SMARTPHONE FORENSICS TOOLKIT

By

NC Muqeet Kamaal

ASC Ivan William

PC Humza Aamir

Submitted to the Faculty of Computer Software Engineering
National University of Sciences and Technology, Islamabad in partial fulfillment
for the requirements of a B.E. Degree in Computer Software Engineering
JUNE 2014

CERTIFICATE

Certified that the contents and form of project report entitled "Software Forensics Toolkit" submitted by Muqeet Kamaal, Ivan William and Humza Aamir have been found satisfactory for the requirement of the degree.

Supervisor: _____

Lec Waseem Iqbal

ABSTRACT

The digital revolution and mobility of devices has caused an increase in the use of smartphones in criminal activities. Mobile forensics, aims to address this issue by providing investigation and legal authorities with software and hardware to gather evidence from smartphones and digital equipment gathered from the crime scene. Major artifacts and items of interest which include calls, text messages, logs and file structure are critical in providing information that can lead to culprits and assist in proceeding with investigations.

The Smartphone Forensics Toolkit aims to provide a forensic solution for smartphones operating on iOS and Android. The project is a Windows based that is multi-platform in terms of providing forensic tools and is a desktop application made using C#, provides GUI to the investigator with the help of Windows Forms.

The toolkit has been divided into two portions for handling iOS and Android phones respectively. A USB connection is established between the workstation and smartphone (evidence) before starting a case followed by phone attributes transfer including manufacturer name, model, OS version and IMEI.

Each case is followed by image extraction for Android/iOS in which artifacts are extracted and displayed given the state of the mobile evidence. An investigator initiates a new case, proceeds with image extraction, analyses contents by viewing them in database, xml, and text or pdf/word format and concludes case with a report.

DECLARATION

No portion of the work presented in this dissertation has been submitted in

support of another award or qualification either at this institution or elsewhere.

DEDICATION

In the name of Allah, the Most Gracious, the Most Beneficent,

Dedicated to our parents, our teachers and especially Military College of Signals.

ACKNOWLEDGEMENTS

All praise to Allah Almighty, whose guidance, provision of knowledge and resources has helped us to achieve a task of this magnitude. We are also highly grateful to our parents who have given us every kind of support and to our teachers who over the course of this degree have imparted valuable knowledge with utmost dedication.

TABLE OF CONTENTS

1. **Introduction** **2**

1.1 Purpose 2

1.2 Project Background 2

1.3 Project Scope 4

1.4 Objectives 4

1.5 Work Breakdown Structure 5

1.6 Deliverables 6

2. Literature Review **8**

2.1 Introduction 8

2.2 Shortcomings/issues 11

2.3 Proposed Project 12

2.4 Technological requirements 12

2.5 Software Requirements 13

2.6 Hardware Requirements 13

3. Software Requirements Specification **15**

3.1 Purpose 15

3.3 Intended Audience and Reading Suggestions 15

3.4 Product Scope 16

3.5 References 17

3.6 Overall Description 17

3.6.1 Product Perspective 17

3.6.2 Product Functions 18

3.6.3 User Classes and Characteristics 19

3.6.4 Operating Environment 19

3.6.5 Design and Implementation Constraints 19

3.6.6 User Documentation 20

3.6.7 Assumptions and Dependencies 20

3.7 External Interface Requirements 20

3.7.1 User Interfaces 20

3.7.2 Hardware Interfaces 21
3.7.3 Software Interfaces 22
3.7.4 Communication Interfaces 22
3.8 Other Nonfunctional Requirements 22
3.9.1 Performance Requirements 22
3.9.2 Safety Requirements 23
3.9.3 Security Requirements 23
3.9.4 Software Quality Attributes 23
3.9 Business Rules 24
4. Architectural Design 26
4.1 System Block Diagram 26
4.2 Software Components 26
4.3 Hardware Components 27
4.4 High Level Design Diagram (Modules Identification) 27
4.5 Interaction among Modules (Abstract View) 28
4.6 Architectural Style 28
4.7 Detailed Design 30
4.7.1 Database Diagram 30
4.8.1.2 Entity Relationship Diagram 31
4.9 UML Diagrams 32
4.10 Logical View 43
4.10.1 State Transition Diagrams 43
4.11 Dynamic View 44
4.12 Implementation View 46
5. System Implementation 49
5.1 Smartphone Forensics Process: An Overview 49
5.1.1 Forensics for iPhone (iOS) 49
5.1.2 Forensics for Android 50
5.1.3 View extracted information 51
6. Testing 53

6.1 Interface Testing 53
6.2 Functional Testing 58
APPENDIX A-1 USER MANUAL 63

List of Figures

Figures **Page Number**

1.1. Work Break Down Structure……3

1.2. iPhone Forensics……8

1.3. System Block Diagram……26

1.4. High Level Design.……27

1.5. Interaction Among modules……28

1.6. Model View Controller.……30

1.7. Database Diagram……30

1.8. Entity Relationship Diagram……31

1.9. Use Case Diagram……32

1.10. Device Connection……39

1.11. Image Acquisition……39

1.12. Image Analysis……40

1.13. Report Generation……41

1.14. Collaboration Diagram (Dev Connect)……42

1.15. Collaboration Diagram (Image Acquisition)……42

1.16. Collaboration Diagram (Report Generation)……43

1.17. State Transition (Dev Connect)……43

1.18. State Transition (Report Generation)……44

1.19. Activity Diagram……44

1.20. Dataflow Diagram……45

1.21. System Class Diagram……46

1.22. iOS Forensic Process……49

1.23. Tool bar and phone connect icon……50

1.24. Device Connection Manager……50

1.25. Viewer Tab……51

1.26. Test case #1 ……54

1.27. Test case #2……54

1.28. Test case #4……55

1.29. Test case #5……56

1.30. Test case #6……57

1.31. Test case #7……58

1.32. Test case #8……59

1.33. Test case #9……59

1.34. Test case #10……60

1.35. Test case #11……61

1.36. Test case #12……61

1.37. Test case #12……62

List of Tables

Table Number **Page Number**

1.1.Deliverables……..……..…….…….....6

1.2.Product Scope…….……..…….…….....16

1.3.System Classes Description……..…..……..…….……47

1.4. Test Case 1……..……..…….……..…..53

1.5.Test Case 2……..……..…….……54

1.6. Test Case 3……..……..…….……55

1.7.Test Case 4……..……..…….……55

1.8. Test Case 5……..……..…….…….....56

1.9.Test Case 6……..……..…….……....57

1.10. Test Case 7……..…….…….....57

1.11. Test Case 8……..…….…….58

1.12. Test Case 9…….…….....59

1.13. Test Case 10……...…….……59

1.14. Test Case 11……...…….…….....60

1.15. Test Case 12……...…….…….....61

1.16. Test Case 13……..……....62

CHAPTER 1

INTRODUCTION

1. Introduction

1.1 Purpose

The emergence of smart phones has been remarkable over the recent years. There has been a radical change in the smart phone industry. New and advanced smart phones are being developed each day with features that have become an integral part of our daily life. As the smart phones are becoming more advanced, so are the criminals and their method of conducting crimes. In the recent times, smart phones have proved quite useful in crimes involving terrorism, kidnapping for ransom, black mailing, money embezzlement, false impersonation, e-money laundering and many more.

The purpose of the system is to develop an open source multi-platform smart phone forensics toolkit that will perform forensic analysis on the smart phones recovered from crime scene and help the crime investigation agencies to access relevant information from the smart phones.

1.2 Project Background

Currently, there is no solution present in the market that can offer crime investigation teams to perform forensic analysis for different crime scenes. As the devices vary, the operating system and therefore, the platform also vary, thus limiting the activities which the teams can carry out. A multi-platform open source tool for performing digital analysis of devices involved in a crime scene will enable an all-in-one solution for smartphones operating on iOS, Android or Windows Phone.

There are many existing tools in the market for the extraction of the data. The significance of the project is highly emphasized as the only tool which caters for more than one smartphone operating system. The existing data recovery tools target a specific platform and operating system. These tools are premium and are highly expensive to purchase. Thus this project is designed as an open source tool for data extraction from multiple smartphones.

The tool is designed to extract evidence from the smartphones with supported OS which can serve as material for forensic investigation such as short messages, contacts and call logs. The material to be extracted is subject to the research which is going to be implemented. We tend to design a Windows based tool which integrates support for the three smartphone platforms.

Our tool is basically designed to provide digital forensic services on most widely used smartphone platforms. The tool will cater for forensic extraction of sensitive data from smartphones that can serve as viable evidence.

1.3 Project Scope

Our aim is to develop a forensically secure windows based open source data extraction tool which will provide digital forensic services for multiplatform smartphone's. The tool will be able but not limited to extracting sensitive information involving smartphone content such as messages, call logs, contacts and gallery image items. All of this will be done with keeping the phone data intact and ensuring its integrity.

The purpose of the project is to develop an open source multi-platform smart phone forensics toolkit that will perform forensic analysis on the smart phones recovered from crime scene and help the crime investigation agencies to create crime scene using digital evidence collected from these Smartphone's and send the criminal behind bars.

1.4 Objectives

Given below are the objectives of our project:

- To learn forensics and mobile forensics in particular for smartphone operating systems.
- To learn file structure, memory management and content extraction for iOS and Android.
- To utilize databases and xml parsing for readability of content.
- To develop a Windows application for forensically sound extraction of useful data and integration of available free to use tools that help in jail breaking (iOS) and rooting (Android).

1.5 Work Breakdown Structure

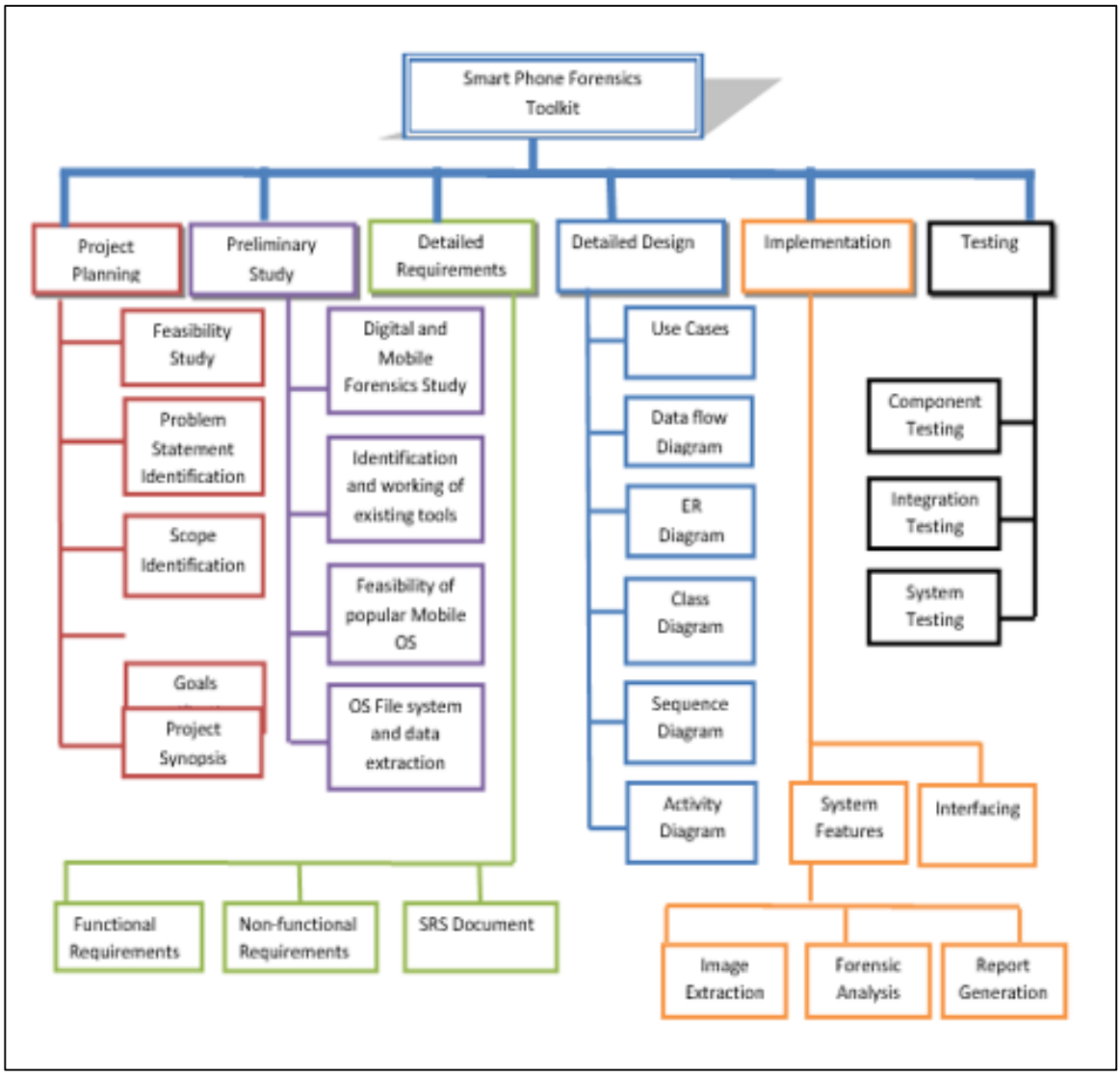

Figure 1: Work Breakdown Structure

1.6 Deliverables

Table 1: Deliverables

Deliverable Name	Deliverable Summary Description
Software Requirements Specification(SRS) Document	Complete Description of WHAT system will do, who will use it. Detailed description of functional and non-functional requirements and system features.
Analysis Document	Detailed requirement analysis and analysis models are included.
Design Document	Complete description of How the system will do. Design models are included.
Code	Complete code with the API.
Testing Document	Whole system is tested corresponding to the specifications. System is tested at all levels of Software Development Life Cycle (SDLC).
Complete System	Complete working system.

Table 1: Deliverables

CHAPTER 2

LITERATURE REVIEW

2. Literature Review

2.1 Introduction

2.1.1 iOS

iOS (the operating system for iPhone) is based on Mac OS X which is a variant of FreeBSD and is commonly known as a Unix-like operating system. The difference between Mac OS X and a common Unix operating system is the proprietary file system and kernel enhancements as well as the graphical user interface. Logically, iPhone has two partitions: one for storing the OS specific files such as kernel images and configuration files. The other partition is used for the storage of user specific settings and applications. From a forensic point of view, the second partition is more important because it contain all the functions the user performs on the iPhone and the data for those functions. For example, call history, Short Messaging Service (SMS) messages, contact list, emails, audio and video, and pictures taken through the built-in camera are all located on the second partition. Below is a snapshot of the iPhone root file system.

```
lrwxr-xr-x  1 root admin   23 Jul 12 19:17 Applications -> /var/stash/Applications
drwxrwxr-x  2 root admin   68 Jun  1  2010 Developer
drwxrwxr-x 14 root admin  680 Jul 12 19:18 Library
drwxr-xr-x  3 root wheel  102 Jul 30  2010 System
lrwxr-xr-x  1 root admin   11 Aug  9 02:13 User -> /var/mobile
drwxr-xr-x  2 root wheel 1972 Jul 26 19:29 bin
drwxr-xr-x  2 root wheel   68 Oct 28  2006 boot
drwxrwxr-t  2 root admin   68 May 20  2010 cores
dr-xr-xr-x  3 root wheel 1232 Aug  9 02:11 dev
lrwxr-xr-x  1 root wheel   12 Jul 12 19:18 etc -> private/etc/
drwxr-xr-x  2 root wheel   68 Oct 28  2006 lib
drwxr-xr-x  2 root wheel   68 Oct 28  2006 mnt
drwxr-xr-x  4 root wheel  136 Jul 30  2010 private
drwxr-xr-x  2 root wheel 1326 Jul 12 19:18 sbin
lrwxr-xr-x  1 root wheel   16 Jul 12 19:18 tmp -> private/var/tmp/
drwxr-xr-x  6 root wheel  306 Jul 12 19:19 usr
lrwxr-xr-x  1 root wheel   12 Jul 12 19:19 var -> private/var/
iPhone:/ root#
```

Figure 2: iPhone Forensics

Forensic investigation of an iPhone can be quite tedious because the underlying iOS of the iPhone store a large number of files and the relevant information necessary for the investigation in multiple places and files. Files are also stored in a variety of formats (both open and proprietary) therefore, without knowing the location and purpose of each file, forensic investigation is bound to fail.

Implementation

The forensic investigation toolkit is developed in C#.NET to provide cross platform compatibility. Due to restricted environment, the device is jailbreaked so that root access can be obtained. It is preferred to jailbreak the device manually.

Once the device is jailbreaked and root access is obtained, an OpenSSH server is installed on the device. A secure shell client implemented within the forensic investigation toolkit is then used to connect to the OpenSSH server and pull data from the iPhone device to the forensic investigation machine.

Analysis

The following categories of files were extracted and found helpful in forensic analysis.

General Log: This is the first file that should be analyzed in order to find out what version of the iOS is used on the device.

Deb files: These are traditional package files used in debian based Linux OS. Since iOS is based on FreeBSD Unix having the underlying debian package management system, all package files are in deb format. While examining these files can give the examiner the idea, which packages were downloaded by the user and used on the iPhone

Status text file: This file contains the status of each package ever installed on the iPhone such as applications downloaded and installed from Cydia repositories.

Local time settings: Contains information regarding the time settings on the device and can be useful in deducting the actual location of the accused or the victim possessing the iPhone.

/private/var/stash/Applications: This directory has sub directories representing each of the application available on the default SpringBoard desktop of the iPhone and applications installed through the Cydia application repository.

AddressBook.sqlitedb: Contains all the contact information and numbers stored within the iPhone.

Calendar.sqlitedb: Contains calendar entries such as any important events that were set on particular days.

call_history.db: Contains information of Calls dialed, received, and missed.

sms.db: contains SMSs sent and received through the mobile.

Email: Emails are stored in the "Protected Index" file

History.plist: Contains the browser history such as the website visited through the iPhone.

Voicemail.db: Contains voice mails send and received.

2.1.2 Android

Df A Course on "Android Forensics and Security Testing" by Shawn Valle (shawnvalle@gmail.com). pdf

Approved for Public Release: 12-3411 Distribution Unlimited

Abstract:

The document begins with introduction of digital forensics and its expanding needs in today's world. In the era of cellular technology, the mobile phone forensics has become a

dire need. The document expresses concerns on the complexity of mobile phone forensics with special focus on android forensics. Enlisting the history and overview of the android smartphone platform, the paper includes android structure, memory hierarchy and boot loader and RAM structure concerning android.

It then goes on explaining android security models and hardware interfaces. It also explains android forensic techniques and concerned software using ADB (Android Debugging Bridge). This is followed by a detailed illustration of Android File System and Data and how to fetch important data from subdirectories. A handful of SQLite commands is also enlisted that help fetching data from android phone.

The document also teaches forensically sound methods of handling and acquiring devices and how to prevent changes within them. Moreover, it also explains some techniques of circumventing passcodes and gaining root privileges.

Android Forensics by Simson L. Garfinkel, Ph.D

Associate Professor, Naval Graduate School Montery, CA

Reference: http://www.simson.net/

This document explains file carving from android using tools. In addition to it, it also focuses on Fragment Recovery Carving. The document lists various and multiple methods for acquiring forensic data from android smartphone version 2.2 (Froyo). This can be extended to Jellybean as well.

2.2 Shortcomings/issues

Smartphone forensics is an ever evolving subject in the field of computer security. New exploits, patches and workaround are discovered and implemented quickly. The field of forensics requires access to operating system internals and user data to extract

information without leaving any trace as to preserve evidence integrity. Following are the shortcoming/issues in our project:

- An iPhone or Android smartphone protected by keypad lock cannot be bypassed. Thus an evidence needs to be open for access.
- Content that the mobile owner has encrypted using a mobile application cannot be decrypted, it has to be extracted first and decrypted using another tool.
- In case of iPhone, it needs to be jailbreaked (preserves user data) and an android phones needs to be rooted for content access.

2.3 Proposed Project

The proposed project finds the solution to integrate tools and extract information from smartphone evidence involved in a crime scene:

- Connect smartphone via USB.
- Initiate new case as a legal investigator.
- Following assumptions have been made
  - Usb module of workstation and phone is in working state.
  - Phone is not password protected.
  - An iPhone allows jailbreak and Android allows rooting of OS.
- Optional feature(s) include:
  - Report generation in pdf format/word format.

2.4 Technological requirements

Our project requires following software and hardware requirements.

2.5 Software Requirements

The software(s) required for the implementation of our project includes:

- Microsoft .Net Framework
- Microsoft Visual Studio 2012
- Red Sn0w, iFile, terminal (Linux)

Hardware Requirements

The Hardware required for the implementation of our project includes:

- Workstation having Windows 7 or later installed.
- Data Cable for hardware and software interfacing.
- Smartphone for forensic data extraction.

CHAPTER 3

SYSTEM REQUIREMENT SPECIFICATION

3. Introduction

3.1 Purpose

The emergence of smart phones has been remarkable over the recent years. There has been a radical change in the smart phone industry. New and advanced smart phones are being developed each day with features that have become an integral part of our daily life. As the smart phones are becoming more advanced, so are the criminals and their method of conducting crimes. In the recent times, smart phones have proved quite useful in crimes involving terrorism, kidnapping for ransom, black mailing, money embezzlement, false impersonation and many more.

3.2 Intended Audience and Reading Suggestions

Audience for this SRS template is:

- Supervisors
- Forensic Investigators
- Crime Scene Investigators - Private/Agencies
- Faculty Members of respected evaluation panel

Above mentioned audience is required to have understanding of:

- Digital Forensics
- Image Acquisition
- Chain of custody
- Smartphone and Operating System Information

3.3 Product Scope

Our aim is to develop a forensically secure windows based open source data extraction tool which will provide digital forensic services for multiplatform smartphone's. The tool will be able but not limited to extracting sensitive information involving smartphone content such as messages, call logs, contacts and gallery image items. All of this will be done with keeping the phone data intact and ensuring its integrity.

The purpose of the project is to develop an open source multi-platform smart phone forensics toolkit that will perform forensic analysis on the smart phones recovered from crime scene and help the crime investigation agencies to create crime scene using digital evidence collected from these Smartphone's and send the criminal behind bars.

For	Cyber Criminal Investigation Agencies and Digital Forensic Investigators.
What	Multiplatform Smartphone Forensics Application Toolkit.
The	SMARTPHONE FORENSIC TOOLKIT (SFT)
Is	A Windows based desktop application.
That	Provides complete customizable smartphone forensic tools and services.
Unlike	Existing single platform proprietary expensive tools.
Our Product	Ensures efficient, forensic data extraction and evaluation from digital forensics point of view with prime focus on data integrity.

Table 2: Product Scope

3.4 References

- Project Synopsis- An approved document with complete information containing extended title, brief description of the project, scope of work, academic objectives, applications, previous work, material resources required, no of students and special skill required has already been submitted to the department.

3.6 Overall Description

3.6.1 Product Perspective

Currently, there is no solution present in the market that can offer crime investigation teams to perform forensic analysis for different crime scenes. As the devices vary, the operating system and therefore, the platform also vary, thus limiting the activities which the teams can carry out. A multi-platform open source tool for performing digital analysis of devices involved in a crime scene will enable an all-in-one solution for smartphones operating on iOS or Android.

There are many existing tools in the market for the extraction of the data. The significance of the project is highly emphasized as the only tool which caters for more than one smartphone operating system. The existing data recovery tools target a specific platform and operating system. These tools are premium and are highly expensive to purchase. Thus this project is designed as an open source tool for data extraction from multiple smartphones. The tool is designed to extract evidence from the smartphones with supported OS which can serve as material for forensic investigation such as short messages, contacts and call logs. The material to be extracted is subject to the research which is going to be implemented. We tend to design a Windows based tool which integrates support for the three smartphone platforms.

Our tool is basically designed to provide digital forensic services on most widely used smartphone platforms. The tool will cater for forensic extraction of sensitive data from smartphones that can serve as viable evidence. Following are the specified roles for tool actors:

Forensic Investigation Agencies: Under attorney-client privilege, certain agencies obtaining the smartphone evidence with the sole purpose of carrying out forensic investigations.

Lead Case Investigator: Under some circumstances where individual case investigators are hired under attorney-client privilege, the lead case/ forensic investigator may analyze the phone data.

Law Firms/ Associates: Concerned law firms and associates with intention of analyzing evidence and preparing case and arguments based on that.

Prosecutors/ Law Defendants: In case of criminal investigations, government attorneys or defendants may use the tool as well.

3.6.2 Product Functions

- A Microsoft Windows based GUI (.NET Framework 3.5 supported)
- Smartphone Connection and OS detection
- Root Permission check and access
- Sparse Phone Image Acquisition
- Parsing and analyze image
- Extracting messages, bit stream acquisition
- Analyzing Call logs and Contacts

3.6.3 User Classes and Characteristics

Smartphone Forensic Toolkit (SFT) is designed to provide forensic investigators a handy tool to perform smartphone forensics easily and professionally. It allows them to carry out forensic extraction of smartphone data. Generically user classes belong to same category i.e. law enforcement agencies. It comprises of forensic investigation agencies, lead forensic investigators, law firms and associates and government attorneys, prosecutors and defendants.

3.6.4 Operating Environment

Smartphone Forensic Toolkit (SFT) aims to provide an intuitive and appeasing windows based GUI supported on recent Windows Platform. The tool will be designed using Microsoft .NET Framework 4.5 with enhanced compatibility and support of .NET framework 3.5 and later. The tool will interact with the phone using USB port and data cable.

3.6.5 Design and Implementation Constraints

The application will be developed for mobile devices with limited memory and processing power. For an effective text recognition system, extensive training on large data sets is required naturally requiring large memory and high processing power. However, since the training phase is offline, it can be carried out on a desktop machine and the learned parameters can be used on the mobile device for recognition of the input textual content.

3.6.6 User Documentation

The user documentation consists of a user manual.

3.6.7 Assumptions and Dependencies

- The investigator should have the permission granted from the court or attorney dealing with the case to look directly into the evidence under attorney client privilege.
- Prior using the tool, the user should acknowledge that he is solely responsible for ensuring the integrity of "Chain of Custody".
- Though the data is extracted forensically, yet the user should agree that any harm done to the evidence or its integrity as such it poses a threat to the validity of "Chain of Custody" is outside the working of the tool and the user/ investigator is responsible for it.
- OS running on the smartphone must match version with OS supported by tool kit.
- The smartphone OS to be implemented must already have some existing research in the field of forensics.
- Smartphone should provide access to data to be extracted and if it does not, then sufficient root privileges should be provided.
- Critical components of the phone such as storage and USB port should be operational.
- The phone should be switched on and responsive.
- The phone should be unlocked and storage or any data should not be protected by any sort of encryption.

3.7 External Interface Requirements

3.7.1 User Interfaces

The user will be displayed a Windows based graphical user interface with the several options available on the top menu bar and major function on the left side of the display. Clicking on each option on the left side reveals a work area on the center-right corresponding to the task being selected.

- Welcome Screen.
- Help in menu bar of main screen.
- About in menu bar of main screen.
- Phone connection
- Case Initiation/Case Linking
- A workspace in Main screen with following options in left pane menu:

  - Device information

  - Call Logs

  - Phone book

  - Messages
- The selected option will show the output in right work area.
- iOS Forensics
- Android Forensics
- Report

3.7.2 Hardware Interfaces

The main connection between the application and the smartphone evidence is through a USB cable. This is a wired connection, one end consisting of the smartphone with its attributes, files and artifacts and the other end, where the program resides and will extract information from the device.

The toolkit will run on any client PC having optimal hardware. However it is recommended that the machine should have a good and updated processor and efficient processing speed for image extraction.

Following are the minimum hardware requirements for the toolkit.

- Intel Core 2 Duo or Greater processor
- 4 GB Ram (recommended)
- USB Port 2.0
- Smartphone as evidence

3.7.3 Software Interfaces

- Microsoft Windows 7 or later
- Dot NET 3.5 or later
- Compatible smart phone drivers
- Supported Smartphone Operating Systems:
- iOS 4 - 6.1.3
- Android Jelly Bean 2.2 – 4.1

3.7.4 Communication Interfaces

- USB port 2.0
- Compatible USB drivers
- Smart phone USB connector

3.8 Other Non-Functional Requirements

3.8.1 Performance Requirements

3.8.1.1 Throughput:

The throughput is directly dependent upon the type of phone model being connected, its memory content, size of storage and the image that will be extracted. The throughput can vary for each type of smartphone connected, due to its type of operating system and working environment.

The image extraction is a resource intensive task that also relies heavily on the hardware provided on the investigator's workstation. A better CPU, large RAM and cache will ensure better throughput of the toolkit.

3.8.2 Safety Requirements

Similar to forensic tools available, such as FTK Imager or Prodiscover, the phone image under evidence will be saved on the workstation's hard disk.

3.8.3 Security Requirements:

The system requires a secure environment to work. Any external interface except the smartphone connection module needs to be removed to secure the image and other content of the tool kit.

3.8.4 Software Quality Attributes:

3.8.5 Availability:

The availability of the toolkit shall be dependent of the system being installed and will be available for use throughout a criminal investigation.

3.8.6 Mean Time between Failures (MTBF):

The MTBF for our tool kit is such that the system will not fail during forensic analysis and image extraction.

3.8.7 Mean Time to Repair (MTTR):

In case of any system crash or tool kit malfunction, the MTTR will be the same as the reboot time of the workstation and till time the routine software services are restored.

3.8.8 Usability:

3.8.8.1 Learnability:

The program has been designed to be simple and intuitive. A forensic investigator new to the system can be acquainted with the basic features and functionalities in less than a day.

3.9 Business Rules

Smartphone should be operational and connection ports should be working.

- Operating systems need to be either jailbreaked in case of iPhone or rooted in case of Android
- The phone should not have a security code for unlocking.
- Memory/Storage of the phone should not be formatted prior to image extraction.
- The Smartphone Forensic Toolkit is viable to unauthorized access and cannot guarantee safe usage of its features.
- Ensuring physical and logical security of the workstation is beyond the scope of this project.
- Support for latest smartphone operating system is subject to availability of research and practical development in field of forensics.

CHAPTER 4

SYSTEM DESIGN SPECIFICATION

4. Architectural Design

4.1 System Block Diagram

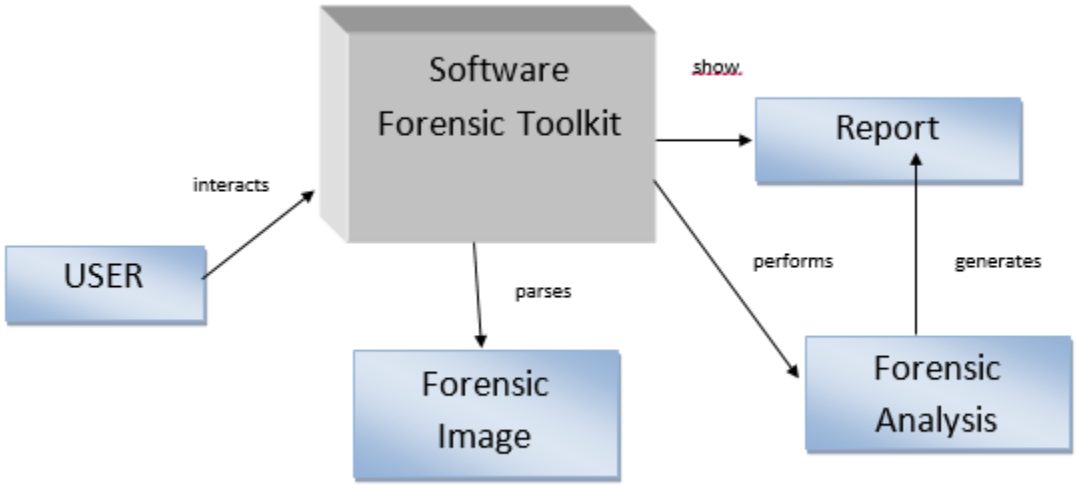

Figure 3: System Block Diagram

4.2 Software Components

Our project contains following software components:

- **Operating Systems**
    1. Microsoft Windows 7/8/8.1
    2. Android 2.3 (Gingerbread) – 4.4 (Kit Kat)
    3. iOS 4.3 – 7.0
- **Software Packages**
    1. C# .NET 4.5
    2. Microsoft Visual Studio 2013
    3. Oxygen Forensics/MobilEdit Forensic 7.5 (Image Acquisition Tool)

4.3 Hardware Components

1. Personal Computer(s) with usb port and cable
2. iPhone having iOS 4.3 or greater /Android Smartphone with OS version 2.3 or greater

4.4 High Level Design Diagram

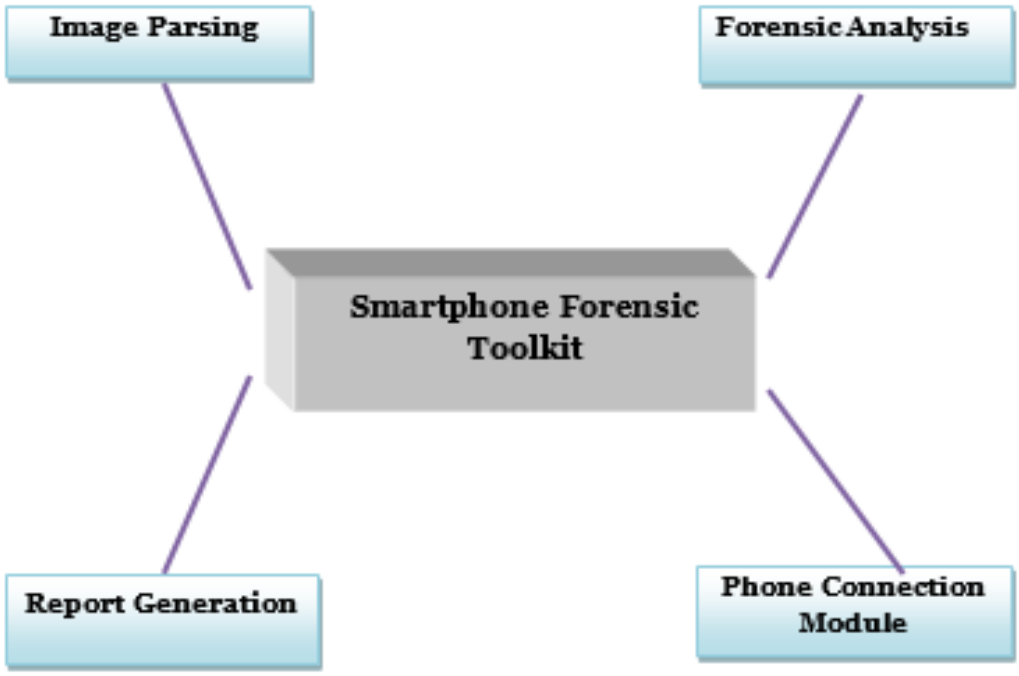

Figure 4: High Level Design

4.5 Interaction among modules (Abstract View)

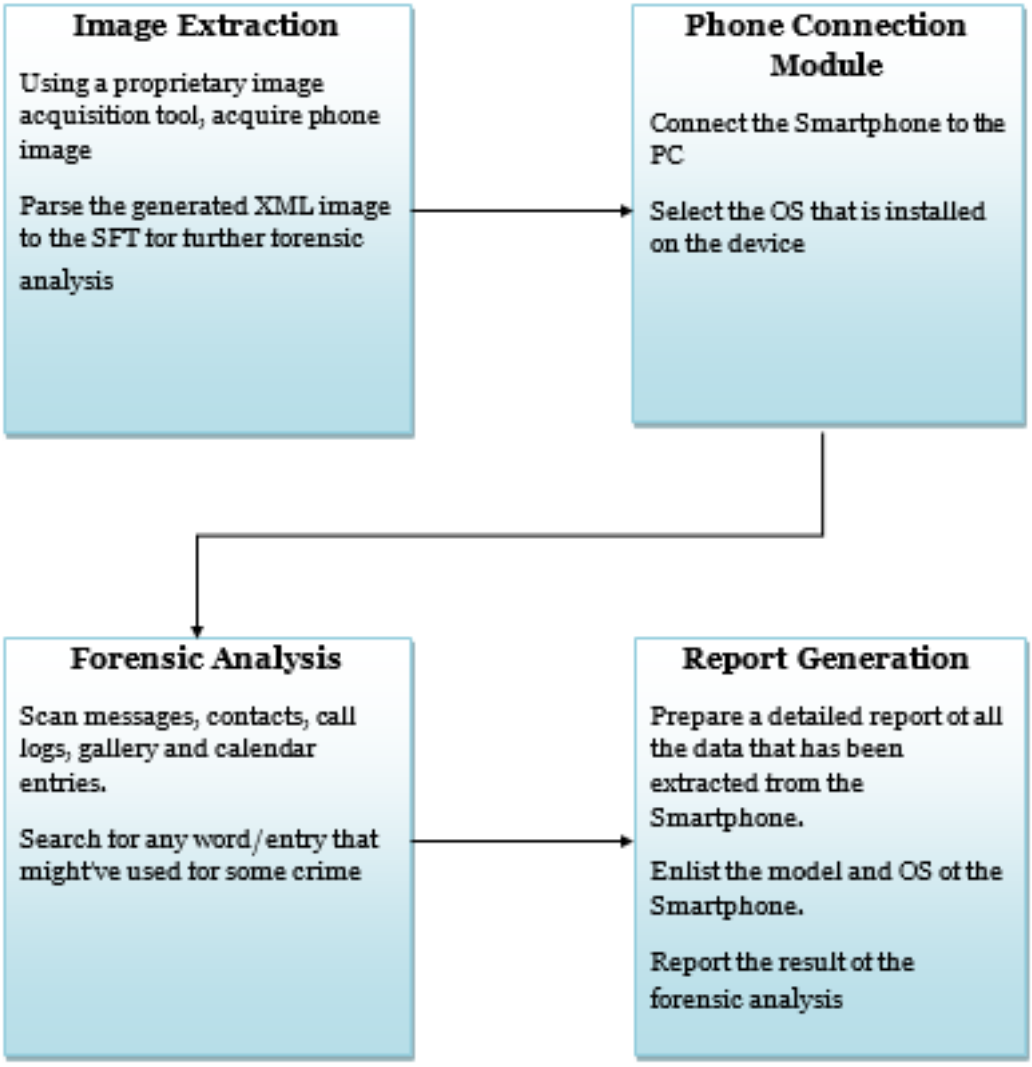

Figure 5: Interaction among modules (An abstract view)

4.6 Architectural Style and Design Pattern

Software architecture of the Smartphone Forensic Toolkit (SFT) will be modeled using **Model-View-Controller (MVC)**. The toolkit will have an easy to use interface that will enable the user to perform tasks with much ease and less difficulty. The interface of the

toolkit will be distinct from the application logic. The interface shall form the **View** of the architecture.

The Application logic of the toolkit handles all the functions of the image extraction and forensic analysis of the device, triggered by the GUI. This will form the **Model** of the toolkit.

Controller controls the coordination between the view and the model. There is no direct communication between model and the view all the communication is directed using the controller. The system is divided into modules as per the main functionality of the system.

View in the toolkit is the main GUI of the toolkit, commonly known as the interface. The view shall display the extracted data from the smartphone and give options to the user to further explore the data.

Model in this system is the Evidence class that maintains all evidence obtained from the smartphone. The class will be triggered by an object that the controller will invoke whenever a specific smartphone is connected. The Evidence class will be responsible for importing and holding all of the data obtained from the phone and sending the evidence data to the **View**. The class in the Model shall be responsible to send the extracted data to the view for user. The user can then further explore the data by selecting a folder. Again, it would be the responsibility of the model classes to display the required data.

Controller in the toolkit will be System.UI class which handles all the UI events in .NET applications since system functionality is clearly divided in modules. These Handlers only invoke the functions of their respect Modules. Controller only invokes the functions but do no processing.

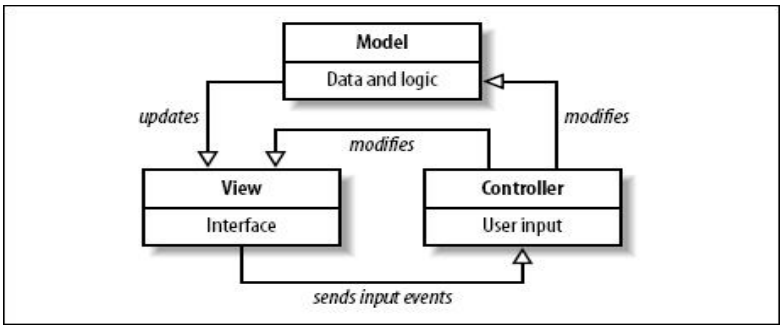

Figure 6: Model-View-Controller (Generic Interaction)

4.7 Detailed Design

4.7.1 Database Diagram

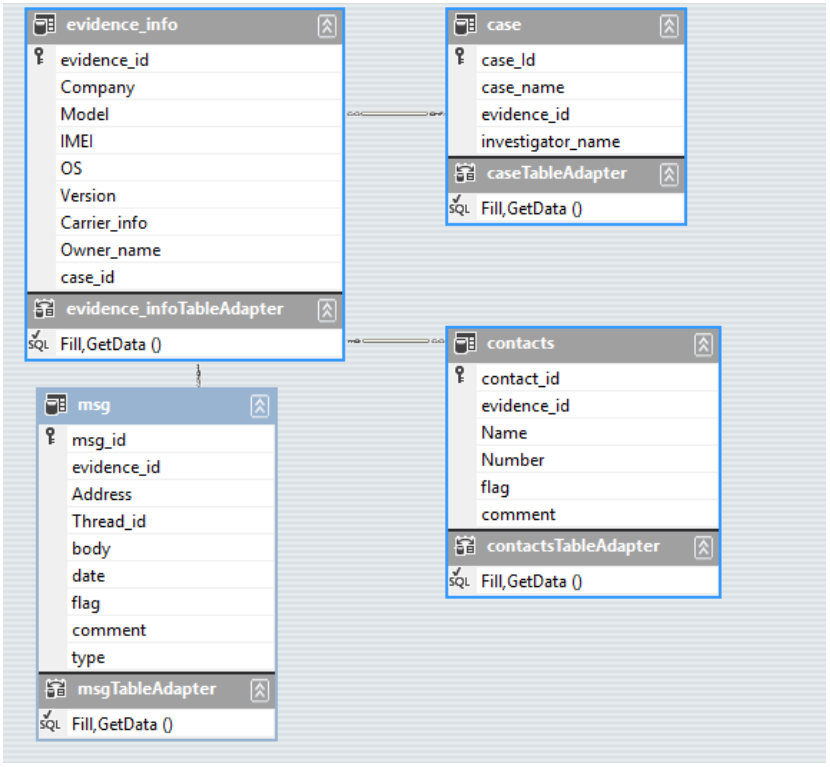

Figure 7: Database Diagram

4.7.1.1 Entity Relationship Diagram

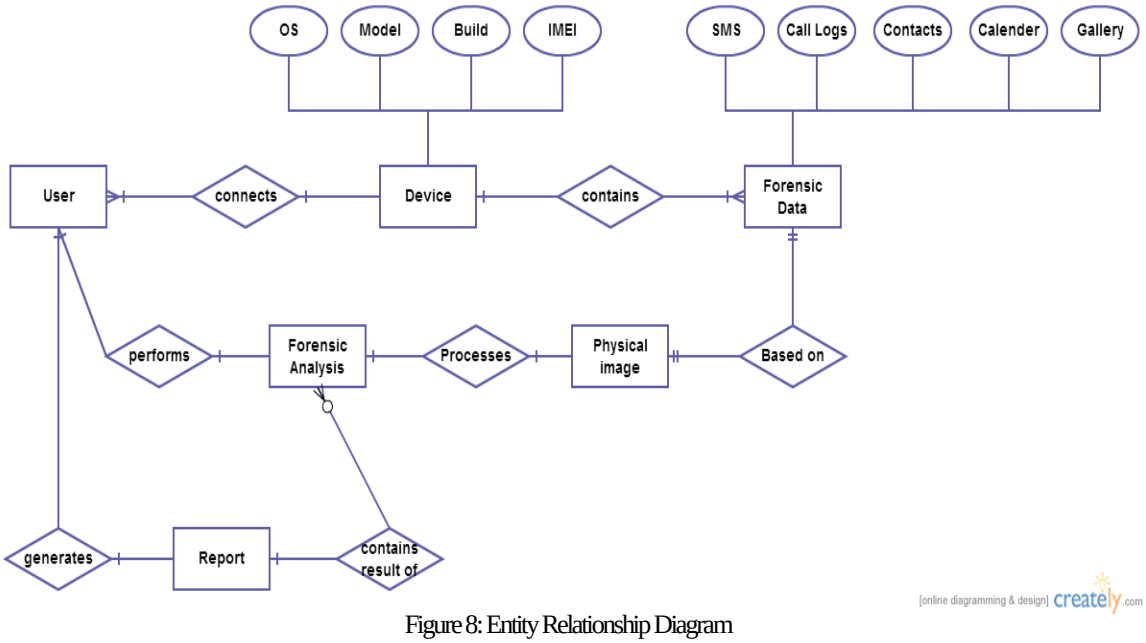

Figure 8: Entity Relationship Diagram

4.8 Entities & Attributes

- **User**
  - Forensic Investigator
- **Smartphone**
  - Operating System (Must be iOS or Android Based)
  - Should have normal boot and a working USB interface.
  - Should not be password protected.
  - For Android, ADB (Android Debugging Bridge) should be enabled.
- **Forensic Image**
  - Contains user data such as contacts, messages and photos.
- **Smartphone detection tool**

- **Image Acquisition Tool**

4.9 UML Diagrams

4.9.1 Use case Diagram

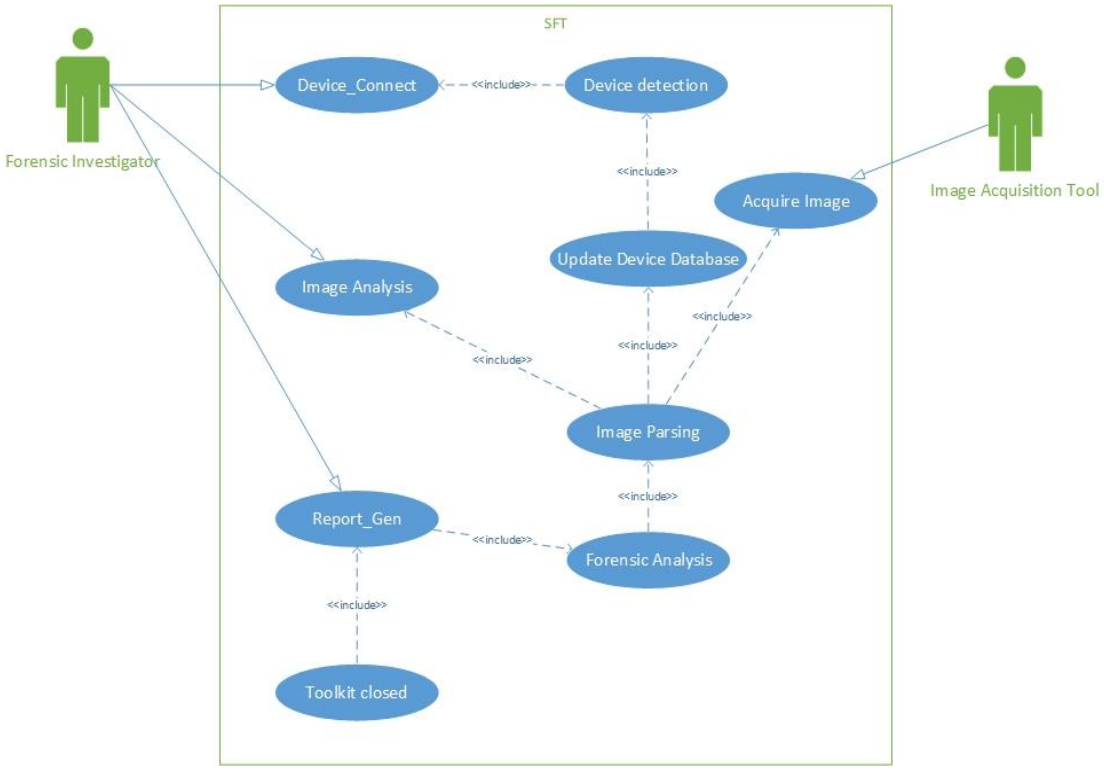

Figure 9: Use case diagram

4.9.1.1 Use case Specifications

Actors:

Forensic Investigator

Use Cases:

Device_Connection

Image_Acquisition

Image_Analysis

Report_Gen

a) Device_Connect

1. Brief Description

This use case describes occurrence of events when smartphone (evidence) is connected to the workstation via USB.

2. Actors

Forensic Investigator

3. Pre-conditions

- Mobile must have normal boot, ADB enabled (in case of Android).
- USB port must be functional.

4. Basic Flow of Events

This use case starts when the investigator connects the device to the workstation.

1. Phone is connected as a media device to the workstation.
2. The auto detect function of the toolkit recognizes the type of phone connected.

5. Alternative Flows

5.1 Missing Functionality

- If in step 1 of the basic flow i.e. the smartphone is password/pattern protected, the information transfer will fail and case will end with a failure condition.

5.2 Wrong Functionality

- If the device drivers are not successfully installed, the device will not connect and case will end with a failure.

6. Key Scenarios

N/A

7. Post-conditions

7.1 Successful Condition

If use case was successful, device will be connected successfully.

7.2 Failure Condition

If not, device will not be recognized and thus become useless.

8. Special Requirements

- Smartphone should not be password protected.
- Requisite device drivers for the smartphone must be present within the workstation.

b) Image_Acquisition

1. Brief Description

This use case describes what happens when forensic evidence is to be collected from the smartphone.

2. Actors

Forensic Investigator

3. Pre-conditions

- Smartphone must have normal boot and not be password protected.
- Smartphone must be connected to the workstation.
- Image Acquisition tool must be installed on the workstation.

4. Basic Flow of Events

This use case starts when the forensic investigator begins the image acquisition of the smartphone after its OS, make and model parameters have been identified:

1. The imaging tool is initialized.
2. The device is identified and image is acquired.

3. Evidence contents, including user data is displayed such as contacts and text messages are included in the image.
4. Forensic investigator is closes the tool and initializes SFT for image parsing.

5. Alternative Flows

5.1 Missing Functionality

- If the phone make/model/OS version is not supported by the toolkit, the image acquisition cannot be performed and case ends with failure condition.

5.2 Wrong Functionality

N/A

6. Key Scenarios

N/A

7. Post-conditions

a. Successful Condition

If use case was successful, image acquisition will be performed.

b. Failure Condition

If not, image acquisition will not be performed.

8. Special Requirements

- Smartphone image to be acquired must be of reasonable size

c) Image_Analysis

1. Brief Description

This use case describes what happens when forensic investigator opens up a smartphone forensic image for analysis during a crime scene.

2. Actors

Forensic Investigator

3. Pre-conditions

- Image must be acquired from device.

4. Basic Flow of Events

This use case starts when the forensic investigator has acquired an image and proceeds to analyze the contents in it.

1. Investigator opens up the toolkit.
2. The device detection and image acquisition is performed.
3. The image shows the file structure of the phone.
4. User contents such as contacts, text messages are displayed in a tabular form.

5. Alternative Flows

5.1 Missing Functionality

- If in step 2 of the basic flow i.e. device detection is not successful, then the case ends with a failure condition.
- If in step 2 again, if the acquisition is not successful, the case ends with a failure condition.

5.2 Wrong Functionality

N/A

6. Key Scenarios

N/A

7. Post-conditions

a. Successful Condition

If the use case was successful, extracted image will be available for displaying user contents and forensically important data.

b. Failure Condition

If not, the image will not be created and no data is displayed.

8. Special Requirements

- Integrity of the image needs to be maintained.

d) Report_Gen

5. Brief Description

This use case describes what happens when forensic investigator has completed forensic analysis and generates report to proceed with the forensic process.

6. Actors

Forensic Investigator

7. Pre-conditions

- Image must be acquired from device.

8. Basic Flow of Events

This use case starts when the forensic investigator has performed analysis of the image and wants to generate report for future reference.

1. Investigator selects the image.
2. The option for report generation is selected.
3. The toolkit generates report.
4. Important parameters such as time of image taken, phone model/make and investigator name are added to the report.

5. Investigator is provided with the option to print the report or keep it on their workstation.

9. Alternative Flows

5.1 Missing Functionality

- If in step 1 of the basic flow i.e. if the image is corrupted or not present, the case ends with a failure.

5.2 Wrong Functionality

N/A

10. Key Scenarios

N/A

11. Post-conditions

a. Successful Condition

If the use case was successful, report is generated.

b. Failure Condition

If not, the report is not generated.

12. Special Requirements

- Integrity of the image needs to be maintained.

4.9.2 Sequence Diagram of key use cases

a) Use case: Device_Connect

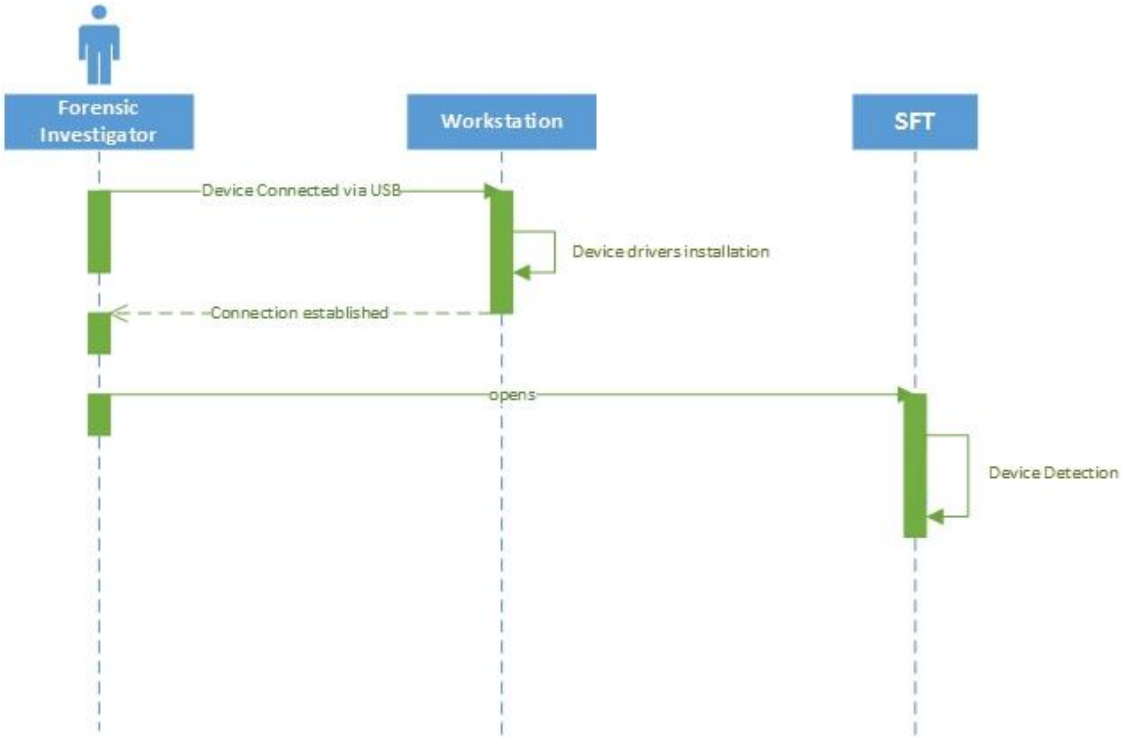

Figure 10: Sequence Diagram of device_connect

b) Use case: Image_Acquisition

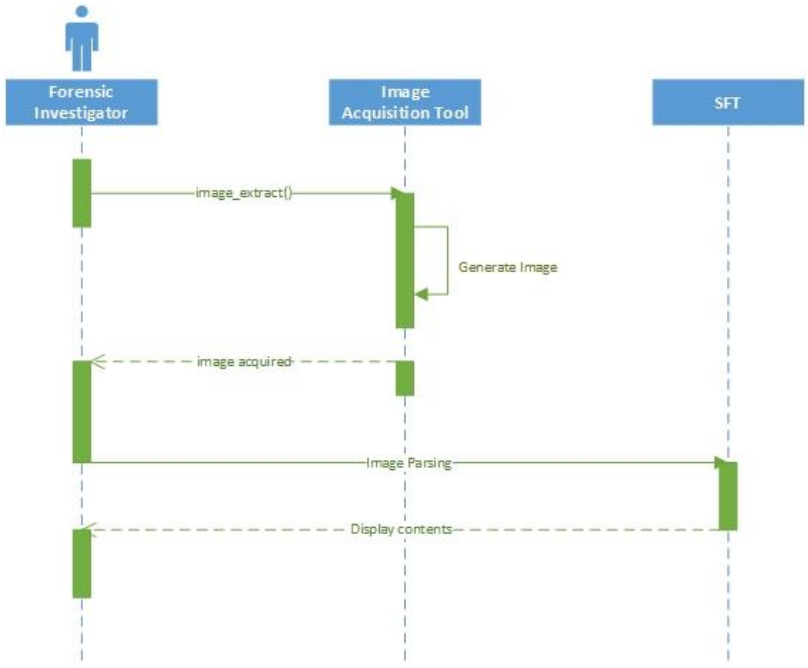

Figure 11: Sequence Diagram of Image_Acquisition

c) Use case: Image_Analysis

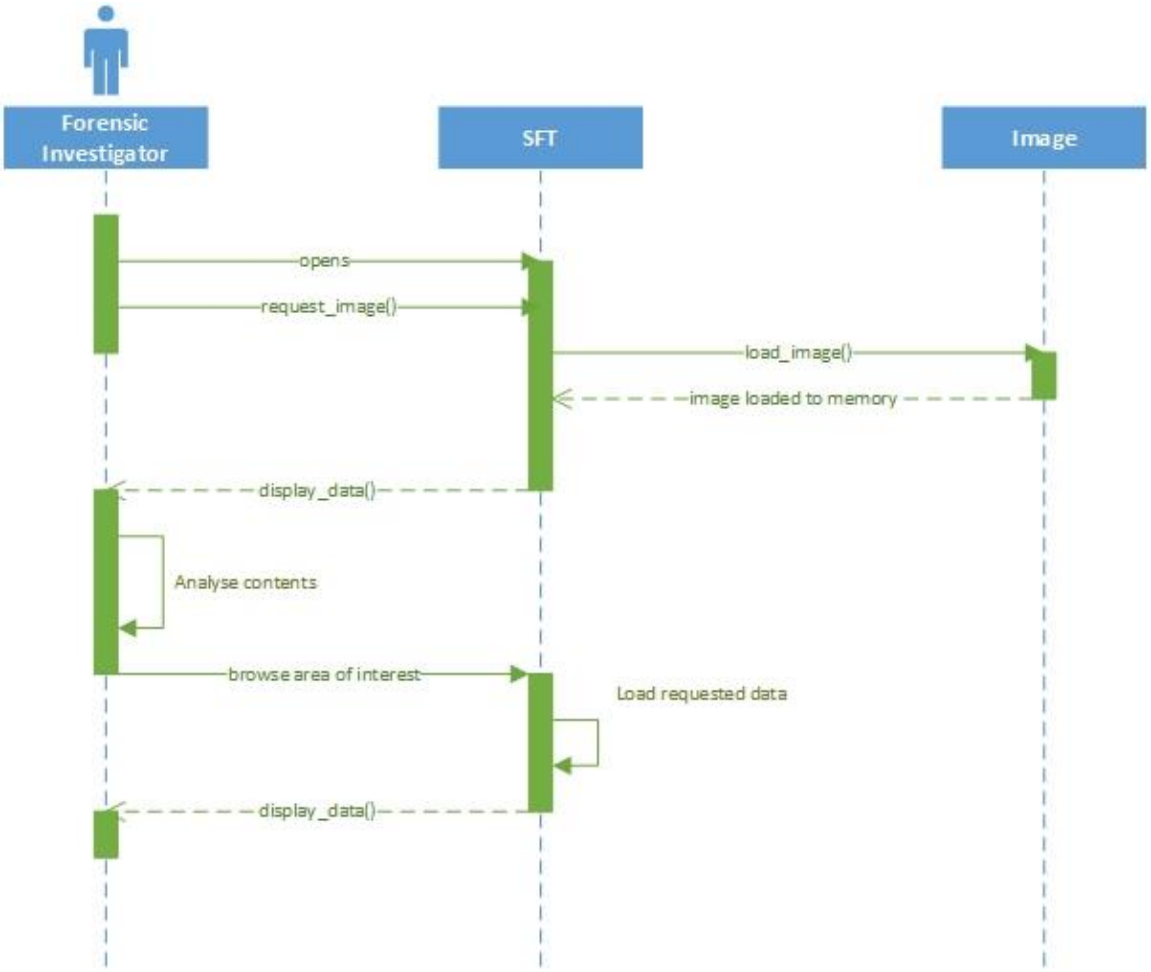

Figure 12: Sequence Diagram of Image_Analysis

d) Use case: Report_Gen

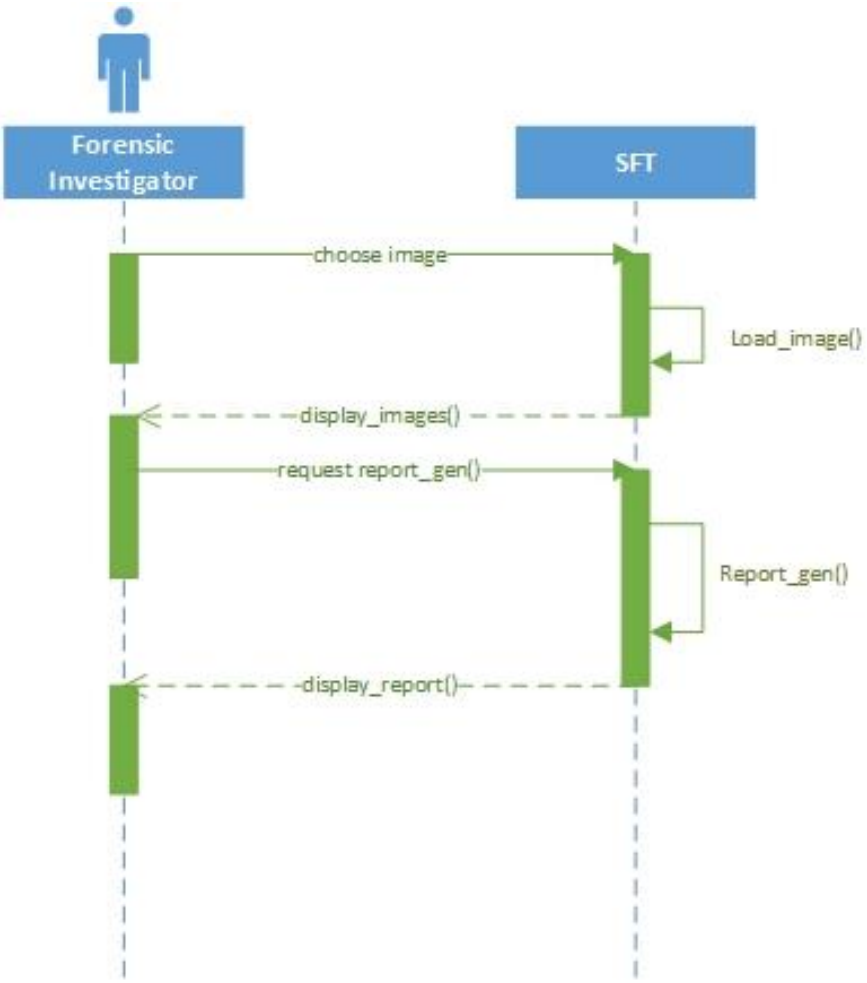

Figure 13: Sequence Diagram of Report_Gen

3.2.1.3 Collaboration Diagrams of key use cases

a) Use case: Device_Connect

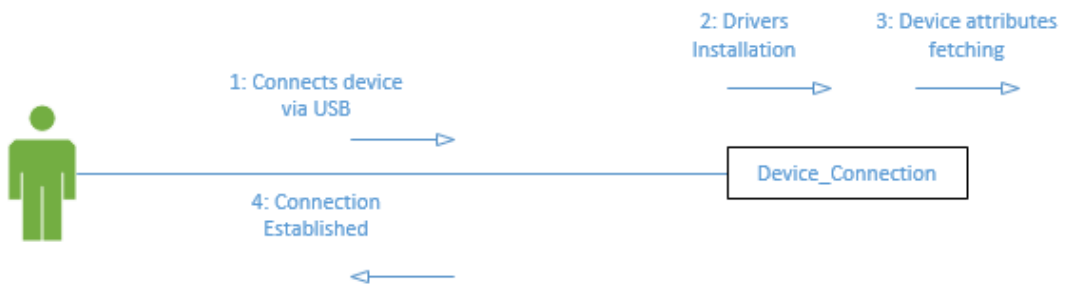

Figure 14: Collaboration Diagram of Device_Connect

b) Use case: Image_Acquisition

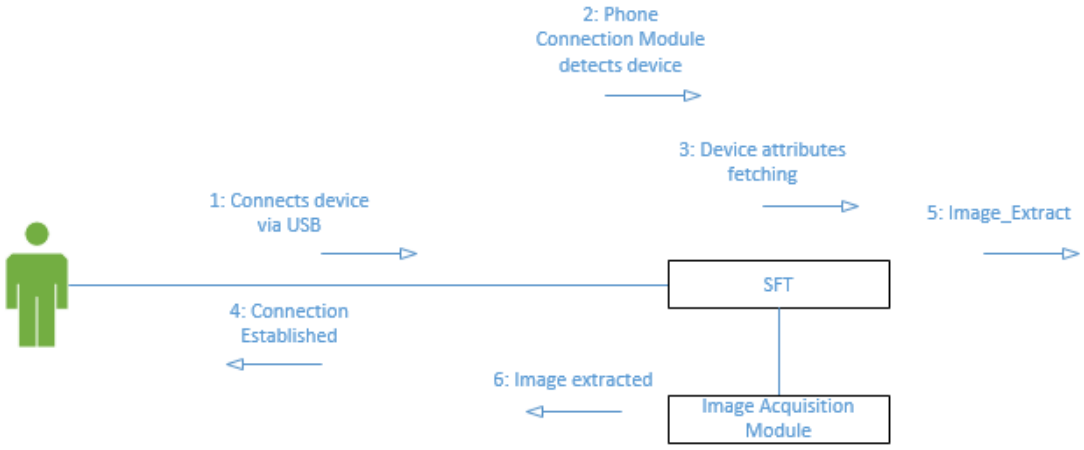

Figure 15: Collaboration Diagram of Image_Acquisition

c) Usecase: Report_Gen

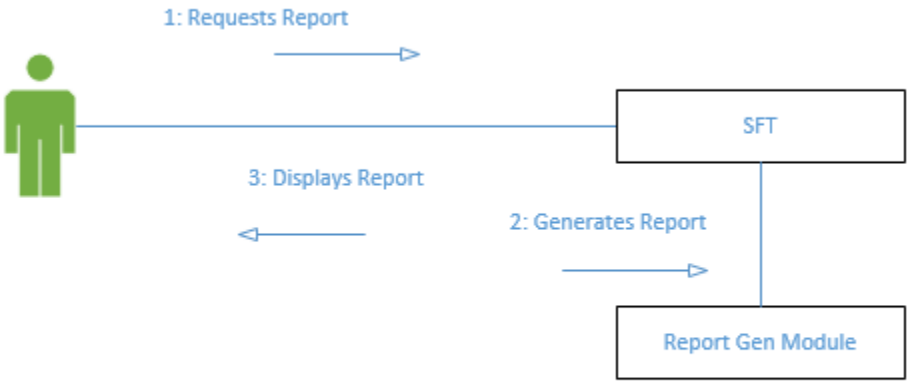

Figure 16: Collaboration Diagram of Report_Gen

4.10 Logical View

4.10.1 State Transition Diagrams

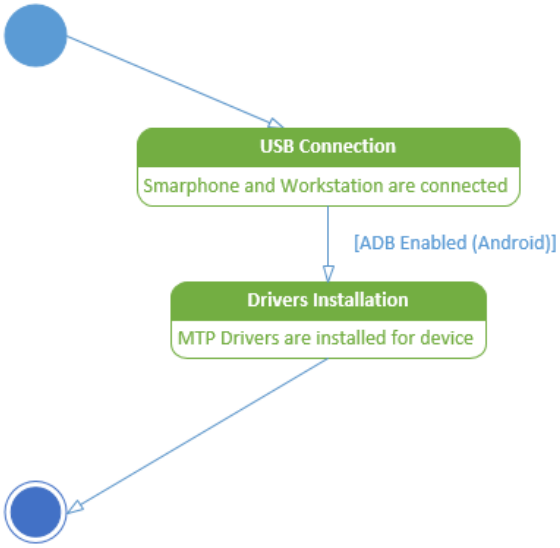

Figure 17: State Transition Diagram of Device_Connect

Figure 18: State Transition Diagram of Report_Gen

4.11 Dynamic View

3.2.3.1 Activity diagram

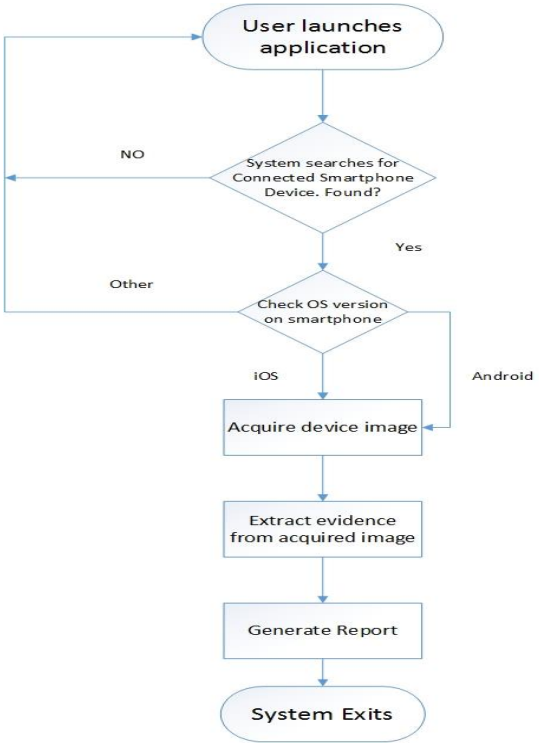

Figure 19: Activity Diagram

3.2.3.2 Data Flow Diagram

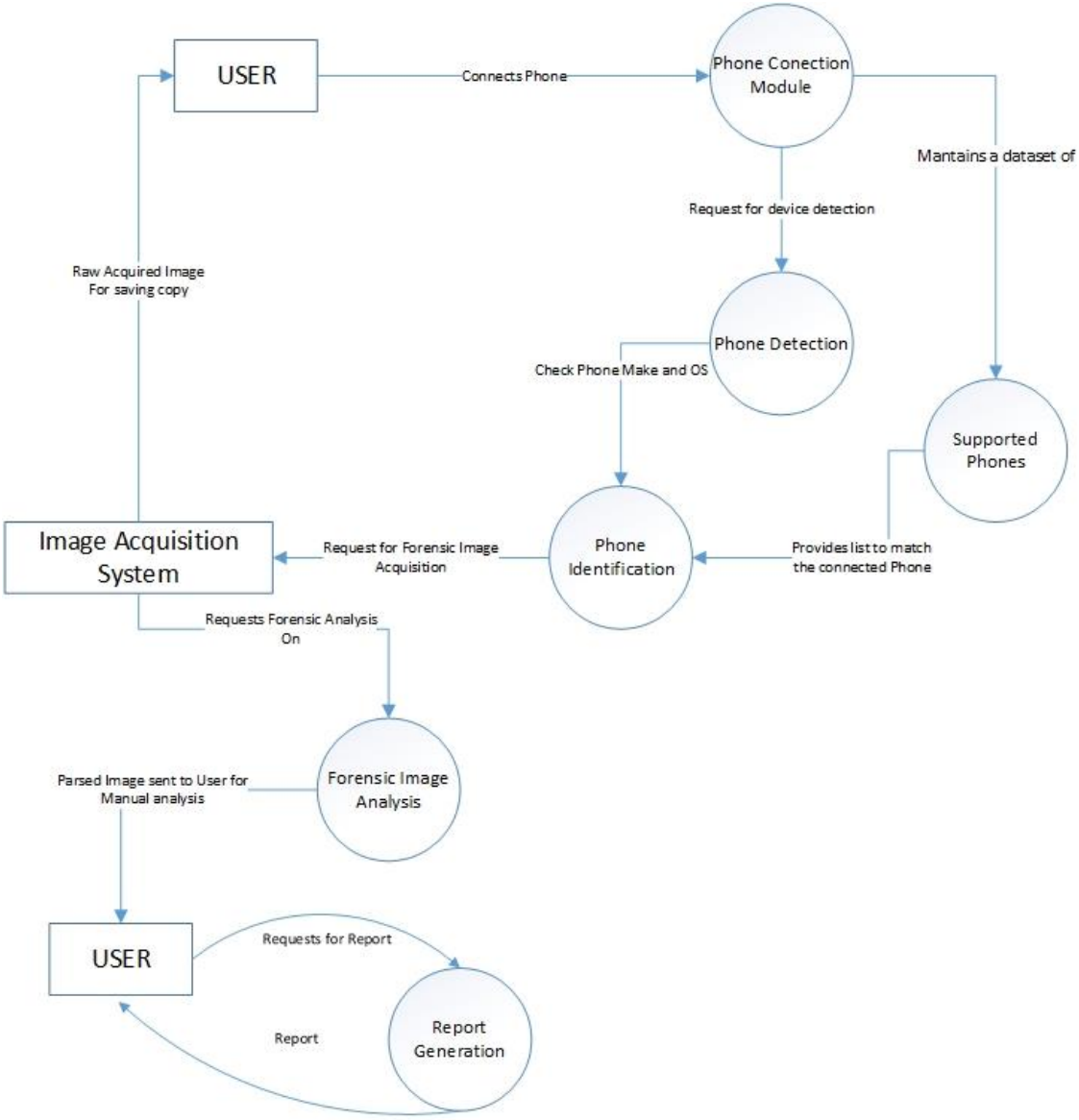

Figure 20: Data Flow Diagram

4.12 Implementation View

4.12.1 System Class Diagram

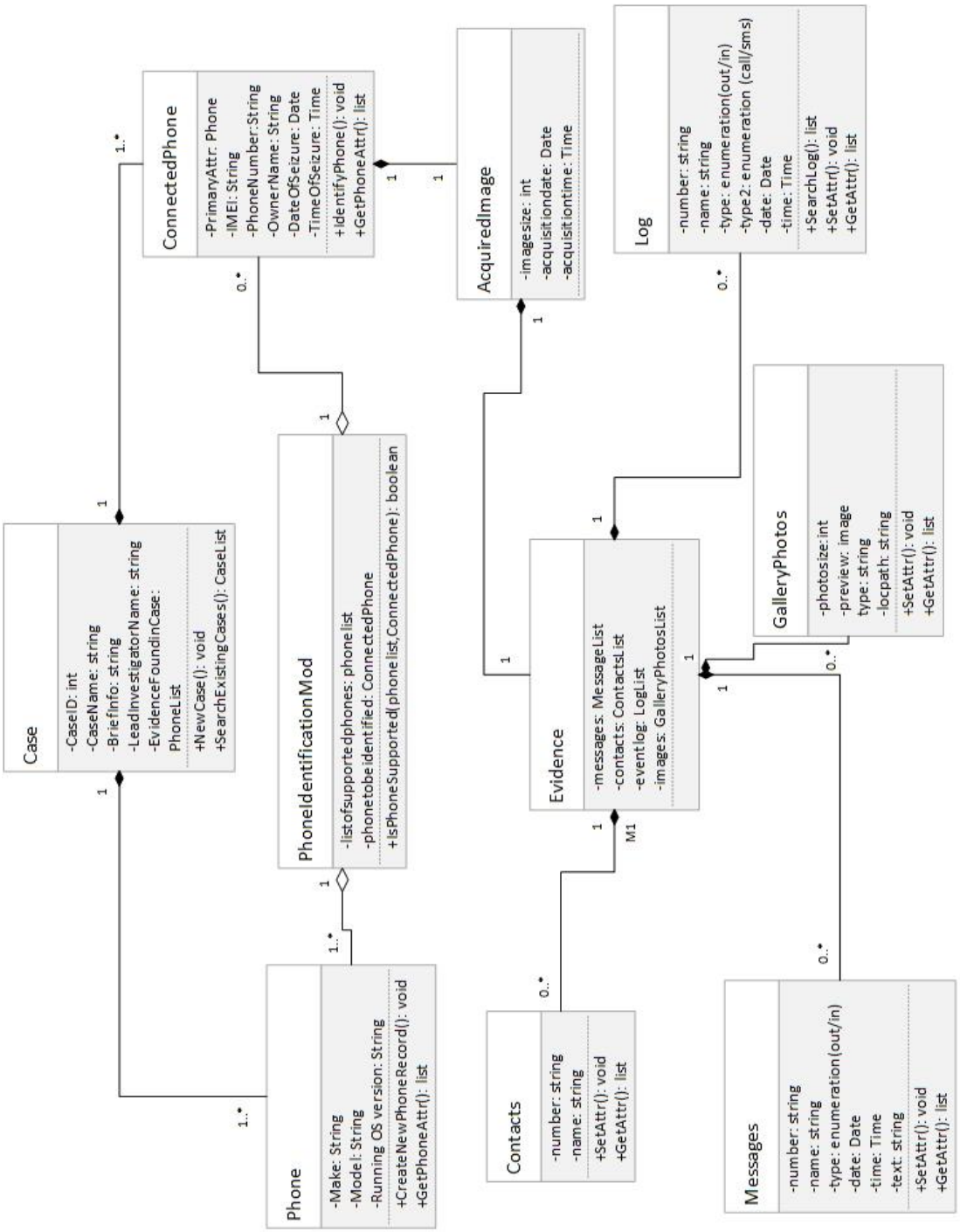

Figure 21: Class Diagram

4.12.2 System Classes Description

Name	Description
Phone	This class contains the preliminary attributes of phones which are supported by the system. It has public function calls CreateNewPhoneRecord() and GetPhoneAttr(). This class is only responsible for storing phone attributes which come in handy in Phone Identification
ConnectedPhone	This class contains an instance of phone class alongwith additional data members like IMEI, Phone Number, OwnerName, Date and Time of Seizure. These data members hold data imported from the phone system information and are chiefly responsible for mantaining connected phones data.
PhoneIdentificaitonMod	This is the main class which plays a crucial role in identification of connected phones once they are detected by the system. It retrieves a list of supported phones from the system in the form of listsupportedphones data member and imports information of the connected phone. Then it calls the function IsPhoneSupported() with list of supported phones and connected phone attributes going in as arguments and returns true if phone connected to the system is supported by it else false.
Case	This class holds information of the case for which the device is being examined such as a unique case ID, investigator name and a list of other evidences seized for investigation regarding the particular case
AcquiredImage	This class holds the attributes of acquired image of the connected phone.
Evidence	This is the most important and main class of the sytem. It holds all the evidence extracted from the smartphone in the form of text messages, contacts and logs
Messages	This class contains text messages extracted from the evidence phone
Log	It contains an event log imported from the phone containing history of all text messages sent or received and all calls received or made from the phone
GalleryPhotos	This class contains the attributes related to photos retrieved from the phone.
Contacts	All contacts imported from the phone are managed by this class

CHAPTER 5

SYSTEM IMPLEMENTATION

5. System Implementation

5.1 Smartphone Forensics: An overview

5.1.1 Forensics for iOS (iPhone)

The forensics for iOS requires the phone to be jail break in order for our project to gain privileged access to user data. In forensics, it is important to preserve the chain of custody that is, maintaining and making sure that contents that have been extracted from evidence in a crime scene are not altered or modified in any way throughout the conduction of the case. Thus affecting system files can lead to change in data and leave traces of presence in the system.

Jail-breaking is a process that only affects the OS files and not the user files, thus in a crime scene, where user data is of importance, jail breaking does not void the chain of custody. All sms, calls, call logs, photos and other file structure remains intact. Out of the several freely available tools in the market, we have used “red sn0w” for jail breaking.

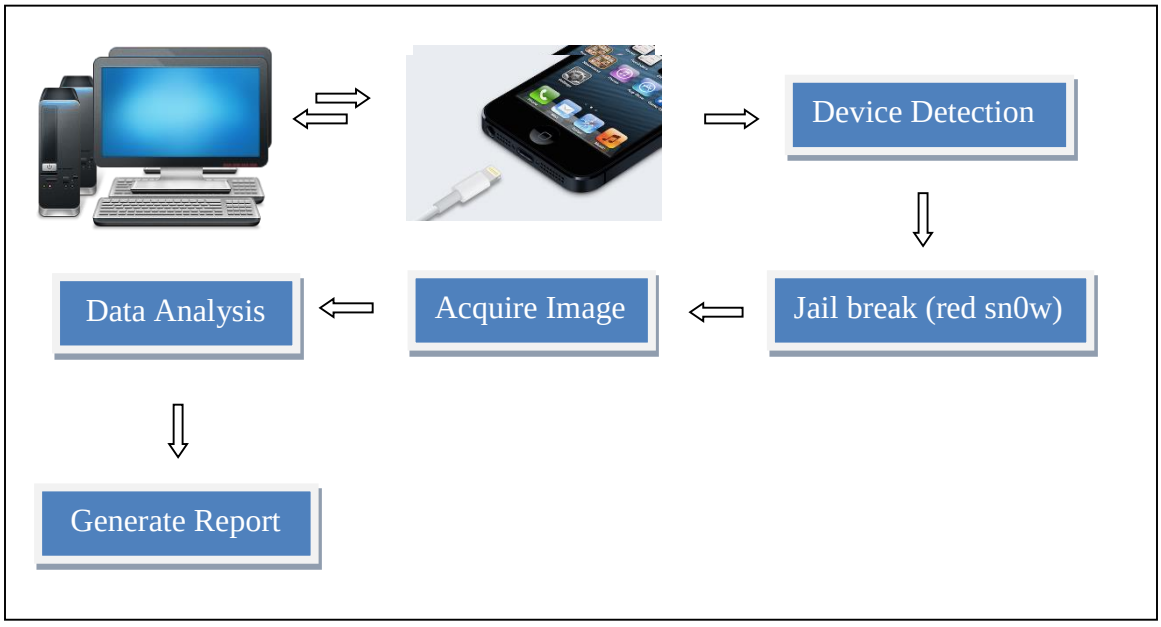

Figure 22 : iOS Forensic Process

5.1.2 Forensics for Android Smartphones

To gain privilieged access for android smartphones, unlike iPhone, it needs to be rooted before establishing connection with workstation for forensic process. Therefore, it is an assumption that Android Phones connected with a computer using our toolkit are rooted. Consequent steps for forensic analysis are similar to that of iOS, where the user has to connect the phone by clicking on the phone connect icon, displayed below:

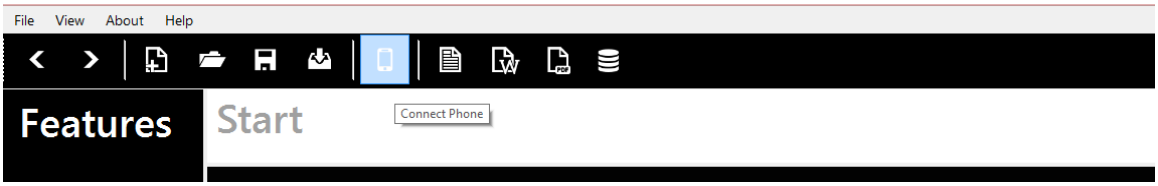

Figure 23: Tool Bar and Phone Connect Icon

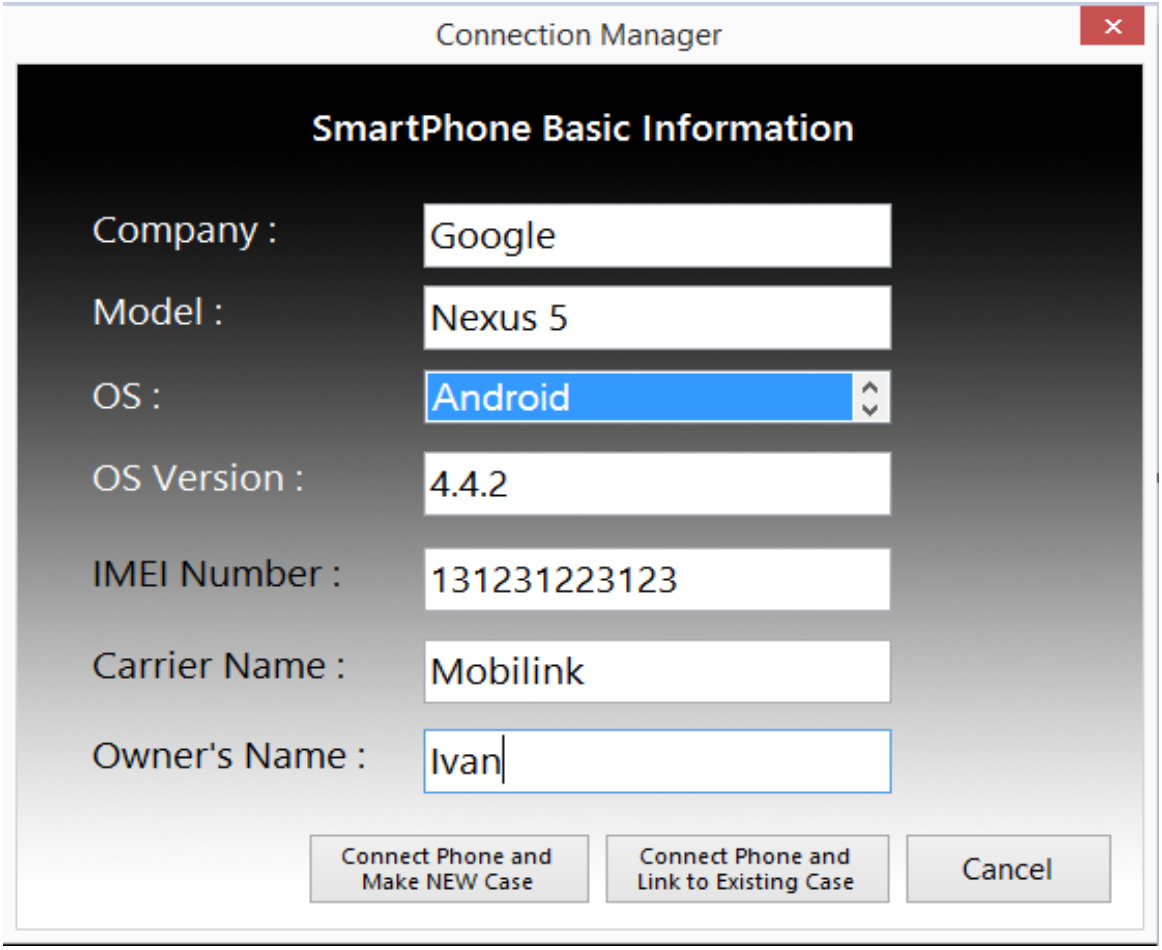

Figure 24: Device Connection Manager

5.1.3 Viewing extracted information

The viewer tab provides several types of views such as database viewer, text viewer, and xml viewer. The figure below shows sms recipients along with other flags.

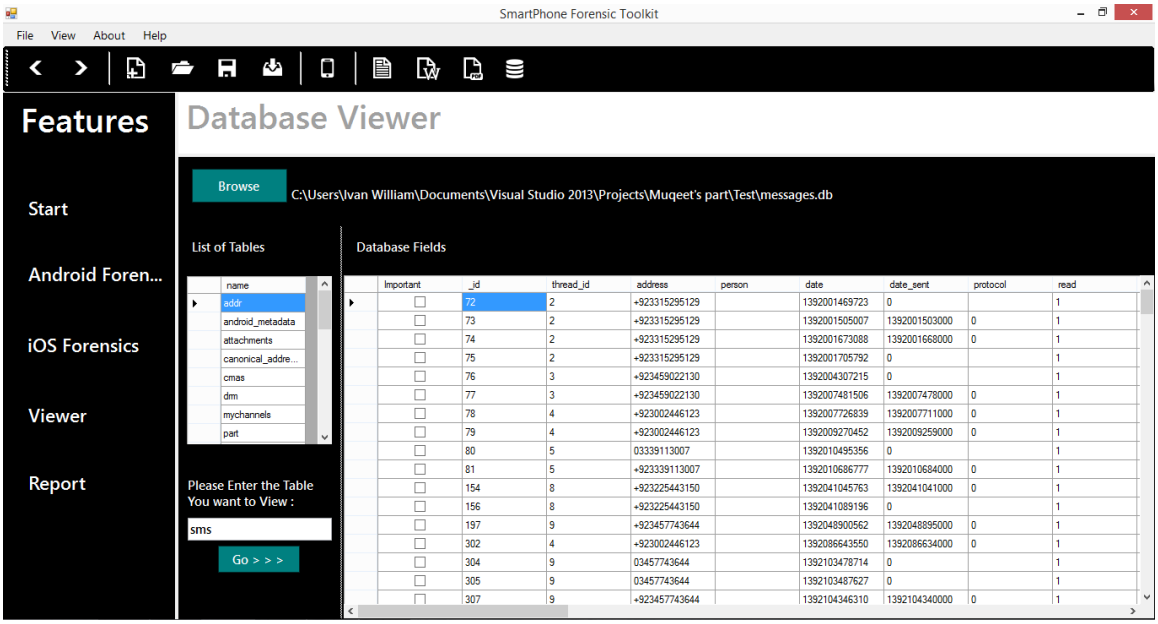

Figure 25 Viewer Tab

CHAPTER 6

TESTING AND EVALUATION

6. Testing

Testing any software project/product/program is essential to check and ensure the provision of intended functionality and quality of software product. We have tested our software product on two levels:

- Interface Testing
- Functional Testing
    - Device attributes recognition Testing
    - Application Testing
    - Data Extraction Testing
    - Viewer Testing

6.1 Interface Testing

Test Case 1

Test Case ID	01
Test Case name	Starting of Application
Input(s)	Press the .exe of the application
Output	Opens the SFT Toolkit application
Sequence of Action(s)	Press .exe of the application from hard disk and toolkit will open.
Result	Success

Table 4: Test Case 1

Execution of test case

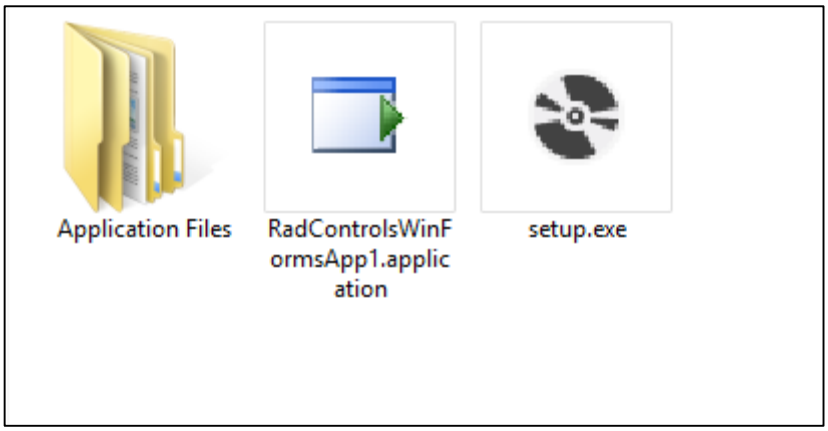

Figure 26: Test case # 01 Execution

Test Case 2

Test Case ID	02
Test Case name	Start New Case
Input(s)	Click on New Case Button
Output	Form with device attributes
Sequence of Action(s)	Click on New Case Button and view device details
Result	Success

Table 5: Test Case 2

Execution of Test Case

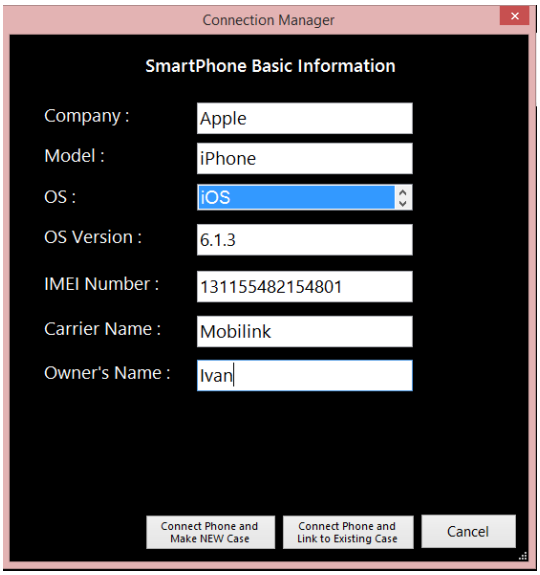

Figure 27: Test case # 02 Execution

Test case 3

Test Case ID	03
Test Case name	Open Existing Case
Input(s)	Click on open case button
Output	Display archived cases
Sequence of Action(s)	Click on open case button and new window will open displaying existing cases.
Result	Success

Table 6: Test Case

Test Case 4

Test Case ID	04
Test Case name	Device Connection
Input(s)	Connect smartphone via usb and click on phone connect button
Output	New Form for selecting OS
Sequence of Action(s)	Connect smartphone via usb -> Click on phone connect -> Choose OS
Result	Success

Table 7: Test Case 4

Execution of test case

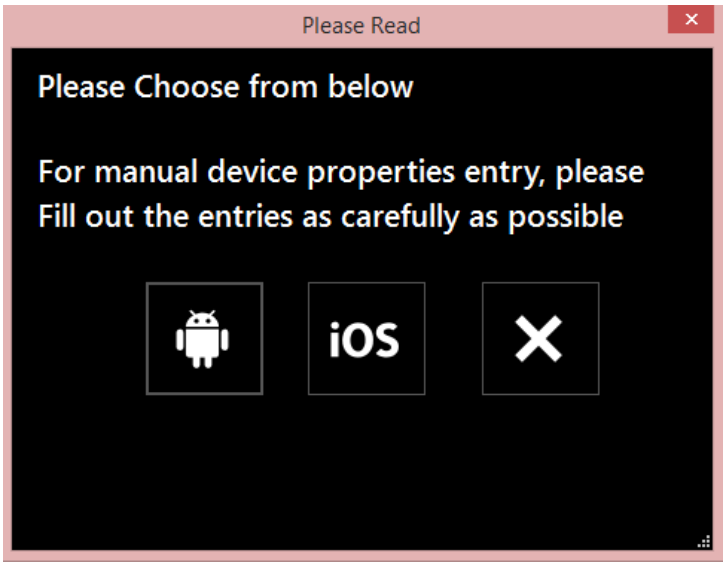

Figure 28: Test case # 04 Execution

Test Case 5

Test Case ID	05
Test Case name	Text Viewer
Input(s)	Click text viewer button
Output	Displays text file from extracted data
Sequence of Action(s)	Click on Viewer Icon/Viewer Tab -> Click on Text Viewer
Result	Success

Table 8: Test Case 5

Execution of test case

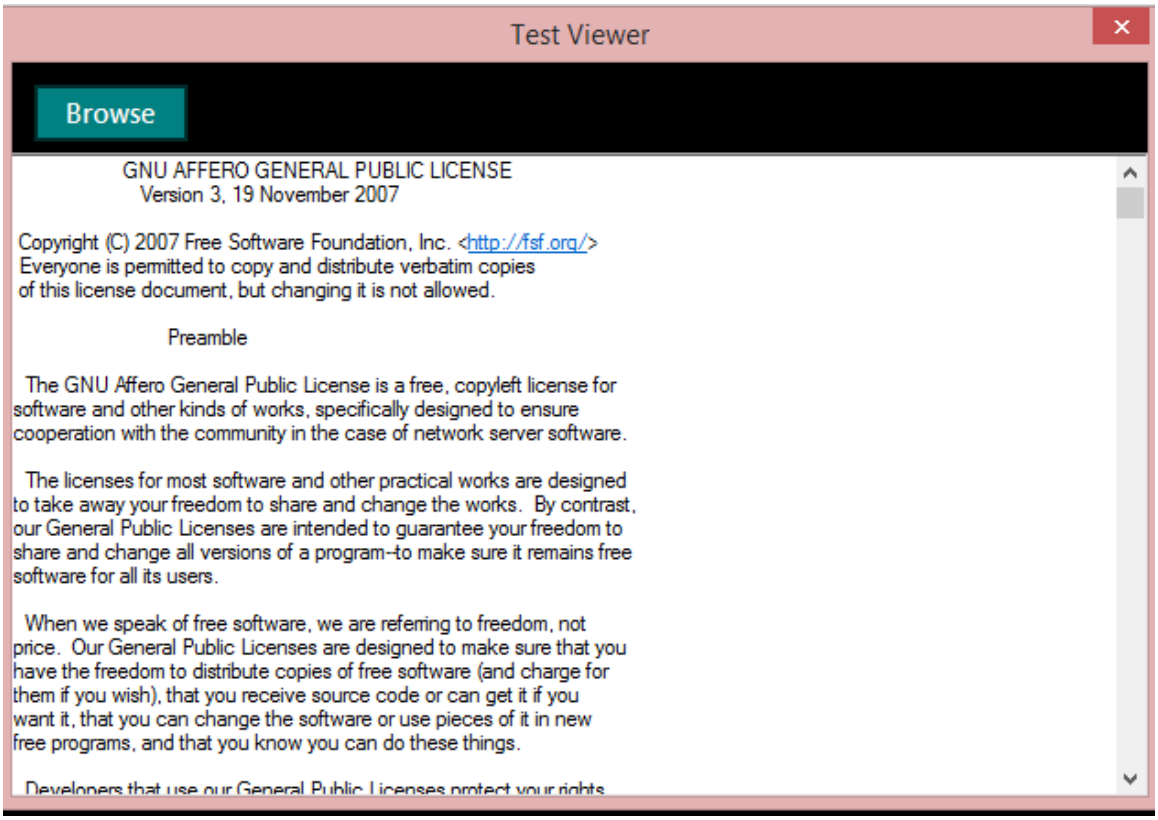

Figure 29: Test case # 05 Execution

Test Case 6

Test Case ID	06
Test Case name	Word Viewer
Input(s)	Click on Word Viewer button
Output	Displays word file from extracted data

Sequence of Action(s)	Click on Word Viewer Icon/Tab -> Click on Word Viewer Button
Result	Success

Table 9: Test Case 6

Execution of test case

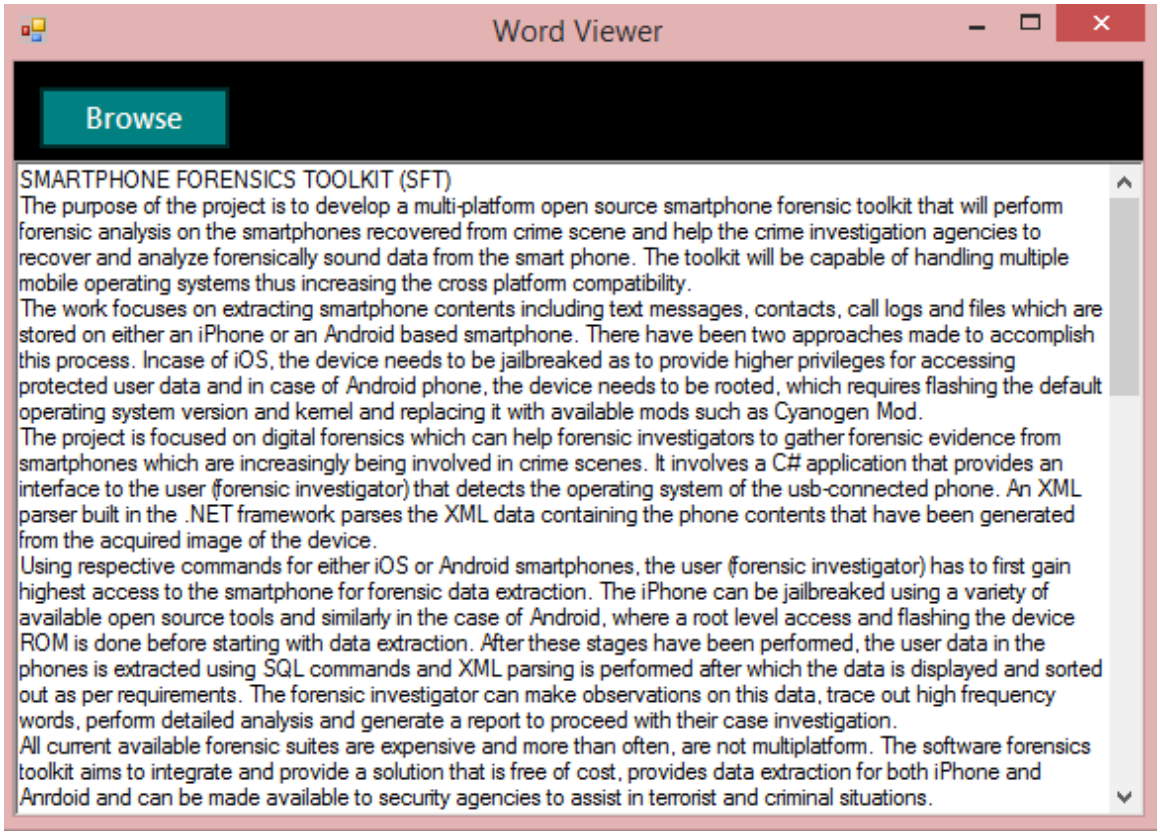

Figure 30: Test case # 06 Execution

Test Case 7

Test Case ID	06
Test Case name	Database Viewer
Input(s)	Click on Database Viewer button
Output	Displays database from a .db file
Sequence of Action(s)	Click on Database Icon/Viewer Tab -> Click on Database Viewer Button
Result	Success

Table 10: Test Case 7

Execution of Test Case

Figure 31: Test Case #7 Execution

6.2 Functional Testing

6.2.1 Establish New Case

Test Case: 8

Test Case ID	01
Test Case name	New Case testing
Input(s)	Device Connection and making new case
Output	New Case is created against evidence
Sequence of Action(s)	Device Connect->New Case->Start New Case
Result	Success

Table 11: Test Case 8

Execution of Test Case

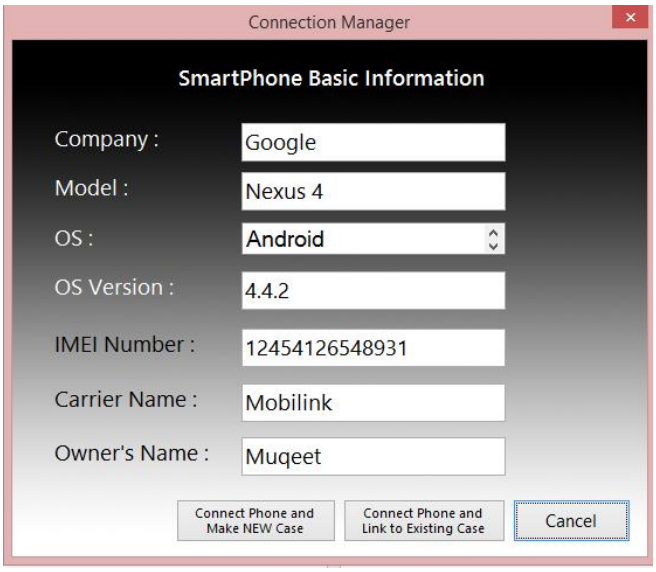

Figure 32: Test case # 8 Execution

Test Case: 9

Test Case ID	02
Test Case name	Link to existing case
Input(s)	Device attributes/New Case to be linked
Output	Existing Case is created against evidence
Sequence of Action(s)	Device Connect->New Case->Start New Case
Result	Success

Table 12: Test Case 9

Execution of Test Case

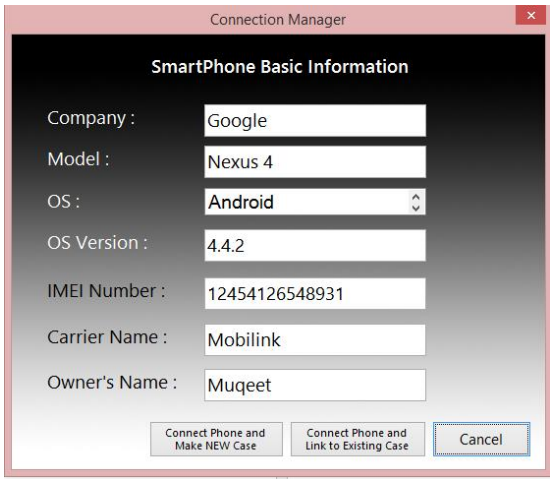

Figure 33: Case # 9

Test Case: 10

Test Case ID	03
Test Case name	Open evidence (existing case)
Input(s)	Evidence_info table
Output	Existing Evidence information
Sequence of Action(s)	Click on open evidence button -> Select evidence
Result	Success

Table 13: Test Case 10

Execution of Test Case

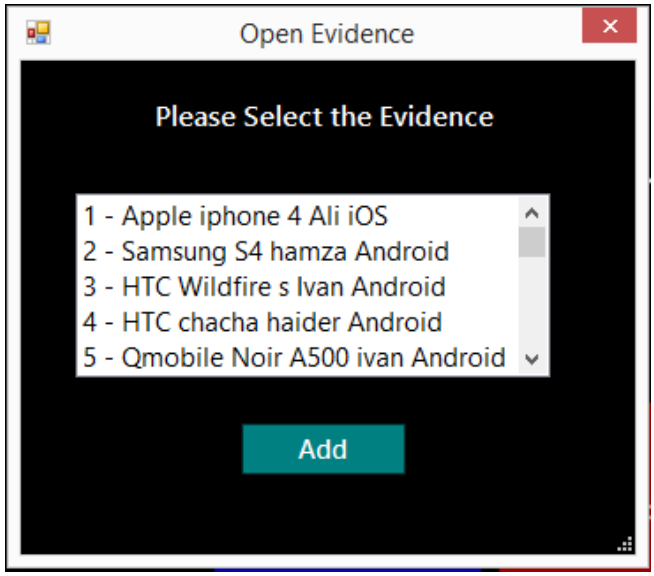

Figure 34: Case # 10

Test Case: 11

Test Case ID	04
Test Case name	iPhone Jail breaking using red sn0w
Input(s)	Phone Connect and New Case
Output	iPhone is jail breaked
Sequence of Action(s)	Connect iPhone -> Select New Case -> Click on Jail Break button
Result	Success

Table 14: Test Case 11

Execution of Test Case

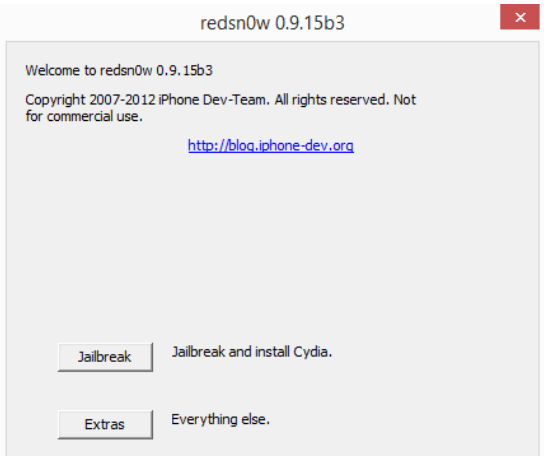

Figure 35: Case # 11

Test Case: 12

Test Case ID	05
Test Case name	Acquiring full/Specific Data (iPhone)
Input(s)	iPhone connection and data (sms, calls)
Output	Database display of extracted data
Sequence of Action(s)	Connect iPhone -> Select New Case -> Click on Jail Break button -> Click on Acquire Image/Acquire Specific Files
Result	Success

Table 15: Test Case 12

Execution of Test Case

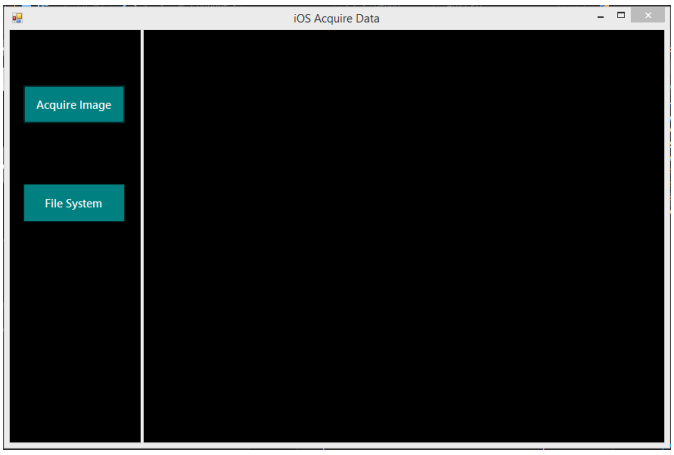

Figure 36: Case # 12

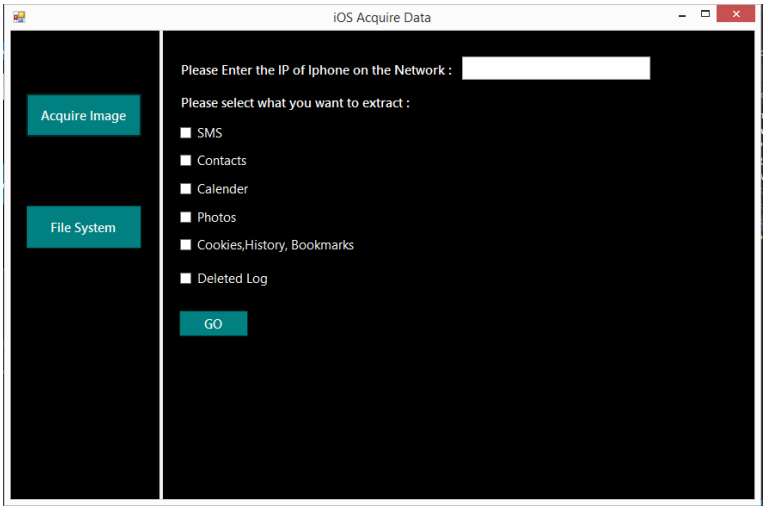

Figure 37: Case # 12

Test Case: 13

Test Case ID	06
Test Case name	Acquiring full/Specific Data (iPhone)
Input(s)	iPhone connection and data (sms, calls)
Output	Database display of extracted data
Sequence of Action(s)	Connect iPhone -> Select New Case -> Click on Jail Break button -> Click on Acquire Image/Acquire Specific Files
Result	Success

Table 16: Test Case 13

APPENDIX A-1

USER MANUAL


SMARTPHONE FORENSICS
TOOLKIT
USER MANUAL

TABLE OF CONTENTS

1. Reading Instructions.....66

2. Installations.....66

3. How to use the system.....66

4. Procedures.....67

4.1 Phone Connection.....67

4.2 Start New Case….....67

4.3 Android Forensics.....67

4.4 iOS Forensics.....68

4.5 Data Analysis and Viewers.....68

4.6 Reports…….....68

1. Reading Instructions

This Manual is a guide to the system "Smartphone Forensics Toolkit" It contains essential instructions for setup and operations.

The system provides a user friendly interface which allows you to directly interact and monitor the system.

This Manual should be read in the order given.

2. Installation

For installation of our software, use the setup file 'SFT.msi'. This file should directly be installed on the hard disk. Click on the setup file and click install to complete the installation process.

3. How to use the system

Operation of our toolkit comprises of the following steps:

1. Connection of smartphone with workstation.
2. Device detection and display of attributes.
3. Starting of new case and entering owner's information.
4. Acquiring image (Click on Android Forensics for Android Phone and iOS Forensics for iPhone).
5. In case of iPhone, jailbreak the device by clicking on the jailbreak button.
6. View information in the viewer's tab of either text, database, xml or pdf/word format.
7. Click on reports tab to view past reports.

4. Procedure

The procedures mentioned below are specific for iOS and Android forensics respectively.

4.1 Phone Connect

Steps for the successful completion of Phone Connect are given as under:

1. Connect the smartphone to the workstation via a USB.
2. In case of Android, ensure ADB (Android Debugging Bridge) is enabled.
3. Click on the phone connect icon placed in the top row of the application.
4. Click on the respective OS platform of the connected device.
5. Device is now connected.

4.2 Start New Case/Link to Another Case

Steps for the successful completion of new case are given as under:

1. Make sure device is connected and requisite steps have been followed as in Phone Connect.
2. In the device attributes section, enter owner's name.
3. Click on start new case to begin new case.
4. Click on link to another case, to associate this instance with an already conducted case.
5. Case has now been created.

4.3 Android Forensics

Steps for the successful completion of Android Forensics are given as under:

1. Make sure the device is connected to the workstation via usb.

2. Start New Case/Link to an old case as required.
3. After completion, click on Android Forensics Tab on left side of the display.
4. Click on Acquire Image to begin data extraction from device.
5. Image will be successfully extracted.

4.4 iOS Forensics

Steps for the successful completion of iOS Forensics are given as under:

1. Make sure an iPhone is connected to the workstation via usb.
2. Start new case/link to old case as required.
3. Now select the iOS Forensics Tab.
4. Click on the jailbreak button to gain administrator privileges of the device.
5. Follow through the consequent steps of Red Sn0w Tool for Jail breaking.
6. Click on Acquire Full Image to extract all data.
7. Click on Acquire Specific files to apply filter and choose from sms/calls etc.

4.5 Data Analysis and Viewers

Steps for the successful completion of viewing content are given as under:

1. Make sure Image has been successfully extracted from iPhone/Android Device.
2. Select from available buttons, XML Viewer, Text Viewer, Database or PDF Viewer to read files of this format as present on the device storage.
3. Analyze information as required and use filters to search for specific words.

4.6 Reporting

Steps for the successful completion of Reporting application are given as under:

1. Click on reporting tab on left side of the display.