LONG).show();
}
else if(!email_edit.getText().toString().isEmpty())
{ len=email_edit.getText().toString().length();
for(i=0;i<len;i++)
{
if(email_edit.getText().toString().charAt(i)=='@'){
num++;
}}
if(num==0)
{Toast.makeText(getApplicationContext(), "Enter the valid mail.ID",
Toast.LENGTH_LONG).show();
}
else if (phone_edit.getText().toString().isEmpty()) {
Toast.makeText(Signup.this,"Please Enter your phone number",
Toast.LENGTH_LONG).show();
}
else
if(!phone_edit.getText().toString().isEmpty()&&phone_edit.getText().toStri
ng().length()!=10)
```

```
{
Toast.makeText(getApplicationContext(),"Enter The Valid
PhoneNumbere",Toast.LENGTH_LONG).show();
}else {
if
(signup_password.getText().toString().equals(signup_reenter_password.getT
ext().toString())) {
pd = ProgressDialog.show(Signup.this, "","Logging in...", false, true);
new Thread() {
public void run() {
try {
httpClient = new DefaultHttpClient();
httpPost = new
HttpPost(Signup.this.getResources().getString(R.string.loginip));
List<NameValuePair> nameValuePair = new ArrayList<NameValuePair>();
nameValuePair.add(new BasicNameValuePair("flag", "1"));
nameValuePair.add(new BasicNameValuePair("email",
email_edit.getText().toString()));
nameValuePair.add(new BasicNameValuePair("password",
signup_password.getText().toString()));
nameValuePair.add(new BasicNameValuePair("username",
signup_username.getText().toString()));
nameValuePair.add(new BasicNameValuePair("phone",
phone_edit.getText().toString()));
httpPost.setEntity(new UrlEncodedFormEntity(nameValuePair));
httpResponse = httpClient.execute(httpPost);
InputStream inputStream = httpResponse.getEntity().getContent();
InputStreamReader inputStreamReader = new
InputStreamReader(inputStream);
```

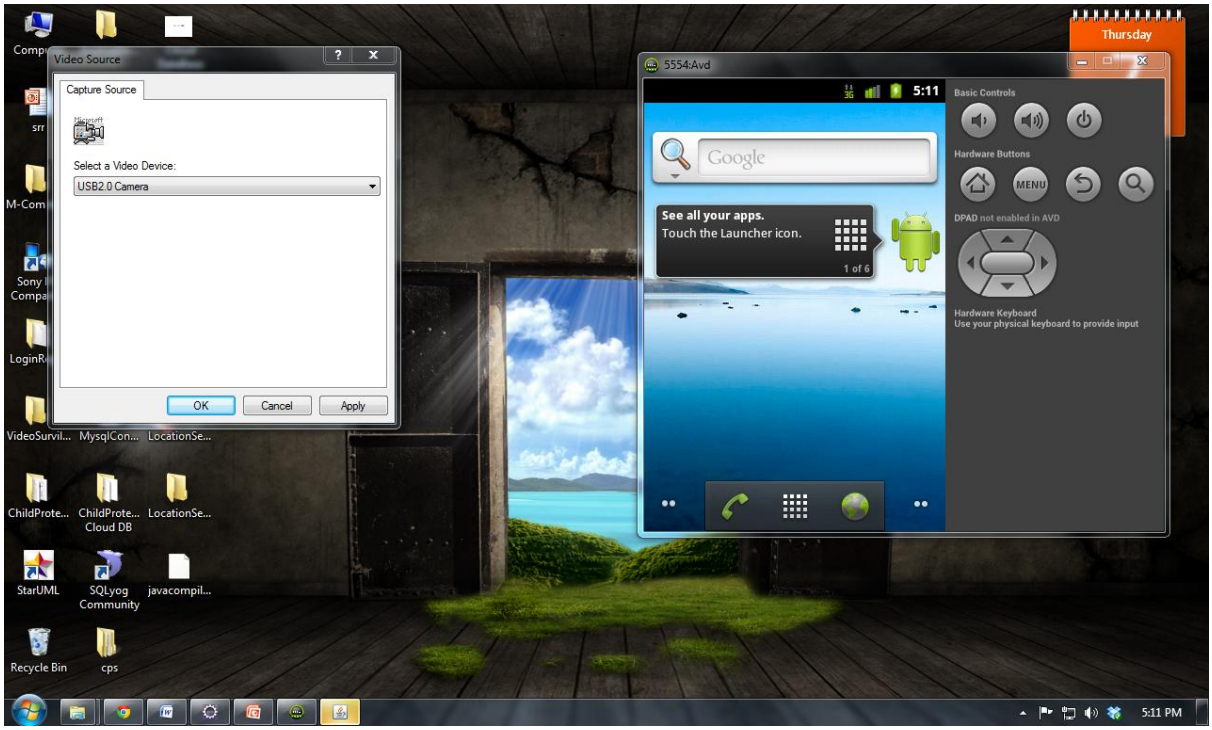
```
BufferedReader bufferedReader = new BufferedReader(inputStreamReader);
StringBuilder stringBuilder = new StringBuilder();
String bufferedStrChunk = null;
while ((bufferedStrChunk = bufferedReader.readLine()) != null)
 {
stringBuilder.append(bufferedStrChunk);
}
if (stringBuilder.toString().trim().equals("yes"))
{
handler.sendEmptyMessage(1);
}
 else
 {
handler.sendEmptyMessage(2);
}
}
catch (Exception e)
 {
handler.sendEmptyMessage(3);
}
}}
start();
}
else {
Toast.makeText(Signup.this,"Re-entered password is wrong!",
Toast.LENGTH_LONG).show();
}}}}
});
}
```

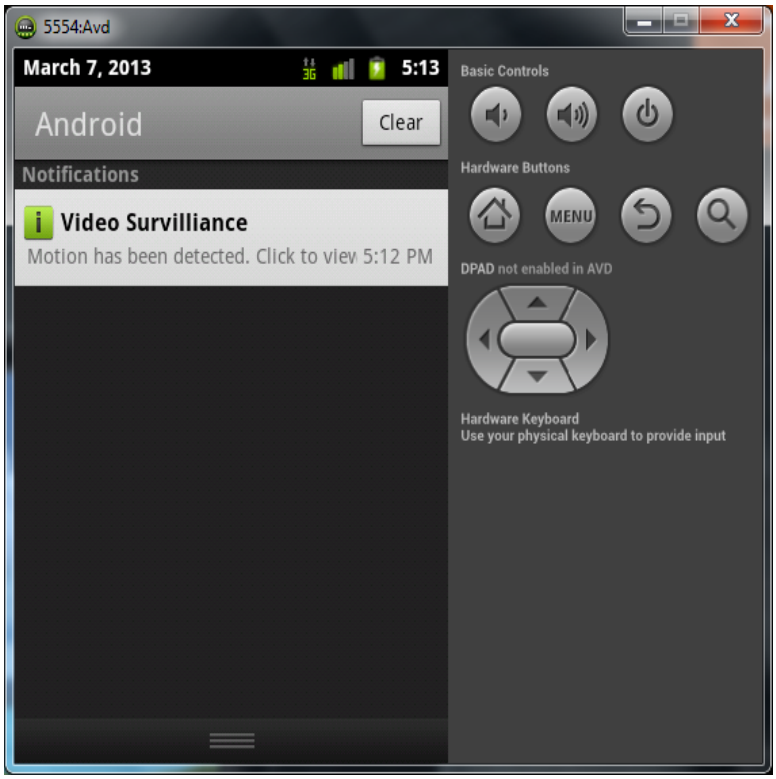
```
private void createWidgetId()
{
signup_submit = (Button) findViewById(R.id.signup_submit);
signup_back_btn = (Button) findViewById(R.id.signup_back_btn);
signup_username = (EditText) findViewById(R.id.signup_username);
signup_password = (EditText) findViewById(R.id.signup_password);
signup_reenter_password = (EditText)
findViewById(R.id.signup_reenter_password);
email_edit = (EditText) findViewById(R.id.email_edit);
phone_edit = (EditText) findViewById(R.id.phone_edit);
}
private Handler handler = new Handler() {
public void handleMessage(Message msg)
{
switch (msg.what)
{
case 1:
pd.dismiss();
AlertDialog.Builder alert = new AlertDialog.Builder(Signup.this);
alert.setCancelable(false);
alert.setTitle("Registration successfull !");
alert.setMessage("You have been successfully registered with Email");
alert.setPositiveButton("Done",new DialogInterface.OnClickListener()
{
public void onClick(DialogInterface dialog, int which)
{
GCMRegistrar.register(Signup.this, R.string.registrationno+"");
finish();}});
alert.show();
break;
```

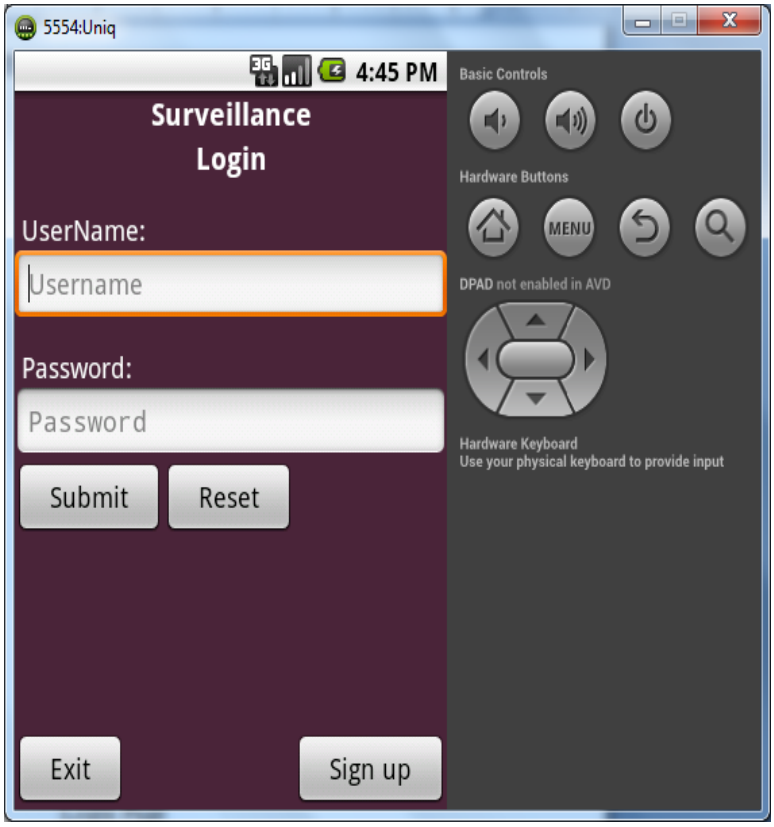
```
case 2:
pd.dismiss();
Toast.makeText(Signup.this, "Your username and password is wrong !",
Toast.LENGTH_LONG).show();
break;
case 3:
pd.dismiss();
Toast.makeText(Signup.this, "Please check your internet connection or
URL!",
Toast.LENGTH_LONG).show();
break;
}
}
};
}
```

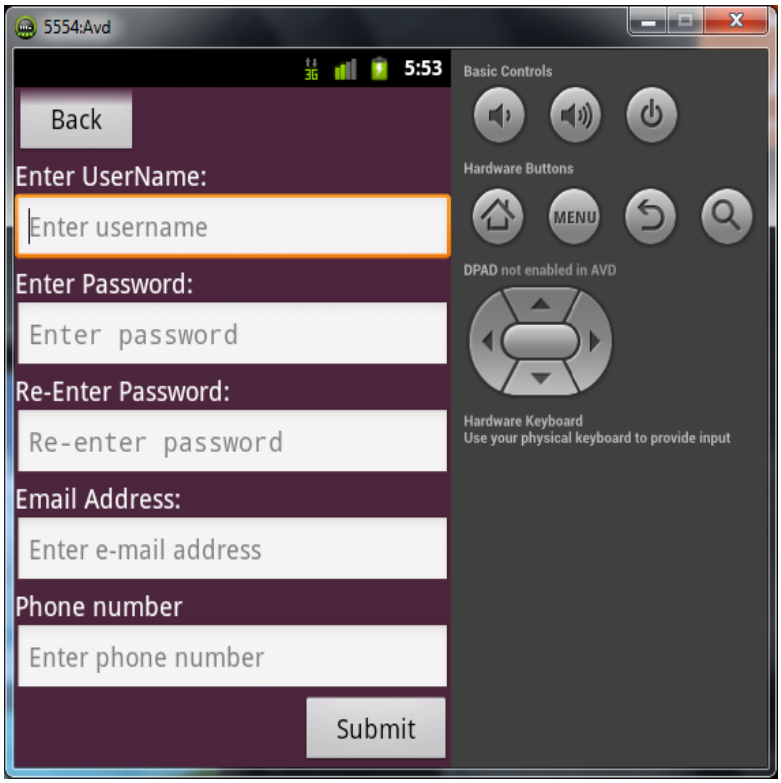
SCREEN SHOTS

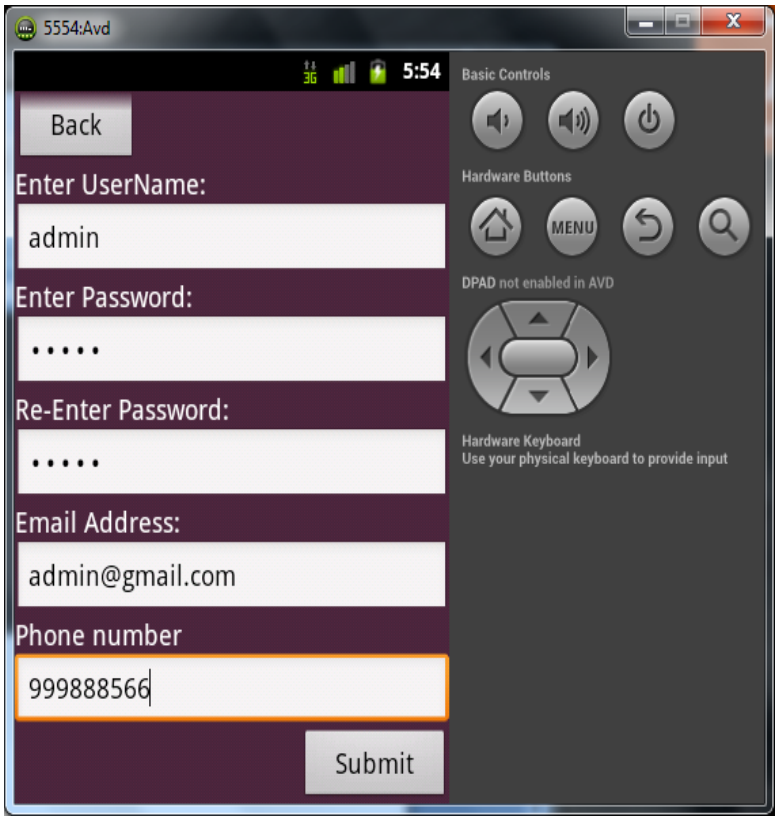
SCREEN SHOTS

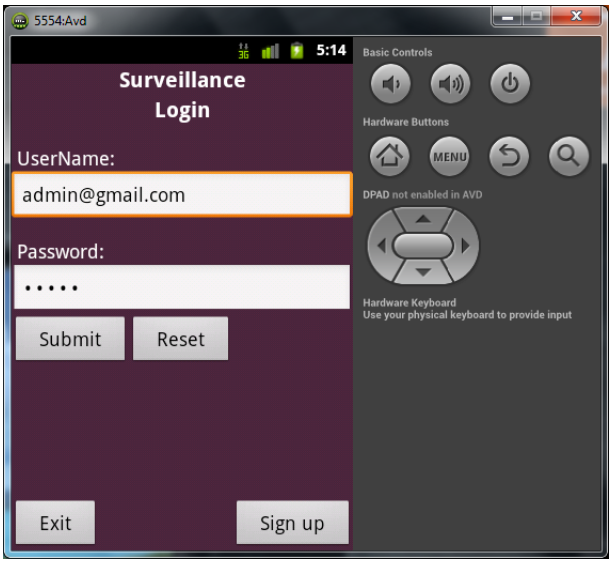


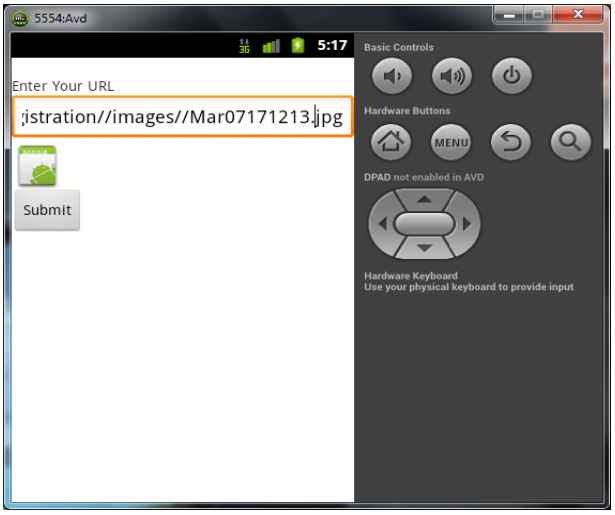

5554:Avd
March 7, 2013
5:13
Android
Clear
Notifications
Video Survilliance
Motion has been detected. Click to view 5:12 PM
Basic Controls
Hardware Buttons
MENU
DPAD not enabled in AVD
Hardware Keyboard
Use your physical keyboard to provide input

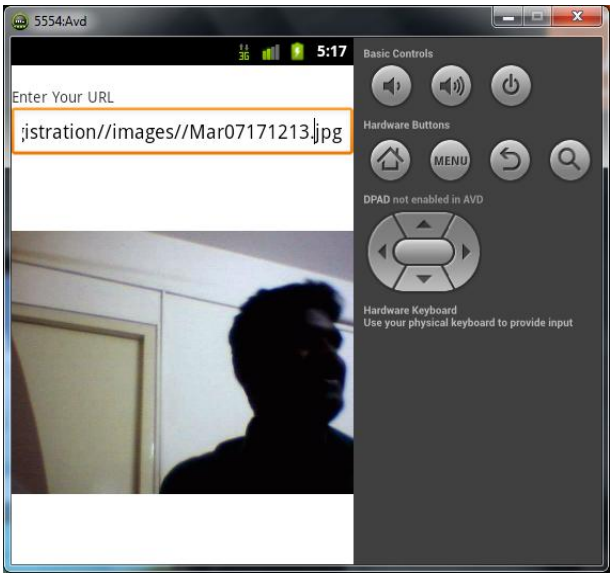

5554:Uniq
4:45 PM
Surveillance
Login
UserName:
Username
Password:
Password
Submit
Reset
Exit
Sign up
Basic Controls
Hardware Buttons
MENU
DPAD not enabled in AVD
Hardware Keyboard
Use your physical keyboard to provide input


5554:Avd
5:53
Back
Enter UserName:
Enter username
Enter Password:
Enter password
Re-Enter Password:
Re-enter password
Email Address:
Enter e-mail address
Phone number
Enter phone number
Submit
Basic Controls
Hardware Buttons
MENU
DPAD not enabled in AVD
Hardware Keyboard
Use your physical keyboard to provide input


5554:Avd
5:54
Back
Enter UserName:
admin
Enter Password:
•••••
Re-Enter Password:
•••••
Email Address:
admin@gmail.com
Phone number
999888566
Submit
Basic Controls
Hardware Buttons
MENU
DPAD not enabled in AVD
Hardware Keyboard
Use your physical keyboard to provide input


5554:Avd
5:14
Surveillance
Login
UserName:
admin@gmail.com
Password:
Submit
Reset
Exit
Sign up
Basic Controls
Hardware Buttons
MENU
DPAD not enabled in AVD
Hardware Keyboard
Use your physical keyboard to provide input


5554:Avd
5:17
Enter Your URL
;istration//images//Mar07171213.jpg
Submit
Basic Controls
Hardware Buttons
MENU
DPAD not enabled in AVD
Hardware Keyboard
Use your physical keyboard to provide input


5554:Avd
5:17
Enter Your URL
;istration//images//Mar07171213.jpg
Basic Controls
Hardware Buttons
MENU
DPAD not enabled in AVD
Hardware Keyboard
Use your physical keyboard to provide input

REFERENCES

REFERENCES

[1] I. Estévez-Ayres, P. Basanta-Val, M. García-Valls, J. A. Fisteus and L. Almeida, "QoS-aware Real-Time Composition Algorithms for Service-Based Applications", IEEE Trans. on Industrial Informatics, vol 5 (3), pp. 278-288, Aug. 2009.

[2] Android Operating System, http://www.android.com

[3] I. Estévez-Ayres, L. Almeida, M. García-Valls and P. Basanta-Val, "An Architecture to Support Dynamic Service Composition in Distributed Real-Time Systems", Proc of the 10th IEEE International Symposium on Object/component/service-oriented Real-time distributed Computing (ISORC), May 2007. Santorini Island, Greece.

[4] OMG, "Data Distribution Service for Real-time systems". Object Management Group, 1.2 formal/07-01-01 edition, January 2007.

[5] H. Schulzrinne, A. Rao and R. Lanphier, "Real Time Streaming Protocol (RTSP)", RFC 2326, http://www.ietf.org/rfc/rfc2326.t