SMS SPAMMING APPLICATION

By

Ibrahim Irfan
CSU-F15-104
Bilal Ahmed
CSU-F15-105

A Project Report submitted to the
DEPARTMENT OF COMPUTER SCIENCE & INFORMATION TECHNOLOGY
in partial fulfillment of the requirements for the degree of
BACHELORS OF SCIENCE IN COMPUTER SCIENCE

Faculty of Computer Science & Information Technology
University of Lahore
Islamabad
December, 2019

DEDICICATION

Every challenge needs self-effort as well as friendly advice from an elder. My humble dedication is to those who are very dear to me, my parents, my siblings, my family and all those who guide me to the right path, support me, encourage me and motivate me to do better in my life.

DECLARATION

I hereby declare that this application neither as whole nor as a part has been copied out from any kind of source. I further declare that I have developed this application and written this report on the basis of my own effort, under the supervision of my project supervisor. If any part of this system is proved to be copied out from any product found to be the reproduction of someone else, I shall be held responsible.

Ibrahim Irfan

CSU-F15-104

Bilal Ahmed

CSU-F15-105

December, 2019

CERTIFICATE OF APPROVAL

It is certified that the project titled "SMS Spamming Application" carried out by Ibrahim Irfan, Reg. No. 70063514 and Bilal Ahmed, Reg. No. 70063567, under the supervision of Mr. Umer Farooq and Dr. Jawad, University of Lahore, Islamabad, is fully adequate, in scope and in quality, as a final year project for the degree of BS of Computer Science.

Supervisor: ------------------------
Dr. Syed Jawad Hussain
Associate Professor
Dept. of CS & IT
University of Lahore, Islamabad

Co-Supervisor -------------------------
Mr. Umer Farooq
Lecturer
Dept. of CS & IT
University of Lahore, Islamabad

Internal Examiner 1: ------------------------
Ms. Maida Khalid
Lecturer
Dept. of CS & IT
University of Lahore, Islamabad

Internal Examiner 2: ------------------------
Ms. Fazeela Tariq
Lecturer
Dept. of CS & IT
University of Lahore, Islamabad

Project Coordinator: ------------------------
Ms. Mehreen Shah
Lecturer
Dept. of CS & IT
University of Lahore, Islamabad

HOD: ------------------------
Dr. Muhammad Yaqoob Wani
Associate Professor
Dept. of CS & IT
University of Lahore, Islamabad

ACKNOWLEDGMENT

I would like to thank my supervisor Dr. Syed Jawad Hussain and co-supervisor Mr. Umer Farooq in assisting me and helping me fulfill the requirements of this project and for being cooperative with me and my colleague Bilal Ahmed during the course of this project.

ABSTRACT

In the recent era, there has been an increase in the popularity of Android smartphones and in this age and time almost everybody around the world uses smartphones. They store all their important data that includes contacts, emails, accounts and passwords on their device. The problem that has risen for all those individuals who use Android smartphones is SMS spams. These spams can be harmful in a sense that they can damage your device or steal some important information. The main reason behind us working on this project is practical experience. To counter this issue we collected data sets, and then we compiled the data using Random Forest Model, SVM, and Logistic Regression in Python. With these approaches we managed to find the accuracy and precision of our data sets. We then developed an Android app in which a user locates spams that have already been stored on his device or are incoming. We developed this application in Android Studio using Java. Once our project is completed, users will be able to download a free of cost app from the Google Playstore which will assist them in detecting spams on their device and then block them. This requires the user first enter text of a certain SMS stored on his device in the app. The app then scans the text and determines if it is a spam or not. Once the app fully scans the text entered by the user, the user is notified whether it is a spam or not. Our app will make it easier for most Android smartphone users to successfully detect SMS spams stored on their devices and will provide them with the capability to delete those spams.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ACRONYMS

AI	Artificial Intelligence
API	Application Program Interface
IDE	Integrated Development Environment
NLP	Natural Language Processing
SDK	Software Development Kit
SMS	Short Message Service
SVM	Support Vector Machine

Chapter 1

INTRODUCTION

SMS Spamming Application is an Android app that can be used by Android smartphone users to protect their Android devices from any harmful spams. The SMS Spamming Application will be a mobile based application developed exclusively for mobile devices built with the Android operating system. Spamming has become a major problem in today's world of smartphones. These spams usually infect a user's device and the creators of these can then easily steal confidential data stored on a user's device without him being aware of it. This project will provide Android smartphone users with the ability to scan for and detect spams on their device and once these spams are detected, users can then block these spams to keep their important data which is stored on their Android smartphone safe and secure. First raw data will be collected and compiled in Python where we will find the accuracy and precision using classification in SVM [Vladimir N. Vapnik and Alexey Ya. Chervonenkis in 1963] and Random Forest [A classification method proposed by Ho in 1995]. After our work in Python is complete, we will design an application in Android Studio which the user will have the option to interact with. The application will allow the user to enter text received in SMS messages to check whether it is a spam or not. This next chapter will describe to the reader about the introduction of the project, its motivation, its objectives and structure of the report. This chapter gives a brief introduction that what is this work area along with its needs and application. In the last section of this chapter, the summary of chapter 1 with brief sentences is given.

1.1 Overview

This project is to assist Android users in eliminating SMS spams from their devices. In the modern age, there has been an increase in communication through SMS messaging. However not many people know whether all the SMS messages they receive are authentic or not so the title of this is report is SMS Spamming Application. The main goal of our work is to reduce the probability of malicious SMS spams infecting and harming a user's Android smartphone. We intend to accomplish this goal by designing and developing an Android based application created specifically for Android smartphones. Once the project is complete, users will be able to install the app and it will assist Android users in successfully blocking any SMS spam.

1.2 Statement of Problem

SMS Spamming has become a major problem in today's day and age of smartphones. It is constantly disrupting the constant flow of smartphone usage and many people who use these smartphones keep all their confidential data stored on their device. These spams can damage a user's device by assisting the creator of these spams in stealing confidential information such accounts, contacts, and even passwords from any user's device. The proposed idea to counter the rising issue is to collect data sets and compile them. Once data compilation is complete, an Android application will be developed. The reason behind the idea of creating an SMS Spamming Application that can detect and then block spams is practical experience.

1.3 Definitions, Acronyms, and Abbreviations

- **Android:** A Linux based operating system created by Google Inc. for smartphones.
- **Android application:** An app developed using Python (occasionally) and Java exclusively for Android devices.
- **Short Message Service (SMS):** A type of text message component for mobile devices.
- **Spam:** Malicious and unnecessary messages sent over the Internet, usually to users in large numbers, for the reasons for publicizing, scamming, and spreading viruses.

1.4 General Description

This section depicts the capacities, points and destinations of the ventures. It likewise incorporates the imperatives, prerequisites, as well as assumptions and dependencies of the project.

1.4.1 Product Perspective

Spamming Application is an Android application developed exclusively for smartphones built with the Android operating system. While the original application is built for Android, there will also be some Machine Learning involved for the purpose of successfully detecting spam messages. The primary scope of this project includes developing an Android app that will provide an Android user with the means to detect spams and block them from harming his or device. Once the app is fully developed, Android smartphone users will have the option to download the app through the Google Playstore for free and install it on their respective Android devices. After installation, users can use the app to delete spams.

1.4.2 Product Functions

- User receives SMS message.
- App then detects SMS.
- It gives user the option to save the spam message or block it.
- App blocks the spam and send it to spam filter instantly.

1.4.3 User Characteristics

Android users – These users possess an android smartphone and constantly receive messages without knowing if it is a spam or not.

1.4.4 General Constraints

1. System will need only SMS Spam messages for functionality.
2. Application must be capable to run on any version of Android 4.4 and above.
3. Application must detect SMS Spam messages.
4. System must perform accurate results.
5. System must be fully and well trained.
6. Dataset must be categorized according to classes being made.
7. Application must be able to detect any SMS Spam and then provide the user with the option to store it or block it.

1.4.5 Assumptions and Dependencies

We will assume that the data sets will be in CSV format. Dependencies include training data sets, that is before the system can begin processing and display any kind of output, the development team will need to provide input in the form of data sets that include known spams (ads) and unknown spams (casual conversations between two individuals) so that the system can perform data training.

1.5 Purpose of the project

The idea behind our work rose due to the clear rise in SMS spams that have started to harm many user's Android devices right in front of our eyes. The main goal of our work is to reduce the probability of malicious SMS spams infecting and harming a user's Android smartphone. There have been previously developed applications for Android based smartphones in the same context to SMS Spam detection but those applications have been proven not to be as efficient and accurate as users would want them to be. And because of these inefficient and inaccurate applications, the ongoing problem of SMS spam infiltration is yet to be solved. With our app, users will have the ability to delete any anonymous SMS messages or malicious spams stored on their devices. These can include advertisement messages from or any messages from an anonymous source that the receiver is unaware of.

1.6 Applications of the project

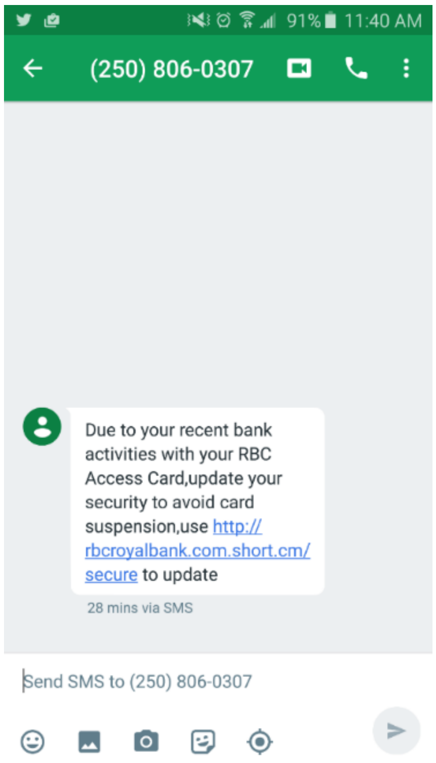
This project will generally teach individuals a lesson on how to keep all their important property, possessions, and belongings safe and secure and what steps need to be taken to make sure that an individual can successfully keep all his property out of the wrong hands and prevent any sort of unauthorized access and illegal possession as these kind of issues may cause problems for the individual. However, in particular, this project will teach Android smartphone users how to keep their confidential data safe and secure from malicious SMS spams and the ones who create and send these spams to several Android smartphones.

1.7 Theoretical bases and Organization

This thesis is set to include 7 chapters altogether. The first chapter will explain the introduction, a brief overview, purpose, problem statement of the project. It will also explain what kind of impact our project is set to have in real daily life. The second chapter will contain the literature review of our project and will explain any kind of technologies or other work that is related to the field based on which we are making our project. The third chapter will give a brief description of all the tools and techniques we will be using to complete the required objectives and goals of our project. The fourth chapter will include the methodologies, implementation procedures used and details about the hardware and software that will be used in our project. The fifth chapter will include brief explanations about system testing. The sixth chapter will explain the findings, results, and conclusion of our project. The seventh and final chapter of this thesis explains what future work is yet to be done and how we will complete it. The final pages of this document explain the references that guided us throughout the project and the terminologies of the report. The entire thesis is based on the project, SMS Spam Detection Application and covers all the steps taken to develop the app such as the systems used and what their hardware specifications are. The software tools that were used in duration of the project, as well as the programming languages the project was implement in.

1.8 Summary

This chapter basically covers the introduction and a brief overview of our project, the purpose behind it and why we chose this topic. It also explains in detail the general description of the project what difference it will bring in the future. SMS Spamming Application is an Android app that can be utilized by Android smartphone users to protect their Android devices from any harmful spams. The main goal of our work is to reduce the probability of malicious SMS spams infecting and harming a user's Android smartphone. SMS Spamming has become a major problem in today's day and age of smartphones. It is constantly disrupting the constant flow of smartphone usage and many people who use these smartphones to keep all their confidential data stored on their device. The proposed idea to counter the rising issue is to collect data sets and compile them. Once data compilation is complete, an Android application will be developed. The reason behind the idea of creating an SMS Spamming Application that can detect and then block spams is personal and practical experience.



Due to your recent bank activities with your RBC Access Card,update your security to avoid card suspension,use http://rbcroyalbank.com.short.cm/secure to update

28 mins via SMS

Figure 1: SMS Spam

Figure 1 explains a sample SMS spam in which the sender as well as the message are both completely anonymous to the reader. This could be a malicious spam.

Chapter 2

LITERATURE REVIEW

This chapter explains the literature review of our project. This chapter also covers brief work of other researches in this field. A literature review is a comprehensive summary of previous studies on a subject. The literature review briefly explains the related studies, books, and different subjects pertinent to certain areas of research. The survey mentions, depicts, condenses, impartially assesses and explains this recent study. Give establishment of information on theme. Recognize zones of earlier grant to anticipate duplication and offer credit to different scientists. Recognize irregularities:

To start our project, we went through various research papers on Machine Learning and SMS spam detection such as SMS spam filtering techniques using Machine Learning by Hedieh Sajedi and SMS spam detection approach by Houshmand Shirani-Mehr. Our main purpose behind reviewing these articles is to get the basic idea of how spam detection and machine learning work and what other resources we will need to kick off the development and design of our Android mobile application. Once our research was complete, we started collecting data.

2.1 Related Technologies

There are several technologies that are very closely associated with the work done in regards to our project. Some of them are Support Vector Machine (SVM) and Random Forest Algorithms. These techniques are an integral part of machine learning and have a crucial role in the study of SMS spam detection.

2.1.1 Support Vector Machine

Support Vector Machine is a managed learning model with related learning calculations that dissect information utilized for characterization and relapse examination. Given a lot of preparing models, each set apart as having a place with either of two classifications, an SVM preparing calculation constructs a model that relegates new guides to one class or the other, making it a non-probabilistic double direct classifier (in spite of the fact that techniques, for example, Platt scaling exist to utilize SVM in a probabilistic characterization setting). An SVM model is a portrayal of the models as focuses in space, mapped with the goal that the instances of the different classes are isolated by an unmistakable hole that is as wide as could be allowed.

New models are then mapped into that equivalent space and anticipated to have a place with a class dependent on which side of the hole they fall. Notwithstanding performing straight order, SVMs can effectively play out a non-direct grouping utilizing what is known as the part stunt, certainly mapping their contributions to high-dimensional element spaces.

2.1.2 Random Forest Model

Random Forest is a regulated order model. As the name proposes, this model makes the forest with various trees. By and large, the more trees in the timberland, the heartier the forest resembles. Similarly, in this classifier, the higher the quantity of trees in the timberland, the higher the precision results will be. On the off chance that you realize the choice tree calculation. You may be believing are we making increasingly number of choice trees and how might we make more quantities of choice trees. As all the computation of hubs determination will be same for the equivalent dataset.

2.1.3 History

The actual Support Vector Machine (SVM) was made by Vladimir N. Vapnik and Alexey Ya. Chervonenkis in 1963. In 1992, Bernhard E. Boser, Isabelle M. Guyon and Vladimir N. Vapnik recommended an approach to make nonlinear classifiers by applying the bit stunt to most extreme edge hyperplanes.] The present standard manifestation (delicate edge) was proposed by Corinna Cortes and Vapnik in 1993 and distributed in 1995.

2.2 Related Projects

There are quite a few projects being done in related to SMS spam detection like detecting spams using H2O framework, SMS spam detection using Natural Language Processing (NLP).

2.3 Related Studies

There are many other researching doing studies in the field SMS spam detection. Some are using Artificial Immune systems. In AI, counterfeit invulnerable frameworks (AIS) are a class of computationally astute, rule-based AI frameworks roused by the standards and procedures of the vertebrate safe framework. The calculations are regularly displayed after the resistant framework's attributes of learning and memory for use in critical thinking. Other researchers are taking the NLP approach. This is a subfield of computer science, information engineering, as well as artificial intelligence concerned with the interactions between systems and people in particular how to program systems to process and analyze large amounts of content and data.

2.4 Project Limitations and Bottlenecks

The storage of time is a major issue in analyzing large amounts of data. Data availability: The identification of the test of spam messages in the claims is a very hard and time-consuming task. It also involves carefully scanning hundreds of SMS spams. The main limitation of "modern NLP technologies" is their dependency on huge computing power. Artificial neural networks are far from matching the efficiency of the brain when it comes to process terabytes of data. As a consequence, deep learning-based NLP tools are reduced to analyze samples of the Big Text Data, which, in the case of email surveillance for example, is just not enough. Precision is a problem too: conventional vector-based approaches are not fine-grained enough and deliver too many "wrong hits". To take again the email surveillance example, it means a flood of false positives and the necessity to hire a bunch of people just to sort the true from the false alerts. But the algorithms that they used were not able to give the correct result. Some algorithms used to give result against fresh meat and not for spoil meat means the algorithms that they used were not perfect and did not give better result as compared to modern algorithms. Most SMS spam detection apps were not as efficient in the past and hence the user could never tell whether the message that he received on his device was a malicious spam or not and this has caused many problems in today's age of smartphone technology. However due to the rise in SMS spams, many Android developers have starting creating apps that are more efficient than previous ones that were developed in the past.

2.5 Summary

In this chapter, we discussed what research we did and what articles we read in SMS spam detection before beginning our core work of our project. We also covered two types of technologies that are closely associated to the field our project is based on, what are projects and studies are being done that associated with this field and what their limitations are. To start our project, we went through various research papers on Machine Learning and SMS spam detection such as SMS spam filtering techniques using Machine Learning by Hedieh Sajedi and SMS spam detection approach by Houshmand Shirani Mehr. There are several technologies that are very closely associated with the work done in regards to our project. Some of them are Support Vector Machine (SVM) and Random Forest Models. These techniques are an integral part of machine learning and have a crucial role in the study of SMS spam detection.

Chapter 3

TOOLS AND TECHNIQUES

We used several software applications to begin the commencement of our work. We did our practical work on multiple operating systems and Laptop computers.

3.1 Hardware used with technical specifications

We used two different laptops as hardware technology during our project, an Acer Nitro V gaming notebook for the practical programming and design interface of our project whereas an HP notebook was preferred for data collection and compilation. Since our app was designed in Android Studio, which is a heavy software, a gaming laptop was preferred as it has very high and heavy hardware specifications because many Android applications require lots of space as well as memory to run successfully. We did most of our work on two operating systems, Windows 10 Enterprise, 64-bit and Ubuntu 16.04. The two main software we used were Sublime Text and Android Studio. PC equipment details are critical representations of the PC's hardware and software. Processor speed, model and maker. Processor speed is normally shown in gigahertz (GHz). Unpredictable Access Memory (RAM), this is ordinarily shown in gigabytes (GB). If a PC has more RAM, it can do more at a time. Hard circle (now and again called ROM) space. This is regularly shown in gigabytes (GB) and alludes for the most part to the measure of data (like reports, music and other information) your PC can hold. Different determinations may incorporate arrange (Ethernet or Wi-Fi) connectors or sound and video capacities. Minimum System Requirement is the specs your pc needs just to play the game (or equivalent performance parts). Recommended System Requirement on the other hand, is the specs recommended you should have (or better) to play the game on full resolution and get 60fps. When developing a project, whether it is an app or website, the developer should always keep in mind the hardware specifications of his system for these have a great role to play for the execution as well as proper running of the software. As there are many heavy software tools in today’s age of computer technology that may run very slowly on systems with minimum specifications or for the worst-case scenario, may not run at all and this can cause some serious problems for professional developers during the course of any kind of software development.

3.1.1 Acer Nitro V Specifications

Table 3.1: Acer Nitro V Specifications

Type	Notebook
SSD	128 GB
HDD	1 TB
RAM	16 GB
Graphic Card	Nvidia GeForce GTX
GPU	4 GB
Processor	Intel®Core™ ii7-7700HQ (7th Gen)
CPU	2.80 GHz

3.1.2 HP Probook 6470b Specifications

Table 3.2: HP Probook 6470b Specifications

Type	Notebook
SSD	None
HDD	500 GB
RAM	4 GB
Graphic Card	None
GPU	None
Processor	Intel®Core™ 5-3230M
CPU	2.60 GHz

Tables 3.1 and 3.2 explain in detail the two systems that were used. They explain the amount of random access memory (RAM) used in both laptops, what processers both systems have, whether there is a graphic card or not, the GPU and CPU, as well as the amount of storage.

3.2 Software, and simulation tools used

A software is a program that assists in the administration, plan, coding, testing, assessment, or services of other projects. Globally available tools are categorized in the size and intricacy from simple software for beginner to intermediate level programmers and end users to extremely high-end and complex software that can support many software projects simultaneously and are mainly used in major businesses or organizations. Online programming support software- These software allow developers, programmers and clients to rapidly right and alter applications projects and test program results. Test information generators-These devices investigate a program and produce documents of information expected to test the rationale of the program. We did the work regarding our project across multiple operating systems and software applications such as Windows 10 and Ubuntu OS. The software we used are Sublime Text, and Android Studio. The languages we used in our project are Java and Python.

3.2.1 Windows 10

Windows 10 is a Microsoft based operating system mostly used in PCs and Laptop computers. It belongs to the Window NT family of operating systems. It is closed source which means that is must be licensed before it is fully installed and used on any PC or laptop. It was developed using three different programming languages, C, C++, and C# and it is of hybrid kernel type. Its default user interface is Windows Shell. It uses Windows API and the .Net Framework. The features of this operating system are mentioned below:

- User interface-oriented
- Hardware Independence
- Task Management
- File Management
- Disk Management
- Network Capability
- Hardware Adaptability
- Windows Defender Anti-virus
- Stock software (MS Office)
- Personal AI assistant
- Good for gamers

Our Windows operating system has the following specifications:

- Edition: Windows 10 Enterprise
- Version: 1903
- Type: 64-bit
- OS Build: 18362.418

3.2.2 Ubuntu OS

Ubuntu is a variant of Linux and is a Debian based operating system developed for PCs and Laptop computers. It belongs to the Linux based family of operating systems. In contrast to Windows, it is an open source operating system which doesn't need to be licensed before being installed and of any use. It is a popular operating system for cloud computing with support for OpenStack. Its default interface is GNOME and has a kernel type of Monolithic. Most individuals prefer Windows because it provides a highly attractive interface as compared to Linux and has more features such as gaming capabilities. In contrast to Windows, it is much more secure and safe from any bugs, viruses, etc. Overall, Linux is one of the best operating systems for hackers and programmers but is less preferred compared to Windows when it comes gamers. It is also more secure but has a less attractive interface.

3.2.3 Android Studio

Android Studio is a software tool developed by Google Inc. for their Android operating system. It was developed on JetBrain's IntelliJ IDEA software. Its specification is to develop full fledge Android applications. It is a free to use software with no license and hence no payment is required. It can be used on Windows, Linux and even MacOS. It was written in three different programming languages, Java, Kotlin, and C++. It requires a minimum 5 GB for installation and 8 GB RAM to run smoothly. It consists of the following specifications:

- Gradle based build support
- Android specific refactoring and quick fixes
- Lint tools to catch performance, usability, version compatibility and other platforms.
- Can create apps for all devices
- Pro Guard integration and app-signing capabilities
- Support for building Android Wear applications

- Android Virtual Device (Emulator) to run and debug apps in the Android studio.
- APK support
- Visual Layout Editor

3.2.4 Sublime Text

Sublime Text is a free to use source-code editor which is specifically built for the Python programming language. It locally supports many programming vernaculars and markup tongues, and limits can be incorporated by customers with modules, normally arrange created and kept up under free-programming licenses. It was written in two different programming languages, C++ and Python. It can be used in Windows, Linux and even MacOS.

3.2.5 Java

Java is a class based, object-arranged programming language structured with scarcely any execution conditions as could be. Java applications byte code (class record) can run on any Java virtual machine (VM) autonomously of models. It is moveable in light of the fact that PC programs written in Java language must run correspondingly on any equipment/working framework stage. Java is generally utilized for PCs, workstations, gaming supports, vehicle route frameworks, restorative observing gadgets, leaving meters, lottery terminals and cell phones. It is likewise utilized for server farms to store and move Web-based information. Five initial goals in the information of the Java language are as follows:

- Easy, operated and familiar
- Robust and portable
- Architecture-neutral and secure
- Excellent performance
- Elucidated, threaded, and spirited
- Works well with other languages
- Commonly used in Android app development
- Can be exported to other software tools like Android Studio

3.2.6 Python

Python is a deciphered, significant level, broadly useful computer programming language. Python’s structure reasoning accentuates code meaningfulness with its eminent utilization of significant white space. Its language builds and article arranged methodology used to assist developers in writing clear and intelligent code for little as well as enormous scale projects. Python is dynamically typed and trash gathered. It underpins numerous programming ideal models, including procedural, object-situated, and useful programming. It is often and widely described as a “batteries included” language due to its comprehensive standard library. Python interpreters are used in several operating systems, Windows, Linux, and MacOS in particular. A global community of programmers develops and maintains CPython, an open source reference implementation. The Python programming language is widely used in the field of Artificial Intelligence (AI) and Machine Learning. It is also partially used in Android App Development for developing Android apps for Android smartphones and tablets, and it is also compatible with other common and widely used programming languages like Java as well as C++ The most common software tools for implementing the Python programming language are Atom, Sublime Text, as well as PyCharm. The common features of Python are mentioned below:

- Cross-platform language
- Free and open source
- Object Oriented
- Large stand library
- GUI programming support
- Can be integrated with other languages
- Interpreted language
- Extensible
- Easy to learn and use
- Understandable and readable
- Portability
- High-level language
- Embeddable
- Dynamically typed
- Compatible with other languages

3.3 Summary

This chapter discussed the tools that were used in the project. The hardware used included two sets of systems, Acer Nitro V and HP Probook 6479b. The operating systems involved in the project were Windows OS, a closed-source operating system developed by Microsoft, written in C, C++ and C# and belong to the Windows NT family of operating systems while the other operating system that was involved is Ubuntu OS, a Debian based operating system and part of the Linux family of computers with no license and therefore is a free to use operating system. The software applications that were used for the completion of the project were Android Studio, an IDE developed by Google Inc for their Android operating system and specifically developed for Android app development. Android Studio is a software written in Java, Kotlin, as well as C++. It requires a minimum storage of 5 GB for complete installation. The other software used for the completion of the project was Sublime Text, a restrictive cross-stage source code supervisor with a Python application programming interface (API). The languages used in duration of the project were Java and Python.

Chapter 4

METHODOLOGIES

There were several methods used in getting the desired results for the project. The first thing was to collect data and once data was collected, it was loaded into text files. Once data had been collected and loaded, it was compiled in Python using Random Forest models, Support Vector Machine (SVM), as well as Logistic Regression in order to get the desired results.

4.1 Design of the investigation/Algorithms/Hardware

The first step was data collection. Data was collected in the form of plain text and text files. Once this data was collected, it was loaded into one text file with labels. After the data was loaded, it was compiled using Sublime Text in Python to get the desired results. An Android based application was then developed in Android studio using Java Programming Language.

4.2 Other Methodologies

PDCA (plan–do–check–act or plan–do–check–adjust) comprises of a four-step strategy utilized in business for the control and nonstop improvement of procedures and items.

4.2.1 The Plan, Do, Check, Act Rule

Plan: In accordance with the expected output the objectives and process can be established to deliver the result. For the targeted improvement establishment of output expectations is the completeness and accuracy of the specification.

Do: First implement the plan then execute the process and at last developing the product.

Check: Study the actual result and compare with the expected result.

Act: Correction of the difference between the actual data with the outcome results.

4.2.2 The Plan, Do, Check, Act Rule for this project

Plan: Getting familiar with Python, Java, Android Studio and Sublime Text, extracting the SMS for text mining purposes and designing the app.

Do: Using the use of real time data. The data provided by the company for the model development

Check: Checking the model for accuracy, precision, sensitivity, and specificity.

Act: Dependent of the result then act.

4.3 Summary

This chapter explains the methodologies used to achieve the desired results of this project. The first thing was to collect data and once data was collected, it was loaded into text files. Once data had been collected and loaded, it was compiled in Python using Random Forest, Support Vector Machine (SVM), and Logistic Regression in order to get the desired results. Another methodology used was the PDCA rule that is Plan, Do, Check, and then Act.

Chapter 5

SYSTEM TESTING

System Testing may involve checking your system validity. We can apply different kinds of checks on our app for the assurance of quality performance. Some of testing which are used in our system are given below.

- Objective testing
- Usability Testing
- Software Performance Testing
- Compatibility Testing
- Load Testing
- Security Testing
- Installation Testing

5.1 Objective testing

In this we have test our collected data again and again by implementing different type of machine learning algorithms. We then implemented two models, Support Vector Machine as well as Random-Forest model to check the accuracy and precision of our model. Once we finished text classification using these models, we performed logistic regression on our data. After we trained our model we gave our model test data for checking our model whether it is giving our desire output or not. Our model do testing on our data and show results on screen. Due to this we could know that our model is working correctly and is fulfilling. The significant objectives of Software testing are as per the following: Gaining trust in and giving data about the degree of value. An objective test is a test that has right or wrong answers therefore can be checked Fair-mind. Target tests 48 are well known on the grounds that they are anything but difficult to get ready and take fast to check, and give a quantifiable and solid outcome. Software testing has different goals and objectives. We then ran our app in Android Emulator to test its capabilities and to check whether it was fully functional or not.

5.2 Usability Testing

This testing is carried out to measure how much the system will be used by users. Our app can be used by different Android smartphone users. But in future using this model an android app can be built so that every user could use it. For the time being due to lack of technology there is no proper SMS spam detection app that everybody can use. Usability testing, a non-useful testing strategy which is a proportion of how much the framework can be used by individuals. Instances of items that regularly advantage from convenience testing are nourishment, purchaser items, sites or web applications, PC interfaces, reports, and gadgets. Ease of use testing estimates the ease of use, or usability, of a particular object or set of articles, though broad human–PC collaboration thinks about endeavor to plan all-inclusive standards. Convenience testing centers around estimating a human-made item's ability to meet its planned murmur directing a conventional lab ease of use.

5.2.1 Natural Mapping

We applied natural mapping in our Graphical User Interface so that users can understand what the system is giving them.

5.2.2 Task Analysis

We used different task analysis to break down each and every task of our app into different steps so the user does not get annoyed or irritated and understands how to use the app.

5.3 Software Performance Testing

After repeatedly checking and testing we got 100% performance rate from our app. It cannot be hanged while doing operations and gives accurate results. There are no bugs as of now in the app. Our app's performance is very good. It is reliable and very fast. Our app doesn't crash even if user give big data. It is trained on data in just few minutes and also test data in no time. Execution testing is the method to decide efficiency, responsiveness and dependability of a computer, organize, programming system or electronic device under a remaining task at hand. Execution testing can include quantitative tests done in a lab, or happen in the generation condition in predetermined number of spots. These tests can help improve the application sooner rather than later.

5.4 Compatibility Testing

Compatibility testing is performed to examine the compatibility of the software on different platforms and operating systems. Example: The software can run on any of the latest Android versions starting off with Android Nougat.

Test cases no	Operating System	Different test Support	Compatible
1	Windows	No	No
2	Android	Yes	Yes
3	Mac	No	No
3	Linux	No	No

Table 5.1: This table explains the platforms the app will be compatible with

5.5 Load Testing

Load testing is carried out to know the conduct of the framework under the specific expected load. Example: The software is tested in load of 20 users accessing resources at the same time; the system response time was good.

5.6 Security Testing

Security testing is carried out to disclose the weaknesses in the software. We can apply different security checks in our app to provide our users with a secure platform where they can easily perform their task.

5.7 Installation Testing

Installation testing is carried out to install the software. The importance of installation testing - in software development life cycle (SDLC) -can be very important. However, the significance of this process is not limited to limited aspects only, and it can be further elaborated as follows:

5.7.1 Activity Testing

We can test each activity of our app. It starts from registration and ends when a user blocks any detected SMS spam on his Android smartphone. The user enters the text that appears in an incoming SMS message to check whether that content is a spam or not and if it is, he has the option to block it from his Android smartphone.

5.7.2 Operation Testing

We can test each function of our app properly. App operations are as follows:

- Scanning device for SMS spams.
- Blocking SMS spams.

5.8 Test Cases

A test case is a great deal of conditions or factors under which an analyzer will choose if a system under test satisfies necessities or works adequately. The path toward making analyses can in like manner help find issues in the necessities or plan of an application. The component for deciding if a product program or framework has finished or flopped such an assessment is known as a test prophet. In certain settings, a prophet could be a prerequisite or use.

Test Case #1: User Login

Software:

Test Description:

A student can login via its valid email. He enters his email and then presses the login button after the verification the system will allow to use it.

Testing Environment:

Test ID:

Preconditions	And Android phone and an internet connection is required
Actions	User enters a valid Email Address and password
Expected Results	User will successfully login
Result	User will successfully login

Test Case #2: User Profile

Software:

Test Description:

A user can make his complete profile by clicking ion the registration button.

Testing Environment:

Test ID:

Preconditions	An Android Phone and an internet connection is required.
Actions	User enters his details.
Expected Results	Checking the validation of his Email.
Result	User can successfully register

Test Case #3 : Searching for Spams

Software:

Test Description:

A student can search his device for any spams.

Testing Environment:

Test ID:

Preconditions	User has a verified profile.
Actions	User can scan his device for any spams.
Expected Results	User is notified of spams stored on his device.
Result	User is notified of spams stored on his device

Test Case #4: Block SMS Spams

Software:

Test Description:

A user can block any SMS spams that are detected on his Android device

Testing Environment:

Test ID:

Preconditions	User has a verified profile and the app should detect spams stored on user's Android device.
Actions	User scans device for any spams.
Expected Results	User can block detected spams.
Result	User can bock detected spams.

5.9 Specific Requirements

This subsection consists of the complete functionalities for the system, use cases, functional and non-functional requirements, external interface requirements, inverse interfaces as well as design constraints.

5.9.1 Functionalities

This sub-section contains the requirements for the SMS Spamming Application. It also includes what the system will do and what it won’t do. This section will also mention the features and characteristics for the proposed system. Functionalities refer to the total or part of an item, for example, a product application or registering gadget, can accomplish for a client. An item's usefulness is utilized by advertisers to distinguish item includes and empowers a client to have a lot of abilities. Usefulness could possibly be anything but difficult to utilize. If a product does not the meet the functionality requirements of a user, then it is considered a fail otherwise it is considered as a success.

Table 5.6: Searching for Spams

Introduction	
Input	The user clicks the search button to search for spams
Processing	1. First the user goes to first tab in the Main activity, and click Search for Spams 2. Once user reaches search page, user clicks the Search button 3. Once user has clicked the search, starts to detect Spams 4. Once the system has detect spams, the use case terminates.
Output	Spamming App displays SMS spam messages stored in phone
Error Handling	No SMS detected Display error message

Table 5.6 explains what the user will do when he searches for spams. The user will first enter text in the app. Once the text has been entered, the app will detect if the text entered of the certain SMS is a spam or not. If it is a spam, the app will then notify the user and the user will be provided with the ability to delete that spam. However, if the entered text is not a spam, the user will be notified by the app that the text is not a spam.

Table 5.7: Identifying Spams

Introduction	
Input	The user begins to identify spams available on the device
Processing	1. First, the user identifies SMS spam 2. Once the user has identified a spam, he or she chooses to either save SMS or block it 3. Once the user has blocked the spam, the system blocks and deletes the spam 4. Once the system has deleted the spam, the use case terminates.
Output	Spamming App starts detecting spams
Error Handling	No SMS detected Display error message

Table 5.7 explains what will happen when the app starts detecting if the text is a spam or not. If the text is detected as a spam, the user will be able to delete the spam

Table 5.8: Detection of Spams

Introduction	
Input	System starts detecting spams once search is complete
Processing	1. First the user chooses to delete spam 2. Once the user has chosen to delete the spam, App then sends spam to spam filter 3. Once the app has sent the spam to the spam filter, the use case terminates.
Output	Spamming App displays spams to the user
Error Handling	No SMS detected Display error message

Table 5.8 explains what will happen when the detection is complete. If the scanned is text is labeled as a spam, then it will be deleted

Table 5.9: Giving user option to block spam

Introduction	
Input	The system gives user the option to block detected spams
Processing	1. First the user chooses to block detected spam 2. Once the user has chosen to block the detected spams, App then blocks the selected spams 3. Once the app has blocked the selected spams, the use case terminates.
Output	User receives message to block spams
Error Handling	No SMS detected Display error message

Table 5.9 explains what will happen when the text has been labeled as a spam. The user will be able to delete the SMS message.

Table 5.10: Blocking Selected Spams

Introduction	
Input	User chooses to block selected spams
Processing	1. First the user chooses to block the detected spam 2. Once the user has blocked the selected spams, App then blocks the selected spams 3. Once the App has blocked the selected spams, the use case terminates.
Output	Selected spams are sent to the system
Error Handling	No SMS detected Display error message

Table 5.10 explains what will happen when the user is given the option of deleting the spam message. He can choose to delete the SMS message or keep it.

Table 5.11: Deleting chosen spams

Introduction	
Input	User blocks selected spams
Processing	1. First the user chooses to block the selected spams 2. Once the user has chosen to block the selected spams, App then blocks the selected spams 3. Once the app has blocked the spams, the use case terminates.
Output	Spamming App deletes spams selected by the user
Error Handling	No SMS detected Display error message

Table 5.11 explains what will happen to the spams that the user will delete. Those spams will be deleted

5.9.2 External Interface Requirements

External interface prerequisites determine equipment, programming, or database components with which a framework or segment should interface.

5.9.2.1 User Interfaces

The GUI is the most important part of the system as it is used to interact with the users of the system. The first thing which should be considered while designing a user interface is that a GUI must be user friendly. The user interface for SMS Spamming Application shall be Xml, Python, and Java. A first-time client of the smartphone app should see the primary page when he/she opens the application. On the off chance that the client is definitely not a first-time client, he/she ought to have the option to see the pursuit page straightforwardly when the application is opened. Each client ought to have a profile page where they can alter their telephone number.

5.9.2.2 Hardware Interfaces

As the app is Android based and only runs on Android smartphones, all users must have a working Android smartphone with an active phone number. The Android version should be 4.4 or above. If a user does not have an Android version of 4.4 or above on his or her Android device, he or she will be unable to run the application. The app is not very heavy and will download and install instantly depending on the strength of the user’s Internet connection. Once the app is installed, the user will receive weekly notifications on future updates.

5.9.3 Software Interfaces

1) The SMS Spamming application will search for any SMS spam messages in phone’s database.
2) The SMS spamming application will detect spams and give the user with the option to either save the SMS or block it.
3) The SMS spamming application will then send the spam to the spam filter
4) The SMS spamming application will notify the user of incoming anonymous SMS messages.

5.9.4 Communications Interfaces

The SMS Spamming application shall communicate through user’s current phone number. There is a need for any internet connection for the app to run on the Android smartphone. And once the user has full access to the app, he can delete any kind of spam through the app by just entering text from a given SMS spam and verify if the text is a malicious spam or not.

5.9.5 Functional Requirements

A down to earth essential describes a component of a system or its fragment, where a limit is depicted as a detail of lead among yields and information sources. Utilitarian essentials may incorporate tallies, particular nuances, data control and getting ready, and other unequivocal helpfulness that describe what a structure ought to accomplish.

5.9.5.1 Functional Requirement 1

Download mobile application: A client ought to have the option to download the portable application through either an application store or comparative assistance on the cell phone. The application ought to be allowed to download. The app will not be very heavy and should download instantly depending on the strength of the user’s Internet connection. Once the app has been downloaded and installed, the user should have the ability to detect and block spams from his or device.

5.9.5.2 Functional Requirement 2

Download and notify users of new releases: At the point when another/refreshed form or arrival of the product is discharged, the client should check for these physically. The download of the new discharge ought to be done through the cell phone similarly as downloading the versatile application. The user will also receive notifications from the app and will be given the option to update the app to a newer version if he or she wants to get the latest features of the application.

5.9.5.3 Functional Requirement 3

User registration - Mobile application: Given that a client has downloaded the portable application, at that point the client ought to have the option to enroll through the versatile application. The client must give name and phone number to complete his or her registration. Once the user has completed his or her registration, he or she will be directed to the login page where the user will have to enter his or her full name and current phone number to access app's features.

5.9.5.4 Functional Requirement 4

User log-in - Mobile application: Given that a client has enlisted, at that point the client ought to have the option to sign in to the portable application. The sign in data will be put away on the telephone and later on the client ought to be signed in consequently. After the user has logged in, he or she will automatically be directed to the main page where they can search for any spams located on his or her device. The user will also be given the option to edit his or her profile when he or she sees fit.

5.9.5.5 Functional Requirement 5

Retrieve password: Given that a client has enlisted, at that point the client ought to have the option to recover his/her password by email. He/she should also be able to search for any SMS spams on his or her Android smartphone. The user can also retrieve his or her account by providing his or her email or phone number. Once the user provides his or her email or phone number, he or she will receive verification code which he can enter to retrieve his or her account.

5.9.5.6 Functional Requirement 6

Mobile application – Search: Given that a client is signed in to the application, at that point the main page that is demonstrated ought to be the pursuit page. The client ought to have the option to scan for a spam. Once the search begins, the system will start identifying and detecting spams which the user can later block and delete if he or she sees fit.

5.9.5.7 Functional Requirement 7

Spam Blocking: Once the user has detected any spams, he or she will be given the option to store it in the phone's database or to block and delete it. If the user chooses to block the spams that he or she has selected, the app will then instantly delete the spam and send it to the spam filter.

5.10 Non-Functional Requirements

The Non-functional requirement for the system incorporates following quality traits that effect run-time behavior, system design, and Player experience, performance, security, portability, maintainability, reliability as well as availability. They specify criteria that can be used to condemn the movement of a structure, rather than unequivocal practices. They are showed up contrastingly in connection to utilitarian essentials that describe express lead or limits.

5.10.1 Performance

All parts of the system are performing great and effectively. The apparent reaction is prompt. The framework doesn't squander important assets. It does not take initial load time more than 10 to 12 seconds

5.10.2 Reliability

The system can play out its planned capacities and activities in a framework's situation without encountering any disappointment, aside from Operating System blunder. All mistakes are taken care of in a smooth way. The software is a very reliable, one because there is no chance of system failure as there are only limited user which can access the system. The system is 90% reliable.

5.10.3 Availability

As our application is an open source application once, it will connect to the server then it will available for everyone at any time. Once the software is installed, it is always available unless and until the software would uninstall.

METER: Measurements acquired from 1000 hours of use during testing.

MUST: More than 98% of the time.

PLAN: More than 99% of the time.

WISH: 100% of the time.

5.10.4 Security

The app should be able to keep any user’s Android smartphone safe and secure from spams whether they are known spams or unknown spams. It is a secure application and won’t cause any damage to the user’s Android smartphone. And the user will be notified of any regular updates.

5.10.5 Maintainability

The system can be revived and changed if there ought to be an event of any defects. It has ability to change the structure parts to meet new helpfulness. It can without quite a bit of a stretch alter new features and customization. All upgrades can be fundamentally and safely performed. The application should be anything other than hard to grow. The code should be composed with the end goal that it favors utilization of new limits.

5.10.6 Portability

The system currently runs on Android Operating Systems. It is portable, as it has ability to reuse features and source code to any other Operating System as well. The application should be portable with Android.

5.11 Inverse Requirements

1. The app should detect Spams.
2. The system should only detect trained datasets.
3. The system should detect non-PDF formats.

5.12 Use Cases

A use case at its easiest is a representation of a client's collaboration with the system that shows the interaction between the individual and the distinctive use cases in which the individual is included. It can also condense the subtleties of your system's clients (otherwise called on-screen characters) and their associations with the framework. To manufacture one, you'll utilize a lot of particular images and connectors. It demonstrates the relationship between all the actors involved in a Software Project. The entertainer can be a human or other external structure. In systems structuring, use cases are used at a more raised level than inside programming building, every now and again addressing missions or accomplice targets. The point by point requirements may then be trapped in the Systems Modeling Language (SysML) or as legitimate explanations. Use Cases have the following features:

- Organizing practical necessities.
- Modeling the objectives of framework client communications.
- Recording situations from trigger occasions or extreme objectives.
- Describing the essential course of activities and extraordinary progression of occasions.
- Permitting a client to get to the usefulness of another occasion.

5.12.1 Use Case #1

Table 5.12: Use Case #1

ID	1
Description	To Search for an SMS Spam
Actors	User
Preconditions	An Android phone and an active phone is number is required.
Basic Steps	1. First the user goes to first tab in the Main activity, and clicks search for spams 2. Once the user reaches the search page, user clicks the Search Button 3. Once the user clicks the search button, the system starts to detect spams 4. Once the system finishes spam detection, the use case terminates.
Alternate Steps	None
Exceptions	User's Android device has registered phone number
Post-conditions	Once the user has registered with his or her phone number, he or she should be able to detect spams on the phone.

5.12.2 Use Case #2

Table 5.13: Use Case #2

ID	2
Description	To identify and look for any Spams on the device
Actors	System
Preconditions	An Android phone and an active phone is number is required.
Basic Steps	1. First the user searches for spams by clicking the search button 2. Once the search button is clicked, the system starts to search for spams 3. Once the search is complete, the system starts detecting spams 4. Once detection is complete the use case terminates.
Alternate Steps	None
Exceptions	User's Android device has registered phone number
Post-conditions	Once the user has registered with his or her phone number, he or she should be able to detect spams on the phone.

5.12.3 Use Case #3

Table 5.14: Use Case #3

ID	3
Description	Once search is complete, system detects spams on the device
Actors	System
Preconditions	An Android phone and an active phone is number is required.
Basic Steps	1. First the user searches for spams by clicking the search button 2. Once the search is complete, the system then starts detecting spams. 3. Once detection is complete, the system returns results in the form of the spams that were detected. 4. Use case terminates.
Alternate Steps	None
Exceptions	User's Android device has registered phone number
Post-conditions	Once the user has registered with his or her phone number, he or she should be able to detect spams on the phone.

5.12.4 Use Case #4

Table 5.15: Use Case #4

ID	3
Description	Once a spam is detected, give user the option to block it
Actors	System
Preconditions	An Android phone and an active phone is number is required.
Basic Steps	1. First the user chooses to block the spam from his or her device 2. Once the user has chosen to block the selected spams, the system then blocks the spams from harming the device 3. Once the system has blocked the selected spams, the use case terminates.
Alternate Steps	None
Exceptions	User's Android device has registered phone number
Post-conditions	Once the user has registered with his or her phone number, he or she should be able to detect spams on the phone.

5.12.5 Use Case #5

Table 5.16: Use Case #5

ID	3
Description	Once the system has detected a spam, user blocks it
Actors	User
Preconditions	An Android phone and an active phone is number is required.
Basic Steps	1. First the user chooses to block the spam from his or her device 2. Once the user has chosen to block the selected spams, the system then blocks the spams from harming the device 3. Once the system has blocked the selected spams, the use case terminates.
Alternate Steps	None
Exceptions	User's Android device has registered phone number
Post-conditions	Once the user has registered with his or her phone number, he or she should be able to detect spams on the phone.

5.12.6 Use Case #6

Table 5.17: Use Case #6

ID	6
Description	Once user chooses to delete the spam, system blocks it
Actors	System
Preconditions	An Android phone and an active phone is number is required.
Basic Steps	1. First the user chooses to block spam from his or her device 2. Once the user has chosen to block the selected spam, the system then blocks the spam from harming the device 3. Once the system blocks the spam, it then sends the blocked spam to spam filter 4. Once the system deletes the spam and sends it to the spam filter, the use case terminates.
Alternate Steps	None
Exceptions	User's Android device has registered phone number
Post-conditions	Once the user has registered with his or her phone number, he or she should be able to detect spams on the phone.

Use Case Diagram

A use case diagram at its easiest presents a user's connection with the system which shows the interaction between the person and the distinctive use cases in which the person is involved.

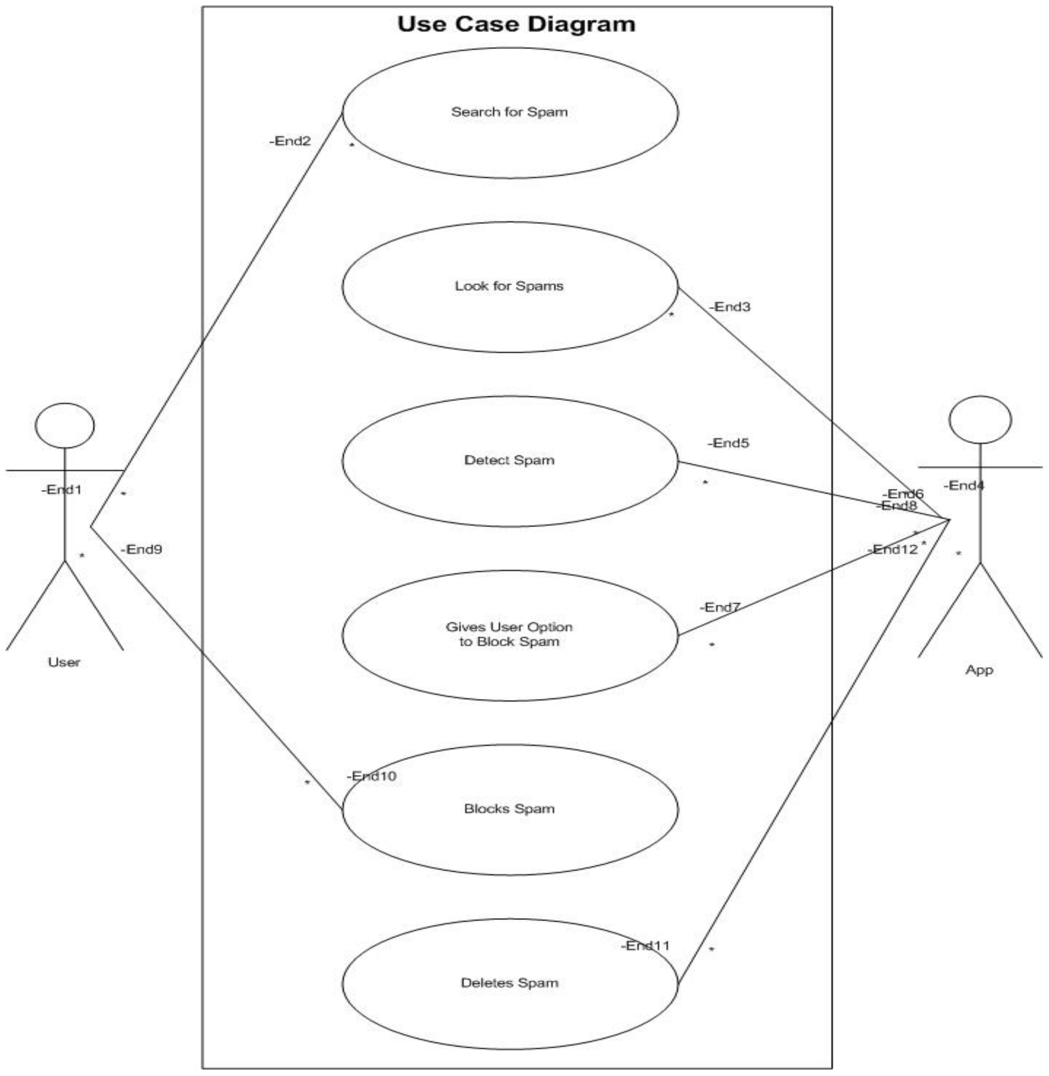


Figure 2: Use Case Diagram

In Figure 2, there are two actors, one is the user and the other is the app. In the first use case, the user clicks the search button to search for spams. Once the person begins the search, the app begins to look for any spams on the device. Once the search is complete, the app then starts to detect spams. After detecting spams, the app gives the user the option to delete the spams from his or her device. If the user chooses to block the selected spams, the app then deletes those spams from the device.

Activity Diagram

An activity diagram outwardly introduces a sequence of activities. Movement outlines are frequently used in business process demonstrating. These will be able to depict the purpose of a case graph. Exercises presented are successive and be done at the same time.

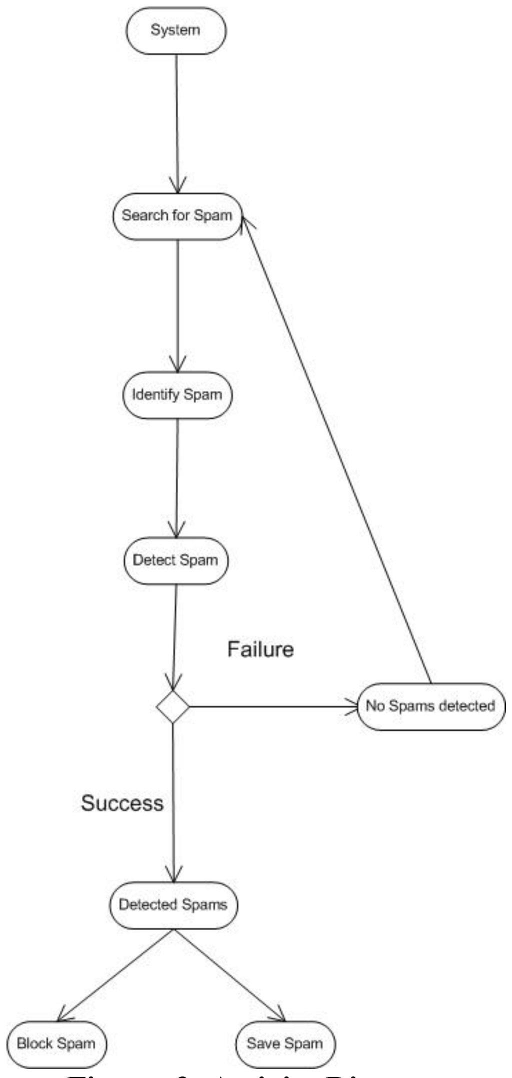


Figure 3: Activity Diagram

In Figure 3, we have seven action states. In the first one, the user clicks the search button to search for spams. Once the search begins, the system starts to identify spams. After identification, the system begins spam detection and when the detection process is finished, it returns the results. If there are no spams detected, the user will be redirected to the search page otherwise he or she will be able to see a list of spams available on the device. The user will then have the option to store the spam or delete it.

Component Diagram

A component diagram clarifies the association just as the composition of the physical segments, in a framework. These outlines ordinarily help model usage subtleties and watch that each part of the framework's necessary capacities is secured by arranged advancement.

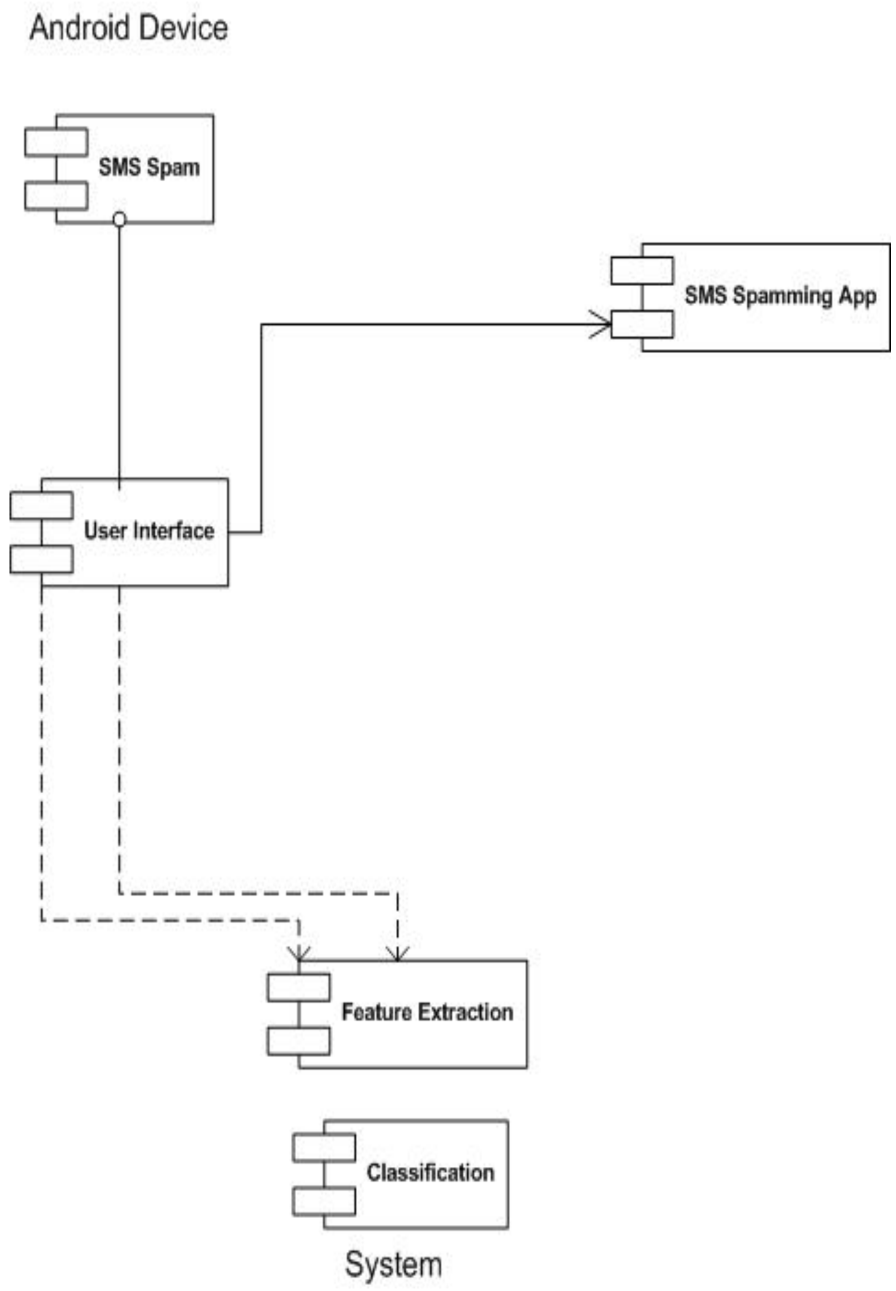


Figure 4: Component Diagram

In Figure 4, there are 5 components, the first one is the SMS spam located on the device. The second is the GUI which is what the client can see on his or her smartphone. The third is the SMS Spamming App which is the software installed on the user's device. The fourth and fifth are feature extraction and classification that are functions performed by the software.

Deployment Diagram

Deployment diagrams have a few significant applications.

- Show which programming components are conveyed by which equipment components.
- Illustrate the runtime preparing for equipment.
- Provide a perspective on the equipment framework's topology.

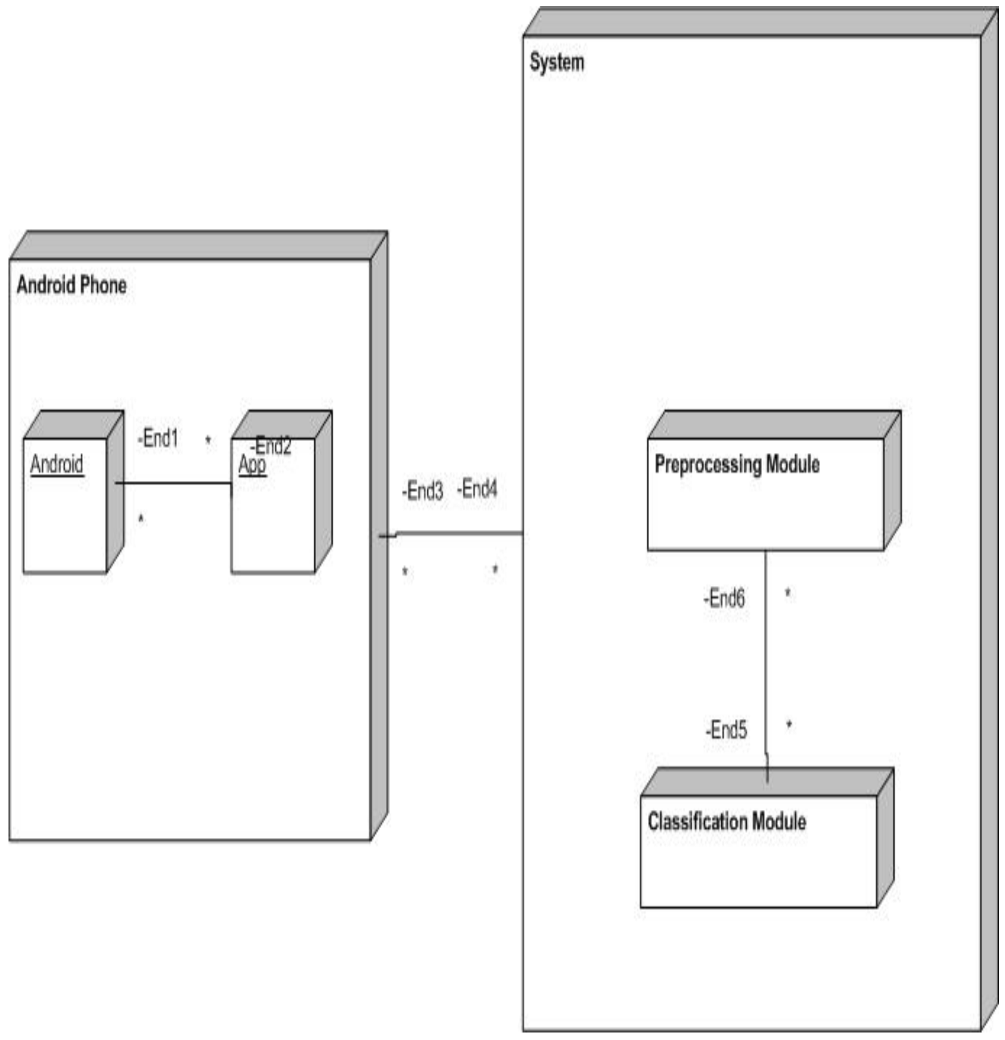


Figure 5: Deployment Diagram

In Figure 5, there are two main nodes which are the Android Phone and the system. In the Android phone, we have two sub-nodes called the Android operating system and the app while in the System, there are two modules called the pre-processing module and the classification module.

<u>Data Flow Diagram</u>

A DFD presents the way information can be dealt with by a framework as far as data sources and yields. As its name demonstrates its attention is on the progression of data, where information originates from, where it goes and how it gets put away.

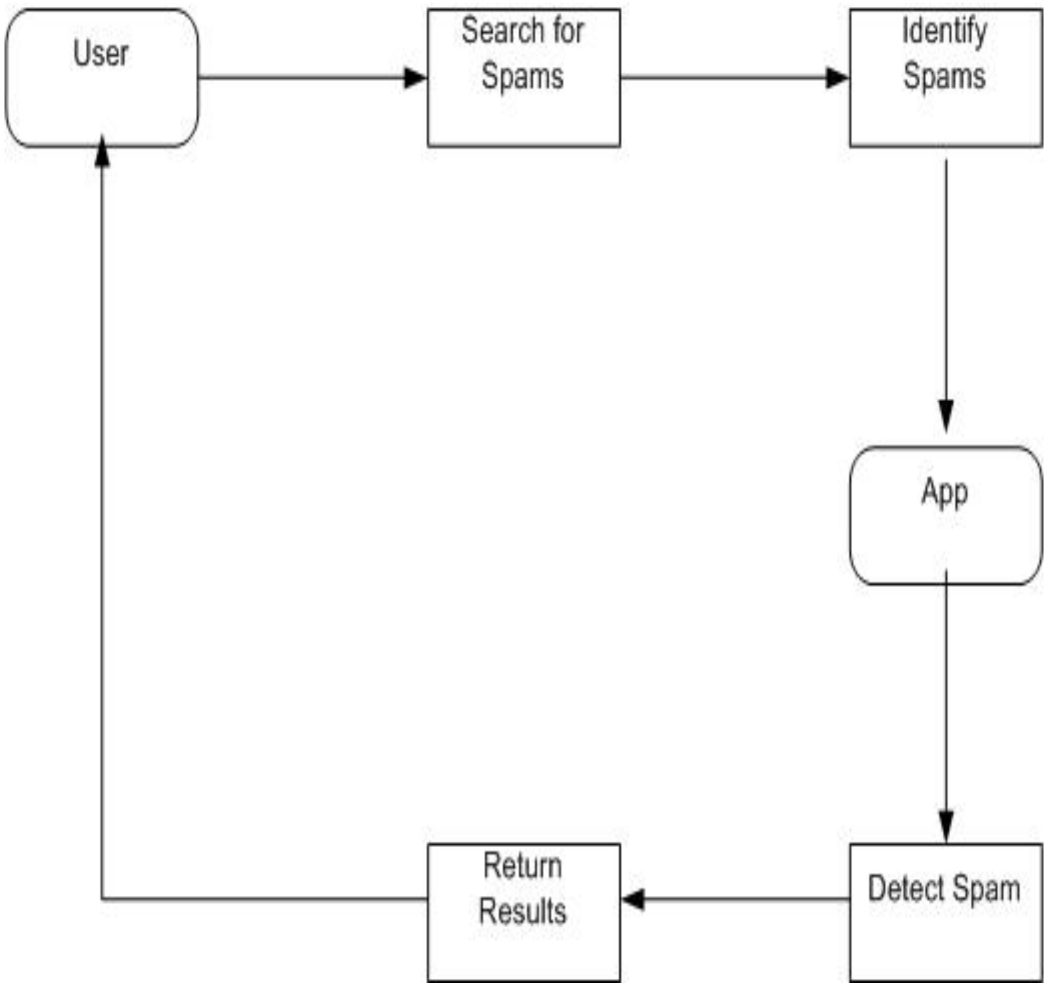


Figure 6: Data Flow Diagram

Figure 6 illustrates the flow of data for the system. The user clicks the search button to search for spams. The structure then starts to identify spams. Once the identification process is finished, the system detects spams available on the device. After spam detection is complete, the system returns the results to the user. Once the user successfully deduces whether the SMS is a spam or not, he is able to delete it from his device.

State Transition Diagram

State machine diagram is a conduct outline which shows discrete conduct of a piece of structured framework through limited state advances

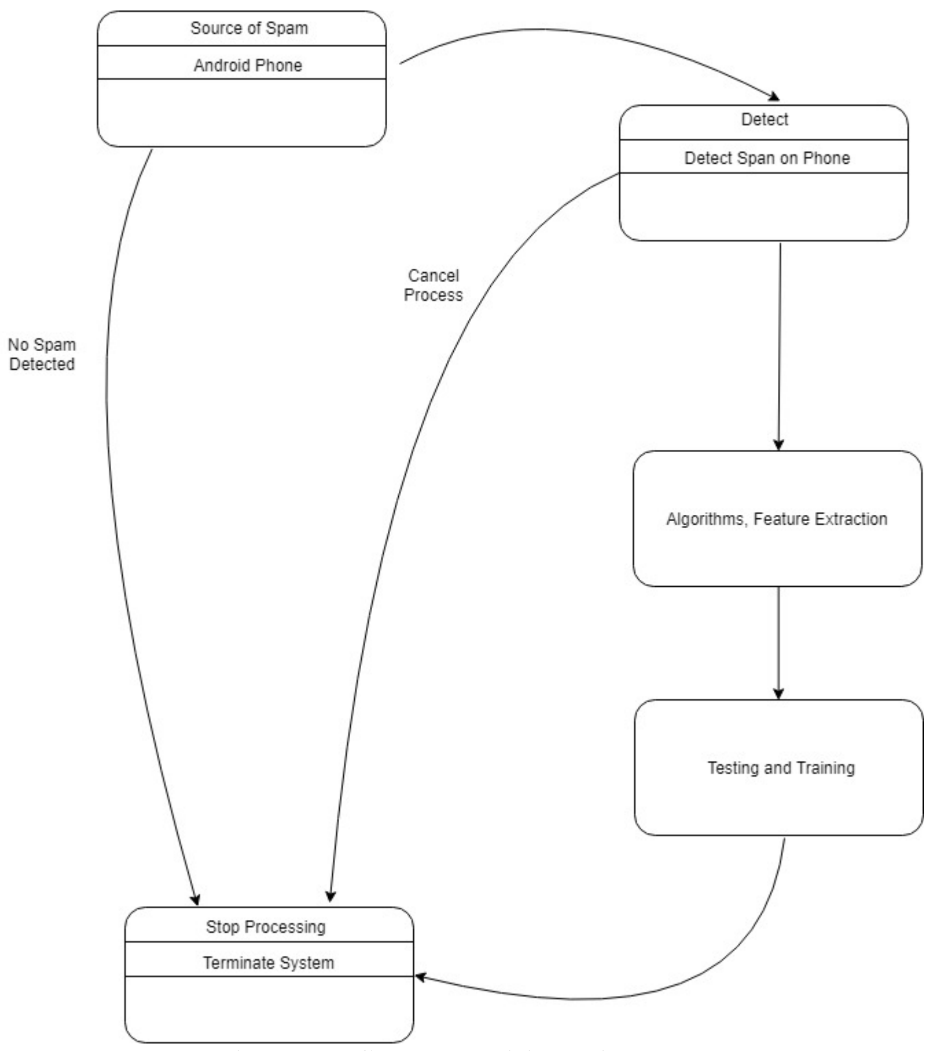


Figure 7: State transition Diagram

Figure 7 illustrates the states that the system goes through. The first is the source of the spam which is the Android phone. The next one illustrates the detection process which will detect spams located on the device. The third one shows the feature extraction and algorithms that will be used. The fourth one shows the training and testing of data sets. And the last state illustrates that the process must be stopped or the system should be terminated if there is no spam detected or if the user manually stops the process

Sequence Diagrams

A Sequence Diagram (SD) refers to a connection diagram which presents the way each item connects to each other as well as to what request. It is a build of a message succession graph. In this, we identify how the functions are carried out in the system. So this can help in making the different functionalities in the application. We carried out sequence diagram in each use case.

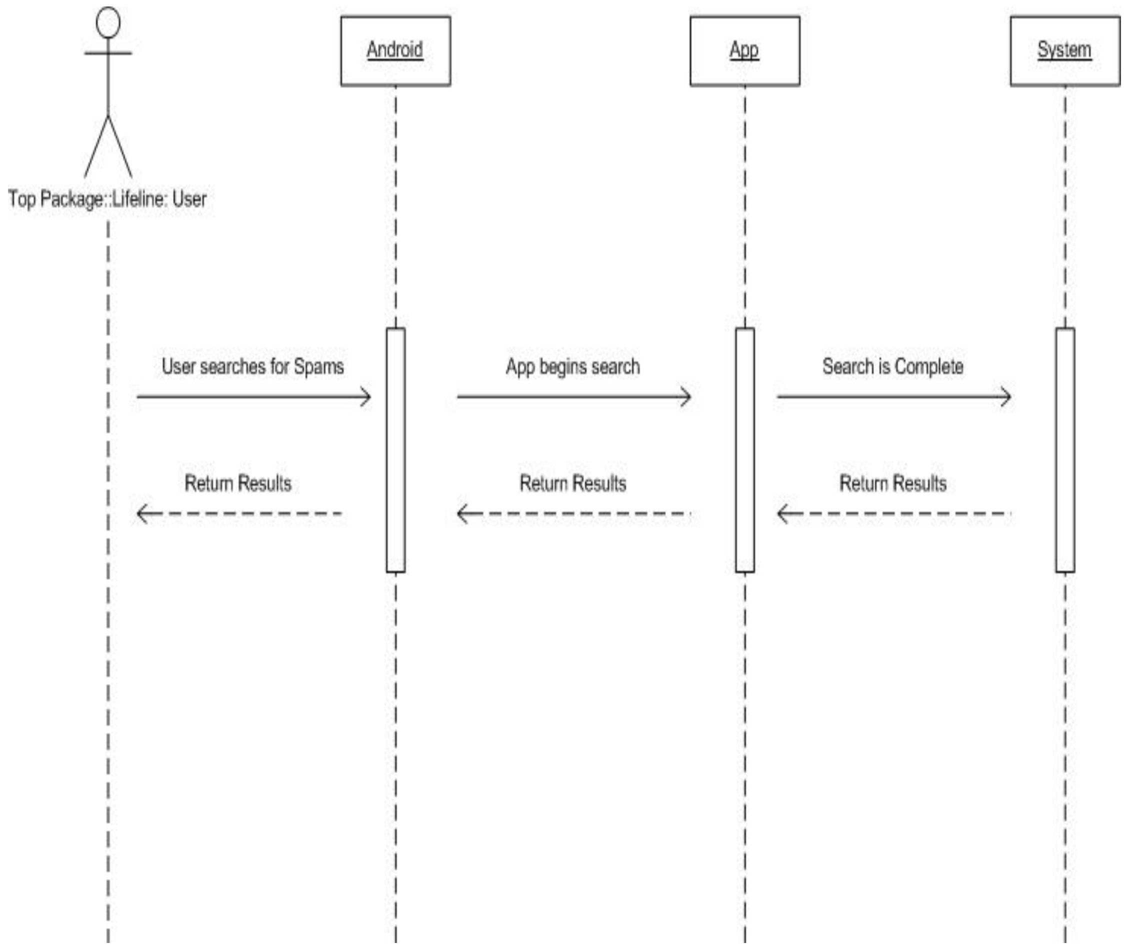


Figure 8: Sequence Diagram #1

In Figure 8, there is an actor which is the user. The user clicks the search button to search for spams. The app then begins the search. Once the search is complete, the app will then return the results to the user in the form of detected spams on the Android device.

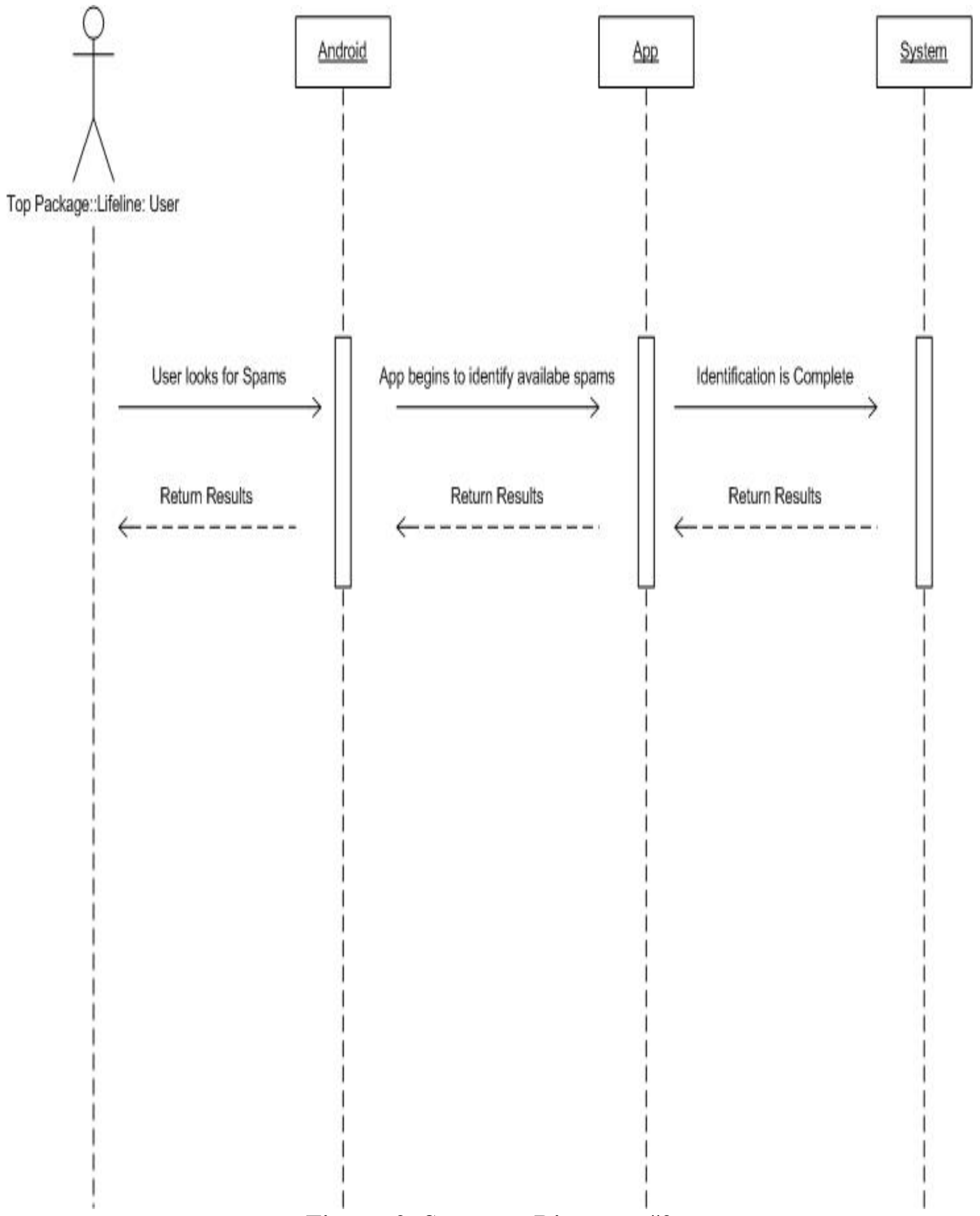

Top Package::Lifeline: User
Android
App
System
User looks for Spams
App begins to identify availabe spams
Identification is Complete
Return Results
Return Results
Return Results

Figure 9: Sequence Diagram #2

In Figure 9, there is an actor which is the user. Once the search is complete, the user has the app identify SMS. The app then starts to identify the spams. Once identification is complete, the app will then return the results in the form of detected spams on the Android device.

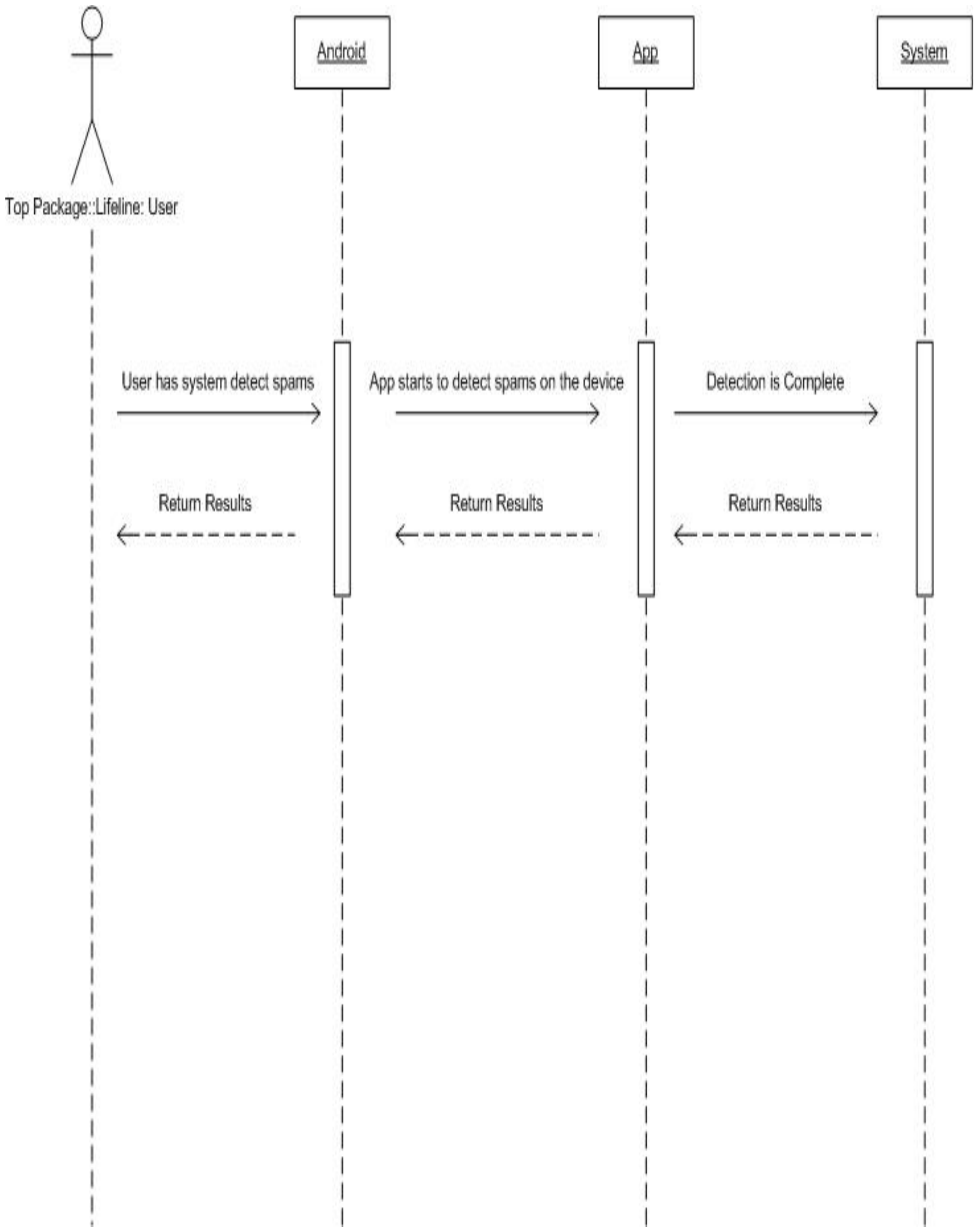

Top Package::Lifeline: User
Android
App
System
User has system detect spams
App starts to detect spams on the device
Detection is Complete
Return Results
Return Results
Return Results

Figure 10: Sequence Diagram #3

In Figure 10, there is an actor which is the user. Once identification is complete, the user will have the app detect spams located on the device. The app then begins spam detection. Once detection is complete, it will then return the results to the user

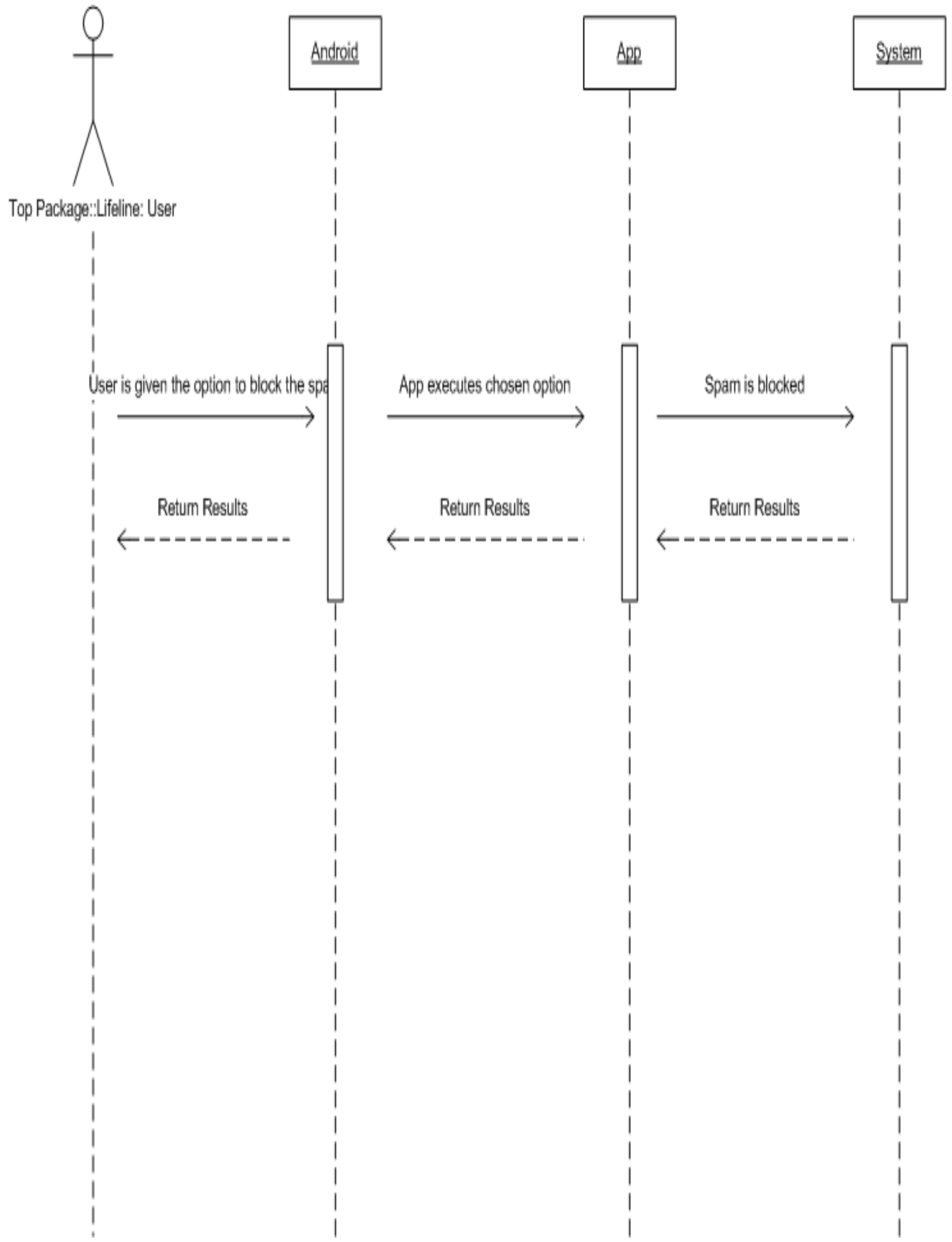

Android
App
System
Top Package::Lifeline: User
User is given the option to block the spa
App executes chosen option
Spam is blocked
Return Results
Return Results
Return Results

Figure 11: Sequence Diagram #4

In Figure 11, there is an actor which is the user. Once the app has detected spams located on the Android device, it will give the user the option to delete the spam. If the user chooses to delete the spam, the app will then block the spams that are selected by the user.

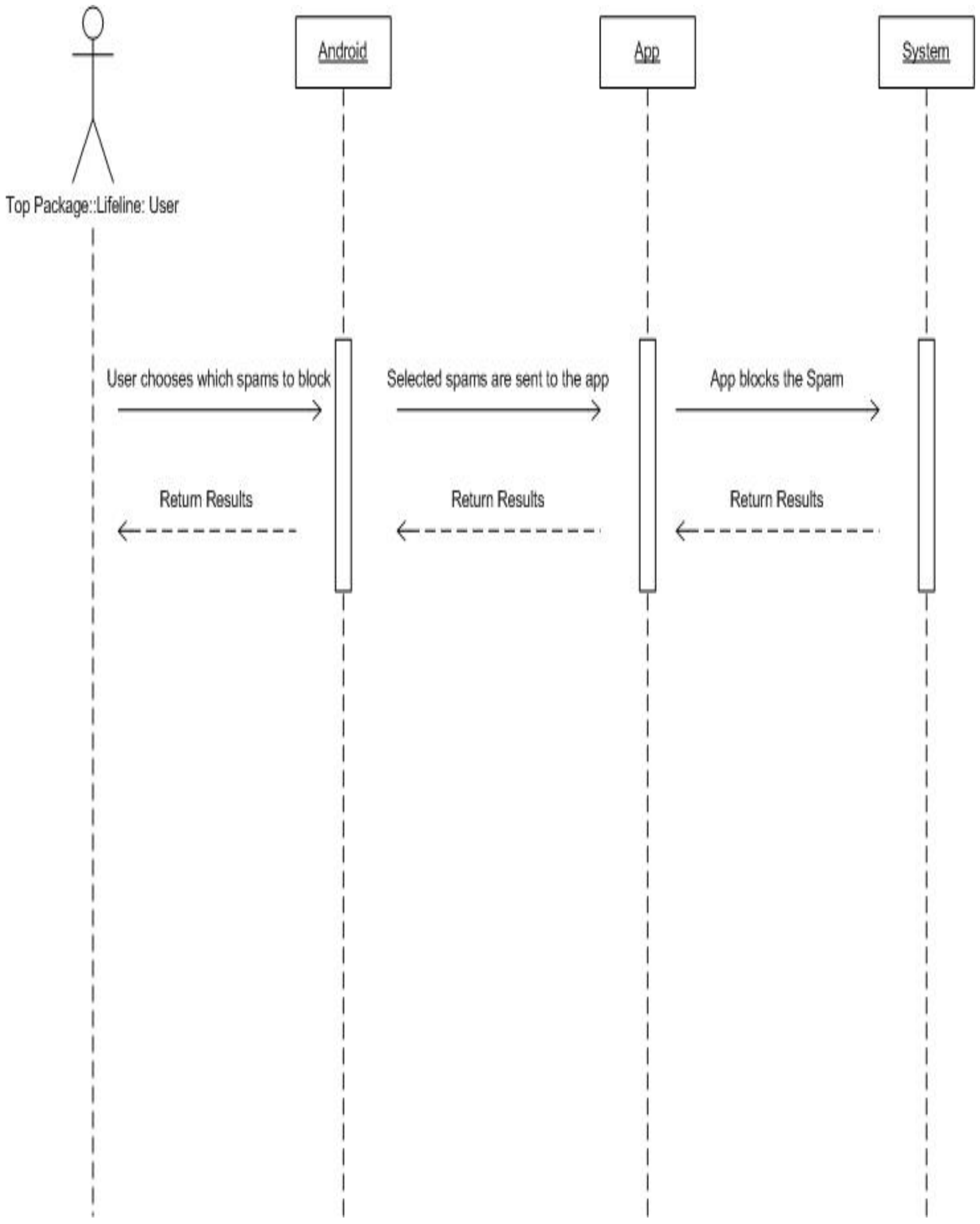

Android
App
System
Top Package::Lifeline: User
User chooses which spams to block
Selected spams are sent to the app
App blocks the Spam
Return Results
Return Results
Return Results

Figure 12: Sequence Diagram #5

In Figure 12, there is an actor which is the user. Once the user is given the option to store or delete a spam, he or she chooses to block it. The app then recognizes the selected spams and then deletes them from the device.

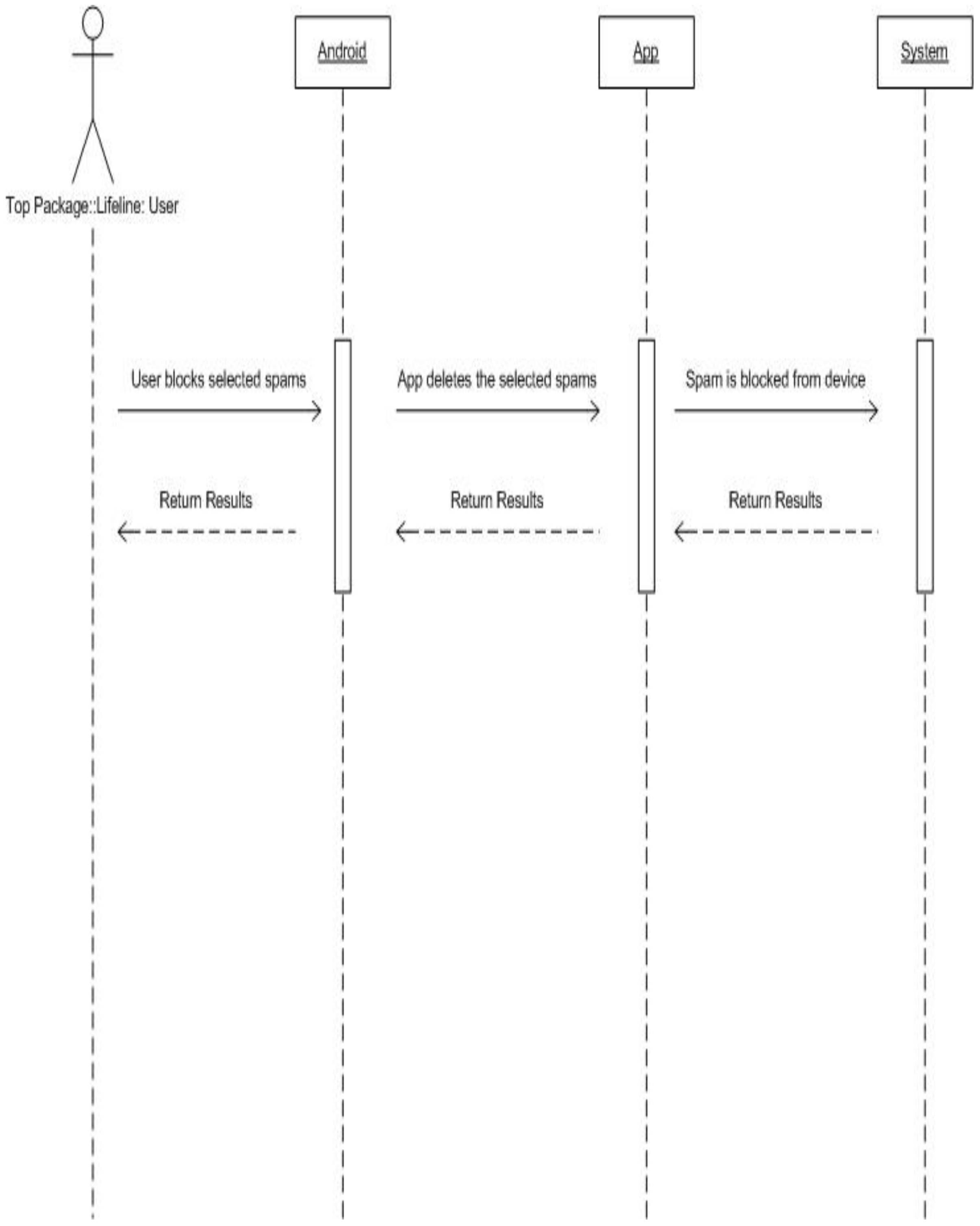

Android
App
System
Top Package::Lifeline: User
User blocks selected spams
App deletes the selected spams
Spam is blocked from device
Return Results
Return Results
Return Results

Figure 13: Sequence Diagram #6

In Figure 13, there is an actor which is the user. Once the user is given the option to store or delete a spam, he or she chooses to block it. The app then recognizes the selected spams and then deletes them from the device

5.13 Summary

This chapter discussed the system testing of our application. The system testing phase of this software project includes objective testing, software performance testing, compatibility testing, usability testing, load testing, activity testing, security testing, installation testing, and operation testing. Also discussed in this chapter are the functional and non-functional requirements of our app as well as all the functionalities, external interface requirements, test cases and use cases of this project. And lastly this chapter provides a brief overview of all the diagrams used to build this project. The diagrams we used to specify the software requirements of our app include the use case diagram, activity diagram, component diagram, deployment diagram, data-flow diagram, state-transition diagram as well as sequence diagrams.

Chapter 6

RESULTS AND CONCLUSION

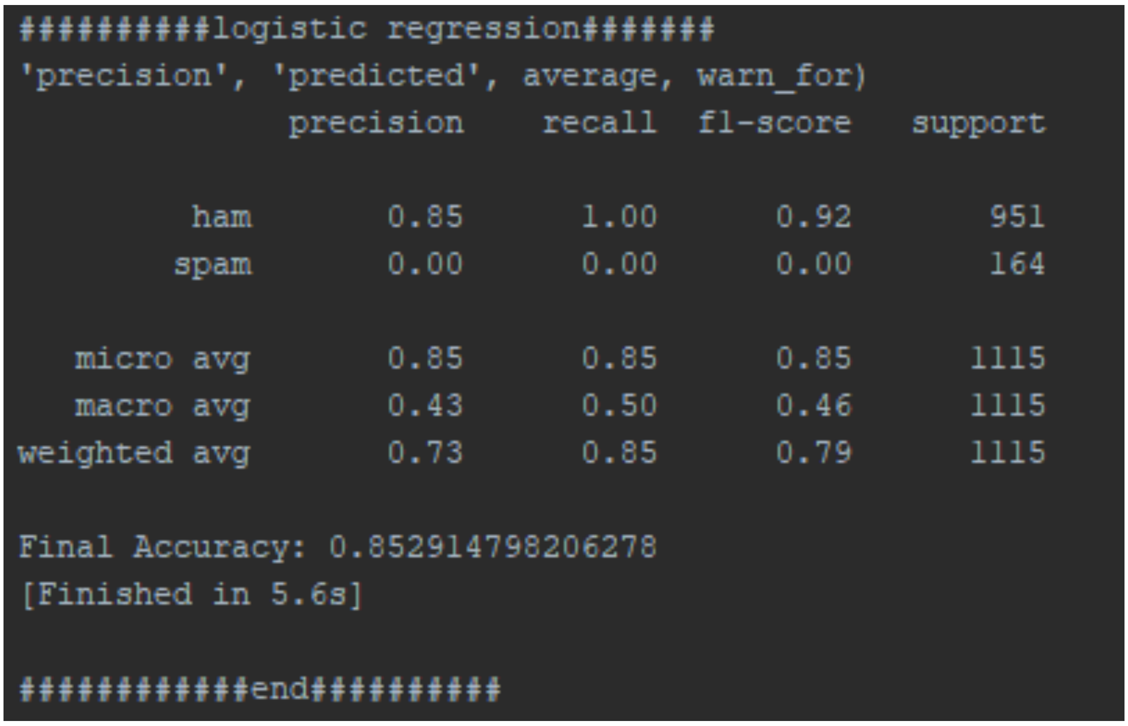
After completing the project, we were able to achieve the desired results. We have broken the results of our project into different sub headings below

6.1 Presentation of the findings

Android smartphone users can download the app from the PlayStore, and once installed, the user will be able to detect any spams that are incoming or currently stored on his device and will be able to block them. After developing and training our model, we performed logistic regression, implemented Random Forest and Naïve Baise and got the following results:

6.1.1 Logistic Regression

This image presents the results we got after training our model and performing logistic regression on both spam and ham datasets. It also shows the accuracy we found of our dataset.

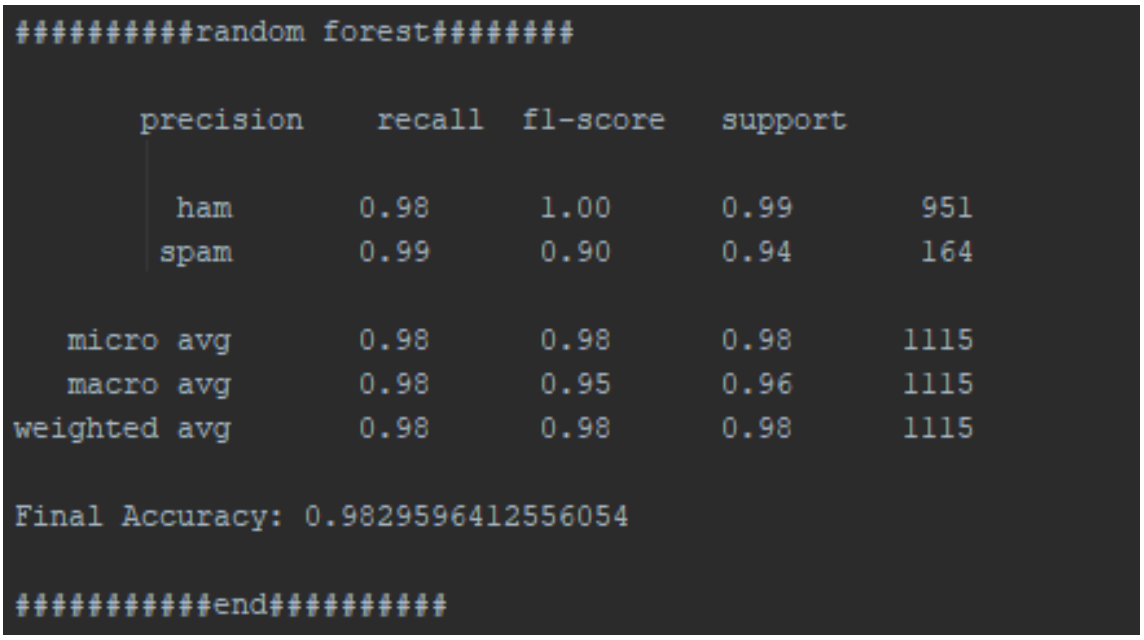


```
##########logistic regression#######
'precision', 'predicted', average, warn_for)
              precision    recall  f1-score   support

         ham       0.85      1.00      0.92       951
        spam       0.00      0.00      0.00       164

   micro avg       0.85      0.85      0.85      1115
   macro avg       0.43      0.50      0.46      1115
weighted avg       0.73      0.85      0.79      1115

Final Accuracy: 0.852914798206278
[Finished in 5.6s]

############end##########
```

Figure 14: Logistic Regression

6.1.2 Random Forest

This image presents the results we found after implementing the Random-Forest classifier on our data set.

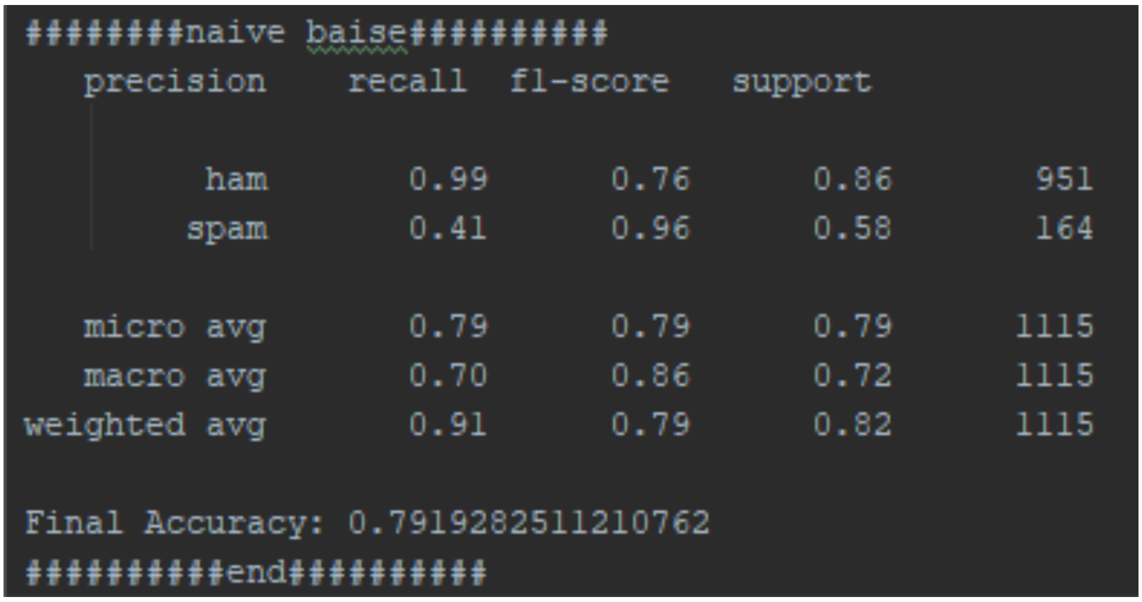


```
##########random forest########

       precision    recall  f1-score   support

         ham       0.98      1.00      0.99       951
        spam       0.99      0.90      0.94       164

   micro avg       0.98      0.98      0.98      1115
   macro avg       0.98      0.95      0.96      1115
weighted avg       0.98      0.98      0.98      1115

Final Accuracy: 0.9829596412556054

###########end##########
```

Figure 15: Random Forest Classification

6.1.3 Naïve Baise

This image explains the results we found after performing Naïve Baise classification on our data set. It also shows the accuracy we got of the data.



```
########naive baise##########
   precision    recall  f1-score   support

         ham       0.99      0.76      0.86       951
        spam       0.41      0.96      0.58       164

   micro avg       0.79      0.79      0.79      1115
   macro avg       0.70      0.86      0.72      1115
weighted avg       0.91      0.79      0.82      1115

Final Accuracy: 0.7919282511210762
##########end##########
```

Figure 16: Naïve Baise Classification

6.1.4 GUI

Once our work in Python was complete and after we did classification of our datasets using Random Forest, Logistic Regression, and Naïve Baise we created a new project in Android Studio and started coding in Java to develop our app's graphical user interface.

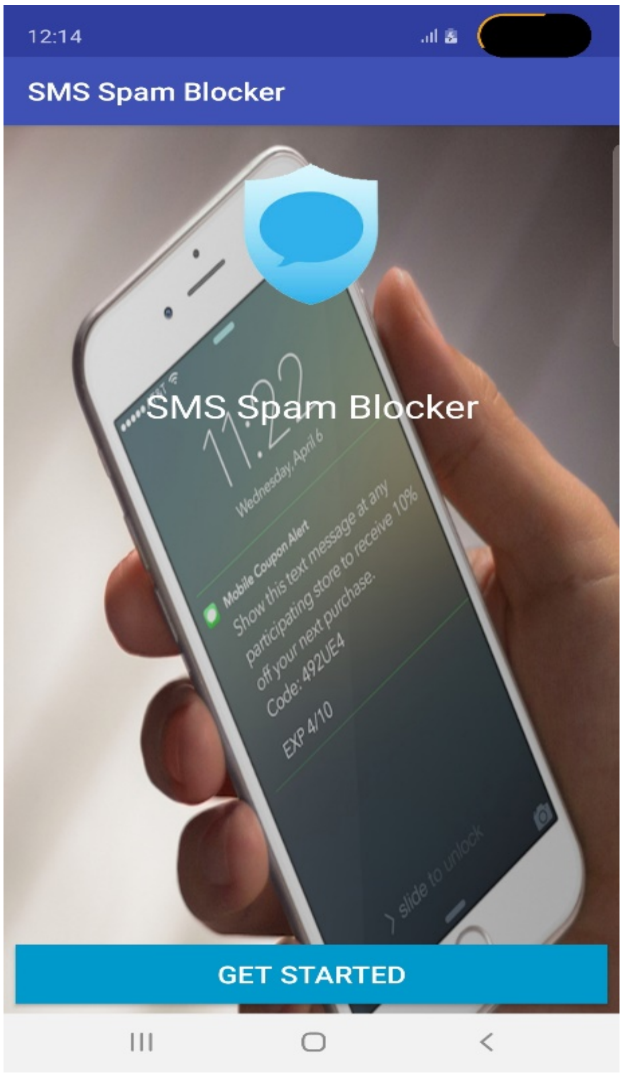


Figure 17: Main Screen

Figure 17 displays the main screen. It is the first thing the user will see once he opens the app on his Android smartphone. He will have the option to either quit the app or continue forward to the login and registration menu. At the bottom of the screen is a big blue button labelled "Get Started". This button tells the user to continue to the next activity if he wishes to continue.

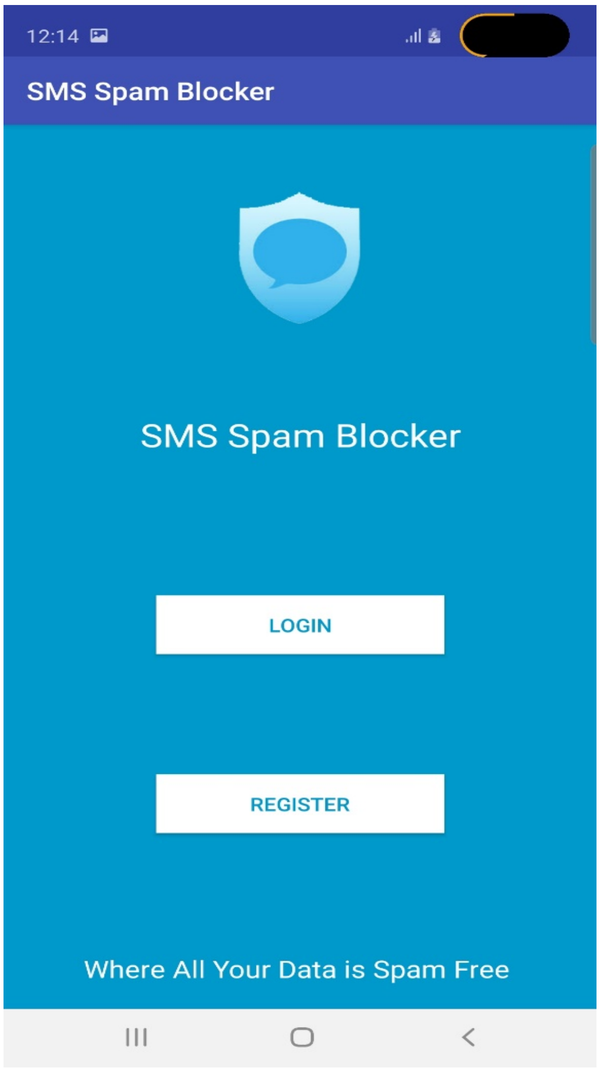


Figure 18: Login and Registration

Figure 18 shows the login and registration menu. On this activity, the user will have two options. The user can login with his phone number and password if he has already created his account. Once the user is logged in, he will have access to all his information such as contact numbers, messages, etc. The user can then scan to see if he has received any spam messages from anyone not included in his contacts list. The app will scan the text of the SMS to check if there is any malicious data in the SMS message. If the user has not created an account, he will have to register to get full access to the app.

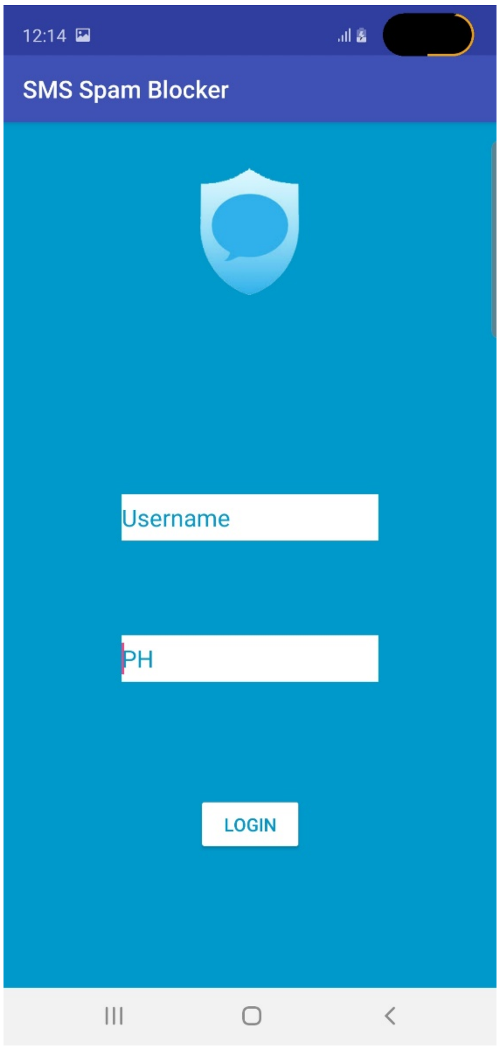


Figure 19: Login

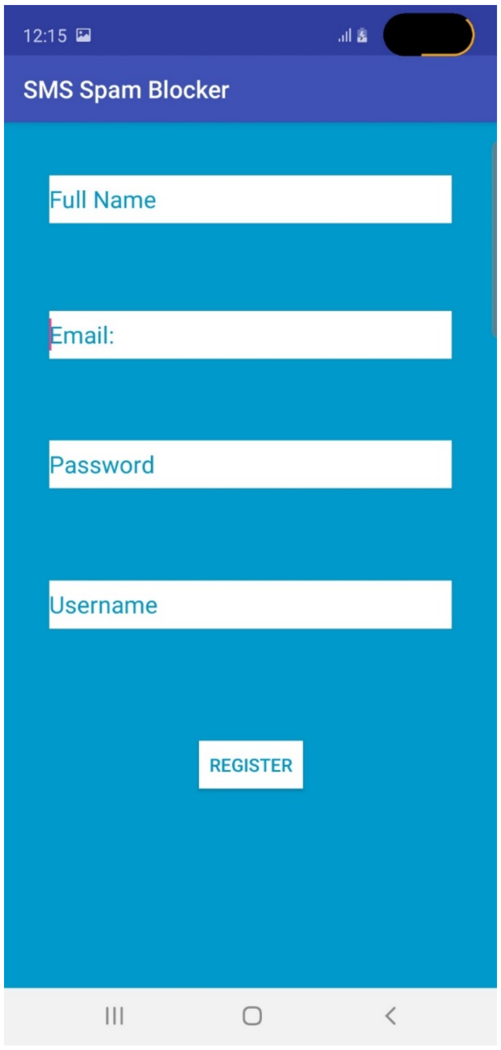


Figure 20: Registration

Figure 19 and 20 show the login and registration screens. In the login screen, the user can login with his username and phone number if he has already created his account. Once the user is logged in, he will have access to all his information such as contact numbers, messages, etc. The user can then scan to see if he has received any spam messages from anyone not included in his contacts list. The app will scan the text of the SMS to check if there is any malicious data in the SMS message. If the user has not created an account, he will have to go to the registration screen so he can register to get full access to the app. A user can register by entering his full name, email, password and username. If a username is already taken, the user will receive a message to enter a different username. If an entered email is already registered, the user will be required to enter a different email.

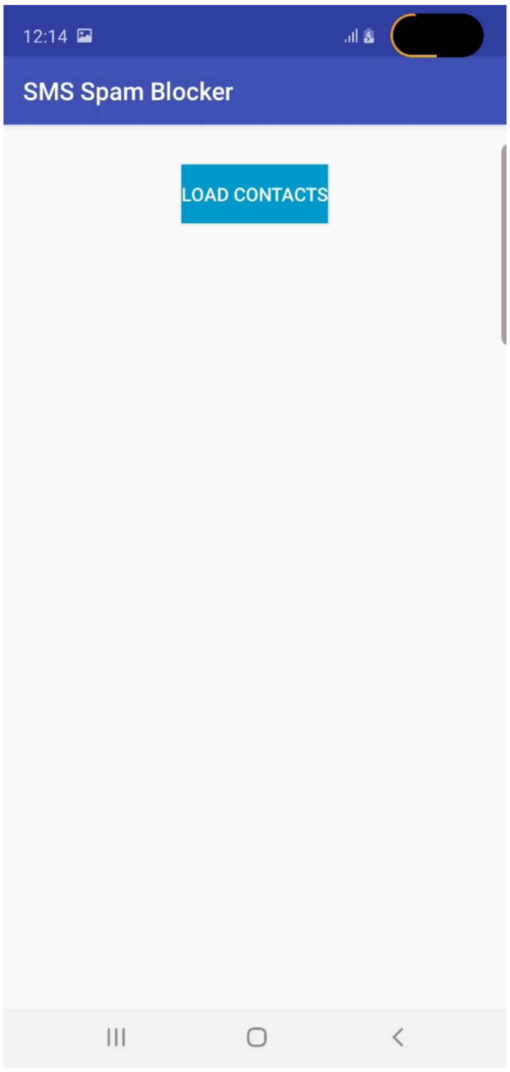


Figure 21: Load Contacts

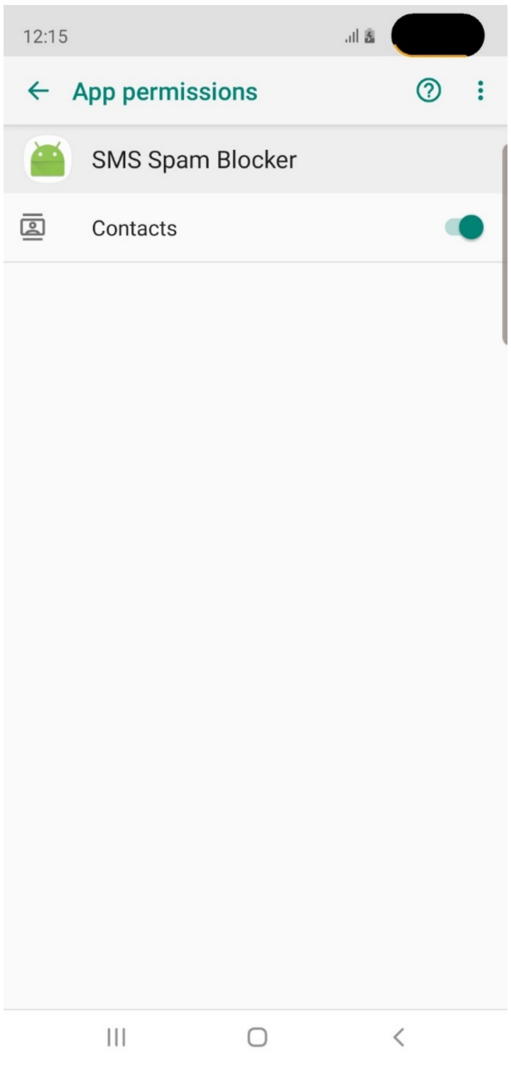
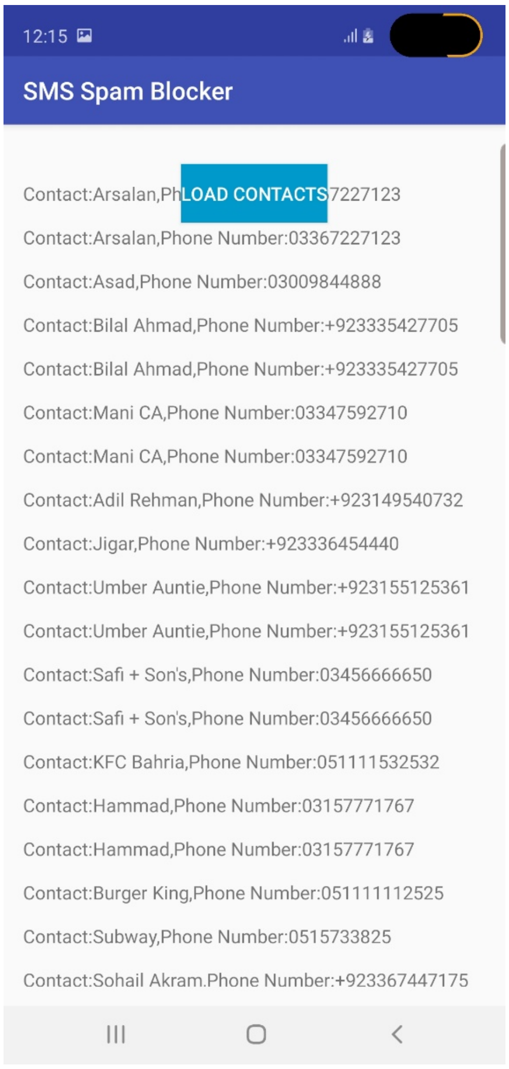


Figure 22: Permissions

Figure 21 shows the load contacts option. Once a user has created an account and logged in, he will have the option to load all his contacts and messages. Once the all the messages and contacts have been loaded, the user will have the option to scan all the messages to determine if they are any spams on the device or not. If there are, the user will have the option to block those spams. Figure 22 shows the permissions screen. Before performing any operations, the app will need permission from the device to access the messages and contacts that are stored on the Android smartphone. In order to get full access to all the features, the user will first have to let his device grant the app permission for full access to the messages and contacts.



Contact:Arsalan,Ph 7227123
Contact:Arsalan,Phone Number:03367227123
Contact:Asad,Phone Number:03009844888
Contact:Bilal Ahmad,Phone Number:+923335427705
Contact:Bilal Ahmad,Phone Number:+923335427705
Contact:Mani CA,Phone Number:03347592710
Contact:Mani CA,Phone Number:03347592710
Contact:Adil Rehman,Phone Number:+923149540732
Contact:Jigar,Phone Number:+923336454440
Contact:Umber Auntie,Phone Number:+923155125361
Contact:Umber Auntie,Phone Number:+923155125361
Contact:Safi + Son's,Phone Number:03456666650
Contact:Safi + Son's,Phone Number:03456666650
Contact:KFC Bahria,Phone Number:051111532532
Contact:Hammad,Phone Number:03157771767
Contact:Hammad,Phone Number:03157771767
Contact:Burger King,Phone Number:051111112525
Contact:Subway,Phone Number:0515733825
Contact:Sohail Akram.Phone Number:+923367447175

Figure 23: Contacts

Figure 23 shows the contacts screen. Once a user has logged in, granted permissions to the app and loaded all his contacts, he will see all his stored contacts and SMS messages. The user will then have the choice to scan all the data for any spam messages that are stored on his device. If the app detects any spam messages, the user will have the choice to block those spam messages and send them to the spam filter.

6.2 Discussion of the findings

Once completing our project, we compared it to other findings or work done by others. After doing a detailed analysis in comparison to other findings, we concluded that our findings were more accurate in context to the goals we wanted to achieve and based on the accuracy of the data we collected.

6.2.1 Comparison with initial GOAL

After doing a breakdown of the goals we wanted to achieve and the goals we achieved, we can say that we did quite achieve the desired results and outcomes of our projects though a few improvements can be made in the near future.

6.2.2 Reasoning for short comings

There were some obstacles we had to face in duration of the project which nearly got in the way of us reaching our desired outcome such as Hardware failure, employee unavailability, timing issues, software unavailability, etc. We do hope that in the future these short-comings do not affect future projects.

6.3 Limitations

The app does require a constant internet connection to function properly otherwise it won't be able to assist the user in detecting any spams on his Android smartphone. The user must have an active account if he wishes to access any of the app's features.

6.4 Recommendations

In case anyone else wants carry on my work in the future or make any modifications, my recommendations are to:

- Thoroughly go through the research before making any changes
- Collect your own set of data
- Perform comparative analysis with other authentic work
- Compare actual results with desired results
- Build your own interface from scratch

6.5 Summary

The result of this project is that the completed app will able to scan and detect SMS spams in any user's Android smartphone. Once detection phase is complete, it will notify the user of any spams stored on his device and the user will then be able to block those spams. This project is intended to make it easier for Android smartphone users to be able to delete spams from their smartphones.

6.6 Conclusion

SMS spams have become a very major issue in today's world of smartphone technology and after doing some research, we have concluded that there are ways to detect these spams and block them from any given smartphone device. We created an app specifically for Android smartphones that can assist Android smartphone users in scanning and detecting SMS spams and then block them. Once the app is complete, it will fully assist the user in being able to delete SMS spams from his device.

Chapter 7

FUTURE WORK

Our project was an Android app that we developed to assist in detecting spams. We first developed a model using spam dataset and performed Logistic Regression, Random Forest and Naïve Baise classification to get certain results. Using this app, Android users will be able to enter text from an SMS and the app will detect if it is a spam or not. In the future, we will be able to expand the scope of our work where the application will even allow a given user to detect Pop-up ads and incoming spam calls. Once the app successfully scans and detects these ads and spam calls, it will notify the user and then the user will be able to block them. To do this we will have to develop and train models using Machine Learning. One of our main objectives for the future is to make the app more user-friendly, and reduce bugs, if any. Regular updates will also be part of our future plans for our application and most importantly, we intend to make it as one of the best mobile apps for any kind of spam detection.

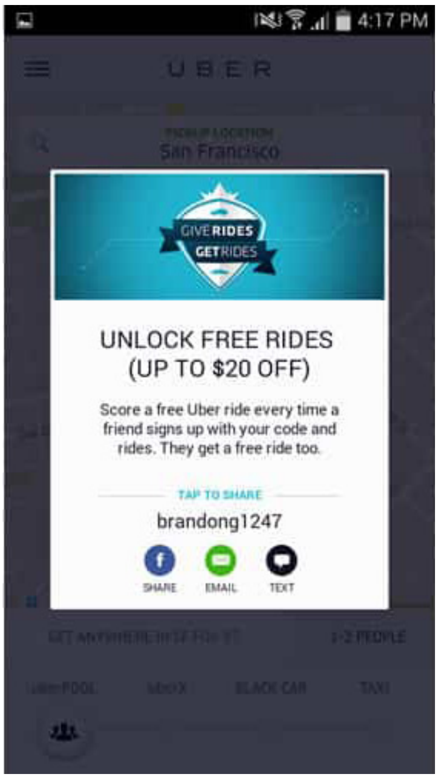


Figure 24: Pop-Up Ads

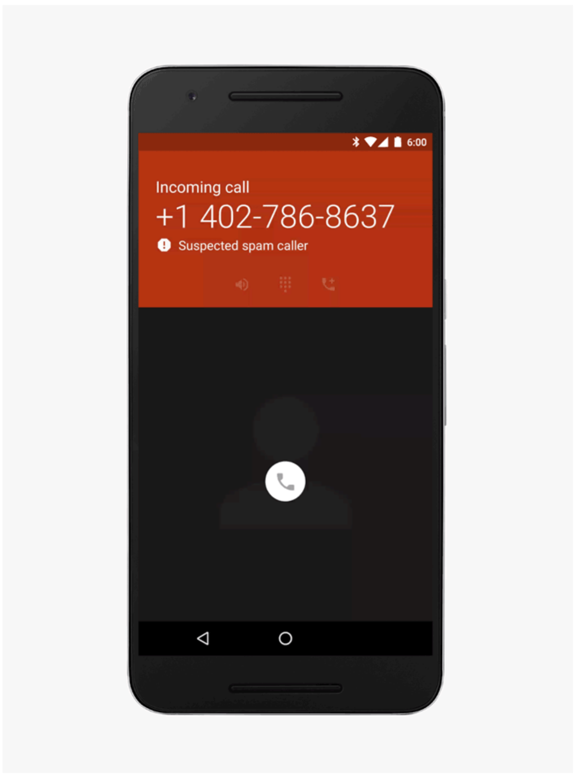


Figure 25: Spam Calls

Figures 24 and 25 display sample pop up ads and spam calls. The purpose of these images is to provide users insight of what add-ons and new features they will get in future updates

REFERENCES

[1]. Mohsin M. and Shahzad H. 2014, Last Card Android Game, BS Project Report, University of Lahore, Islamabad, Pakistan

[2]. Dipak K. and Kavita O., “Content based SMS Spam filtering using Machine Learning,” U.S. Patent 4 084 217, Nov. 4, 2018.

[3]. Quinqing R., Feature-Fusion Framework for Spam Filtering Based on SVM, CA: Department of Control Science and Engineering Zhejiang University Hangzhou, 310027, China.

[4].Hedia S. and Fatemah A., SMS Spam Filtering Using Machine Learning Techniques: A Survey, 1Dept. of Mathematics, Statistics and Computer Science, College of Science, University of Tehran, Tehran, Iran, 2016.

[5]. Dima S. and Ghazi N., “SMS Spam Detection using H2O Framework,” presented at the The 8th International Conference on Emerging Ubiquitous Systems and Pervasive Networks (EUSPN 2017), 2017.

[6]. Houshmand S. M., “SMS Spam Detection using Machine Learning Approach”

[7]. Muhammad R. I. and Morshed U. C, “SPAM FILTERING USING ML ALGORITHMS,” School of Information Technology, Deakin University, Victoria, Australia.

[8]. José M. G. H. and Guillermo C. B., “Content Based SMS Spam Filtering,” University of Madrid, January 2006.

[9]. Shafi’I M. A. and Muhammad Shafi A. L., “A Review On Mobile SMS Spam Filtering Techniques”, February 2017.

[10]. Dr. Ghulam M. and Majid Y., “SMS Spam Detection Using Simple Message Content Features”, Journal of Basic and Applied Scientific Research, January 2014.

[11]. Lutfun. Nahar. Lota. and B. M. Mainul Hossain., “A Systematic Literature Review on SMS Spam Detection Techniques”, University of Dhaka, 2017

[12]. Sarah J. D. and Mark Buckley., “SMS Spam Filtering: Methods and Data”, Technological University of Dublin, 2012.

[13]. Md. Rafiqul Islam and Morhsed U. Chowdhury., “Spam Filtering Using ML Algorithms”, Deakon Univerisity, Victoria, Australia, 2005.

[14]. Qian Xu. and Evan Wei Xiang., “SMS Spam Detection Using Noncontent Features”, December 2012.

[15]. Tiago Almeida., “Towards SMS Spam Filtering: Results under a New Dataset”, November 2013.

[16]. Tarek. M. Mahmoud. and Ahmed. M. Mahfouz., “SMS Spam Filtering Technique Based on Artifial Immune System”, Minia University, Egypt, March 2012.

[17]. Tiago A. Almeida. and Jose Maria G. Hidalgo., “Contributions to the study of SMS spam filtering: new collection and results”, September 2011.

[18]. Gaurav. Sethi. Vijender Bhootna., “SMS Spam Filtering Application Using Android”, RV College of Engineering, 2014.

[19]. Gordon. V. Cormack. and Enrique. Puertas. Sanz., “Feature Engineering for Mobile SMS Spam Filtering”, University of Waterloo, Waterloo, Ontario, Canada, July 2007.
[20]. Gordon. V. Cormack. and Enrique. Puertas. Sanz., “Spam Filtering for Short Messages”, University of Waterloo, Waterloo, Ontario, Canada, November 2007.

APPENDICES

Appendix – A

Android Studio	Android studio is an official integrated development environment (IDE) for Google’s android operating system, built on JetBrain’s IntelliJ IDEA software and designed specifically for android development. It is available for download on windows. It is the replacement for the Eclipse android development tools (ADT) as primary IDE for native android application development
App	A type of mobile application, also known as a computer program designed to run on smartphones
Android	Operating System developed by Google Inc. specifically designed for smartphones and tablets
GUI	Graphic User Interface
Android App	An Android based application developed using Java and Python exclusively for Android Devices
Windows	Operating System
Scripting	A scripting language or script language is a programming language that supports the writing of scripts, programs written for a special runtime environment that can interpret and automate the execution of tasks which could alternatively be executed one-by-one by a human operator.
Graphics	Graphics are visual presentations of app
UX	User experience for using a product.

User	User is the one who use the app to read listen and watch the stories of prophets
SMS	A type of text message component for mobile devices
Spam	Irrelevant or unsolicited messages sent over the Internet, typically to a large number of users, for the purposes of advertising, phishing, and spreading malware.
System	A system is a set of interacting or interdependent components forming an integrated whole or a set of elements and relationships which are different from Relationships of the set.
Software	A collection of data or instructions that tells a computer to do certain tasks
SRS	SRS software requirement specification
Use Case	List of actions typically defining the interactions between a role and a system to achieve a goal
Sequence Diagram	A Sequence Diagram (SD) is an interaction diagram that shows how object operate with one another and in what order.
Component Diagram	Components are wired together by using an assembly connector to connect the required interface of one component with the provided interface of another component. This illustrates the service consumer - service provider relationship between the two components.
Deployment Diagram	A deployment diagram in the Unified Modeling Language models the physical deployment of artifacts on nodes.

Appendix – B

Activity Diagram	An activity diagram visually presents a series of actions or flow of control in a system similar to a flowchart or a data flow diagram. Activity diagrams are often used in business process modeling.
Data Flow Diagram	A data flow diagram (DFD) illustrates how data is processed by a system in terms of inputs and outputs. As its name indicates its focus is on the flow of information, where data comes from, where it goes and how it gets stored.
State Diagram	State machine diagram is a behaviour diagram which shows discrete behaviour of a part of designed system through finite state transitions.
Test Case	A test case in software engineering is a set of conditions or variables under which a tester will determine whether an application or software system is working correctly or not.
Random Forest	Random forests, otherwise known as the random forest model, is a method for classification and other tasks. It operates from decision trees and outputs classification of the individual trees. Random forests correct for the habit of decision trees to over fit to their training set.
SVM	In machine learning, support-vector machines (SVMs, also support-vector networks) are supervised learning models with associated learning algorithms that analyze data used for classification and regression analysis.
Python	Python is an interpreted, high-level, general-purpose programming language mostly used in machine learning
Program	A computer program is a collection of instructions that performs a specific task when executed by a computer.

Sublime Text	Sublime Text is a proprietary cross-platform source code editor with a Python application programming interface (API).
Java	Java is a general-purpose programming language that is class-based, object-oriented (although not a pure OO language, as it contains primitive types), and designed to have as few implementation dependencies as possible.
Smartphone	A mobile phone that performs many of the functions of a computer, typically having a touchscreen interface, Internet access, and an operating system capable of running downloaded apps.
Code	A set of instructions written in a specific programming language given to a computer to perform certain tasks

The University of Lahore, Islamabad Campus
Department of CS and IT
<u>Plagiarism Certificate</u>

SMS Spamming Application
By
Ibrahim Irfan

(CSU-F15-104)

Bilal Ahmed

(CSU-F15-105)

Submission Date: 13-Dec-2019 11:43AM (UTC+0500) Submission ID: 1233725757
File name: FYP_Documentation_Without_Index_and_References.pdf (1.06M)
Word Count: 11579
Character count: 57188

Plagiarism checked by: Mehreen Shah_____________________
Plagiarism Result: 12%