**CHAPTER FOUR**

**RESULTS AND DISCUSSION**

**4.1 Introduction**

This chapter describes the implementation and analysis of end-to-end secure SMS system whose architecture is discussed in Chapter Three. The architecture described includes a number of components and each component has been implemented as a separate module. It is a mobile application on android operating system. It shows through its implementation, security issues which are common that has been tackled when it comes to SMS. The system developed ensures that the messages are secured from the sender’s and receiver’s ends.

**4.2 Implementation Environment**

The implementation of this work was done using the Android Studio. Android Studio is an integrated development environment (IDE) for developing primarily with Java, but also with other languages. It is also an application platform framework for Java Mobile applications and others. The Android Studio IDE is written in Java and can run on Windows, OS X, Linux, Solaris and other platforms supporting a compatible JVM.

The Android Studio Platform allows applications to be developed from a set of modular software components called modules. Applications based on the Android Studio Platform (including the Android Studio IDE itself) can be extended by third party developers. Android Studio IDE is an open-source integrated development environment. Android Studio IDE supports development of all Java application types (Java SE (including JavaFX), Java ME, web, EJB and mobile applications) out of the box. Among other features are an Ant-based project system, Maven support, refactoring, and version control (supporting CVS, Subversion, Git, Mercurial and Clearcase).

All the functions of the IDE are provided by modules. Each module provides a well-defined function, such as support for the Java language, editing, or support for the CVS versioning system, and SVN. Android Studio contains all the modules needed for Java development in a single download, allowing the user to start working immediately. Modules also allow Android Studio to be extended. New features, such as support for other programming languages, can be added by installing additional modules. For instance, Sun Studio, Sun Java Studio Enterprise, and Sun Java Studio Creator from Sun Microsystems are all based on the Android Studio IDE.

From July 2006 through 2007, Android Studio IDE was licensed under Sun's Common Development and Distribution License (CDDL), a license based on the Mozilla Public License (MPL). In October 2007, Sun announced that Android Studio would henceforth be offered under a dual license of the CDDL and the GPL version 2 licenses, with the GPL linking exception for GNU Classpath Framework for simplifying the development of Java Swing desktop applications. The Android Studio IDE bundle for Java SE contains what is needed to start developing Android Studio plugins and Android Studio Platform based applications; no additional SDK is required.

Applications can install modules dynamically. Any application can include the Update Center module to allow users of the application to download digitally signed upgrades and new features directly into the running application. Reinstalling an upgrade or a new release does not force users to download the entire application again. The platform offers reusable services common to desktop applications, allowing developers to focus on the logic specific to their application. Among the features of the platform are: User interface management (e.g. menus and toolbars), User settings management, Storage management (saving and loading any kind of data), Window management, Wizard framework (supports step-by-step dialogs), Android Studio Visual Library and Integrated development tools. Android Studio IDE is a free, open-source, cross-platform IDE with built-in-support for Java Programming Language.

**4.3 System Requirements**

The development of this work was carried out on a HP GS78 Laptop Computer with the following specifications;

* 1 Terabytes Hard disk
* 8 Gigabytes RAM (Random Access Memory)
* 4 Gigahertz Processor Speed (Intel Pentium Dual Core).
* 64-bit Operating System (Windows 7)

The deployment and testing was carried out on HTC M9 mobile phone with the following specifications:

**4.4 Component Interaction of the Developed Secure SMS System**

On first installation the user is asked to create pin so as to secure the app. After a pin has been selected the main layout (home page) is shown. The home page of the secure SMS app shows an inbox tab and a sent tab and a send new message button. The Inbox Tab shows a list of inbox (received) messages. The Sent Tab shows a list of sent messages. The app takes over the SMS functionality of the mobile phone; this means all text messages sent to the mobile phone through the phone number enters the secure SMS app. Only messages sent to the phone from another phone that has this secure SMS app would be encrypted, other text messages sent from the default SMS app will enter the app, but would not be encrypted. All text messages sent from the secure SMS app would be delivered to the recipient phone as encrypted text messages, but only the secure SMS app on the recipient phone can open the encrypted messages, default SMS app would not be able to open the encrypted messages. When the Create New Message is clicked, a screen is brought forward and a prompt for the use to input the destination phone number and a message content. There is also a button at the bottom of the screen to encrypt the message. When the button is clicked a key is generated using ECC encryption and the key is used to encrypt the message using blowfish encryption and then encoded in base64 so as to pass over the network and the encrypted message is sent over the network. After clicking on the message the user wants to view, a screen is brought forward and the encrypted message is shown. At the bottom , there is a button named “Decrypt”, this decrypts the Message content for view,  if the message is of an encrypted type, the plain text is shown but if not,  an error message is shown.

For an encrypted message, the key is generated at the point of creation of the text message. At the recipient’s end, the key is extracted from the content and used to decrypt the main body of the message and it is displayed to the user. When the “Create New Message” is clicked, a screen is brought forward and a prompt for the use to input the destination phone number and a message content. There is also a button at the bottom of the screen to encrypt the message. When the button is clicked a key is generated using ECC encryption and the key is used to encrypt the message using blowfish encryption algorithm and then encoded in base 64 so as to pass over the network and the encrypted message is sent over the mobile network. On every run of the app, the user has to input the security pin created so as to be able to access the app. All the explained moves are shown in figures 4.1 to 4.9

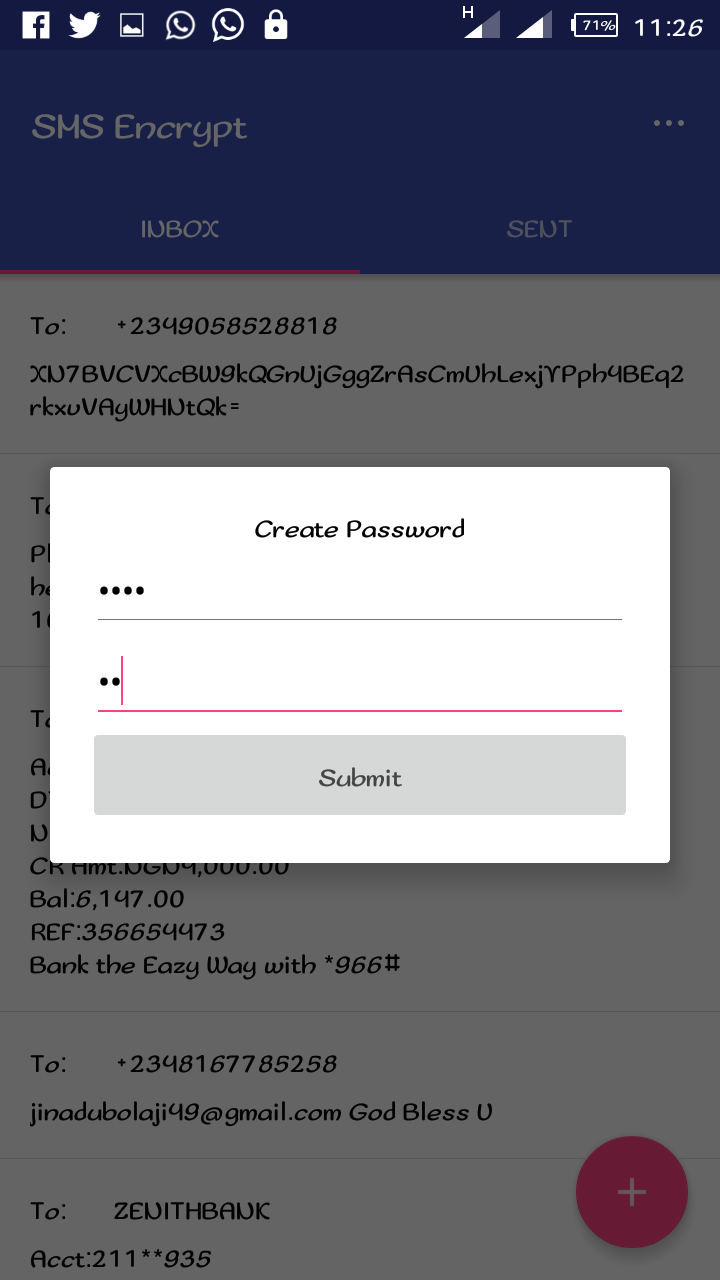


Figure 3.1 Login Pin generation page

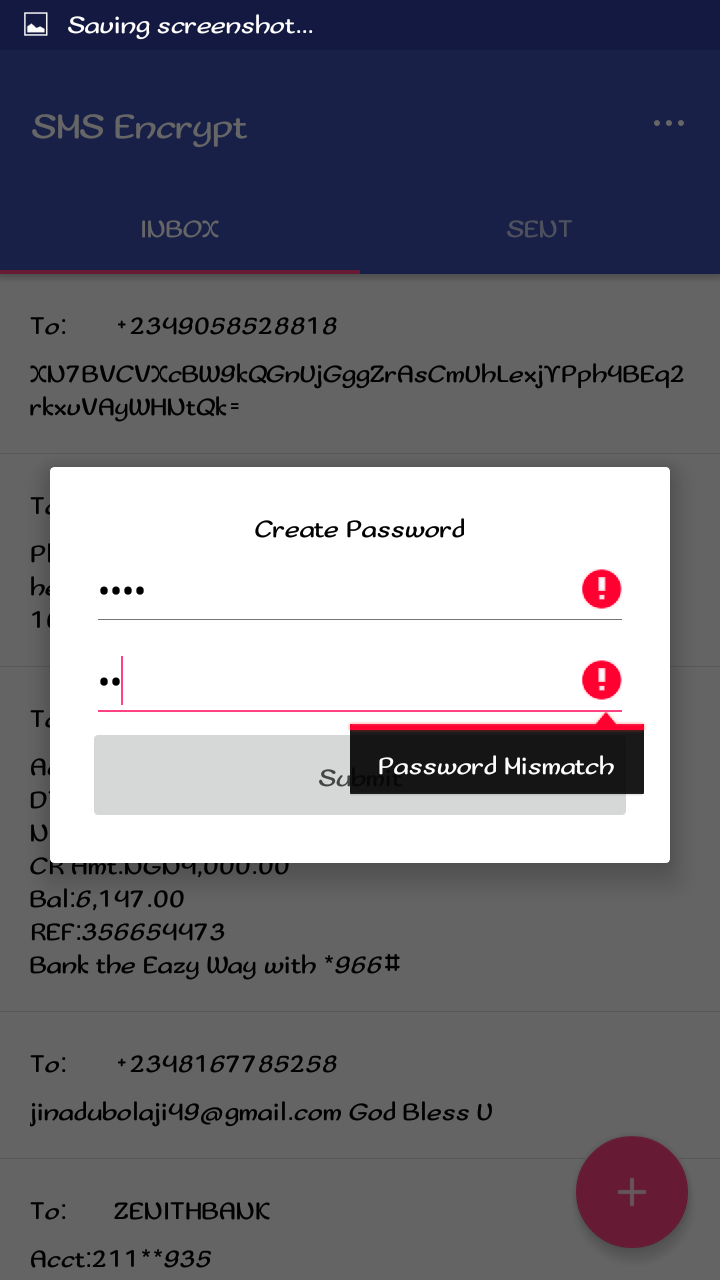


Figure 4.2 Security Pin authentication page

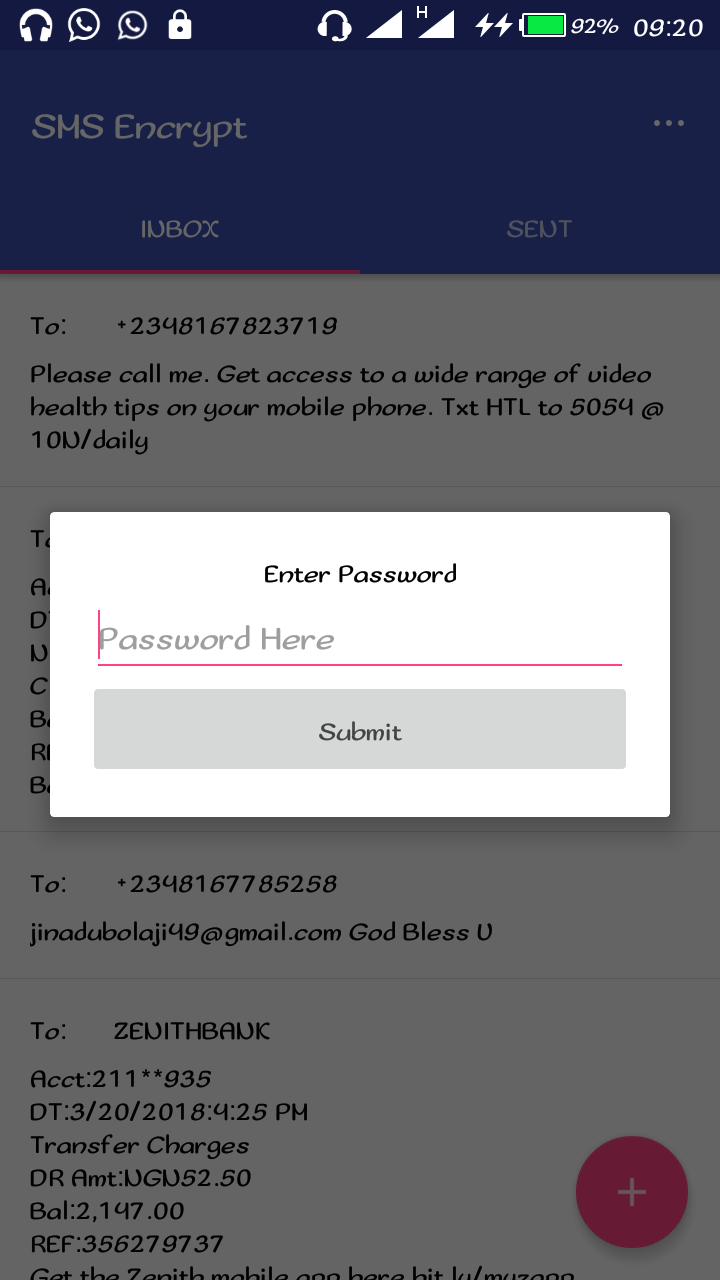


Figure 4.3 Login page to the SMS app after pin creation

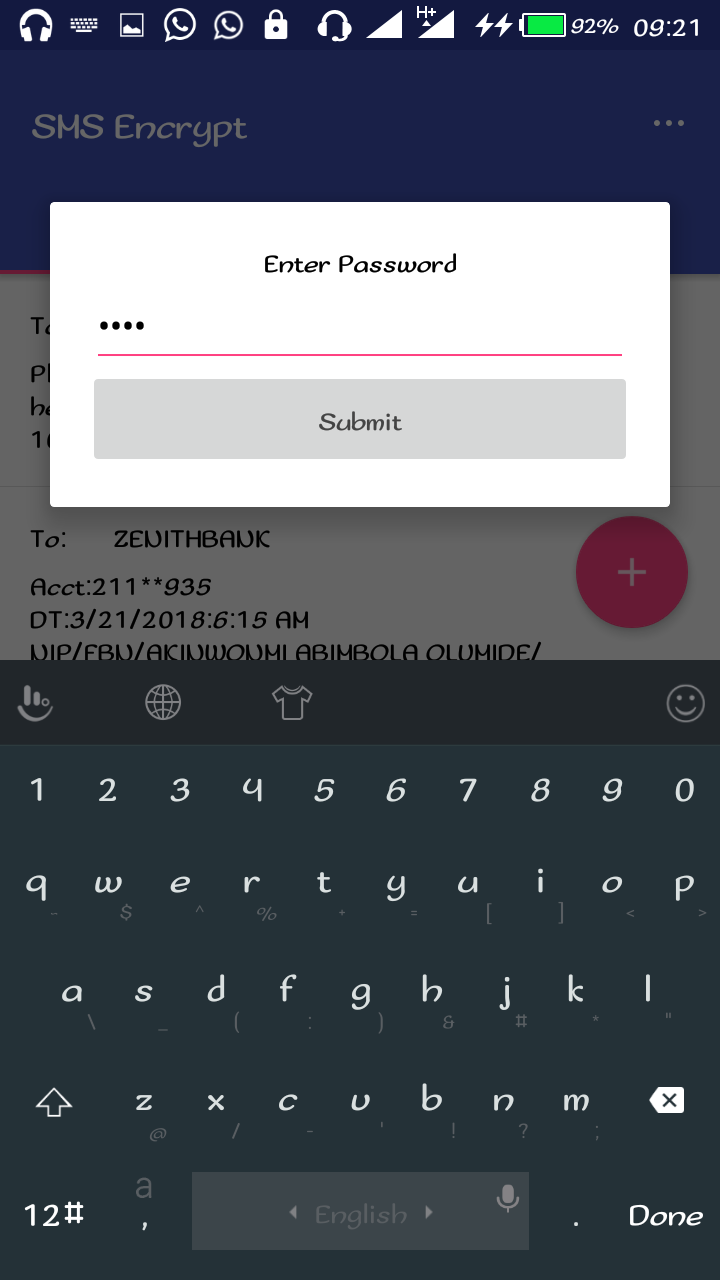


Figure 4..4 Login page after the input of the security pin

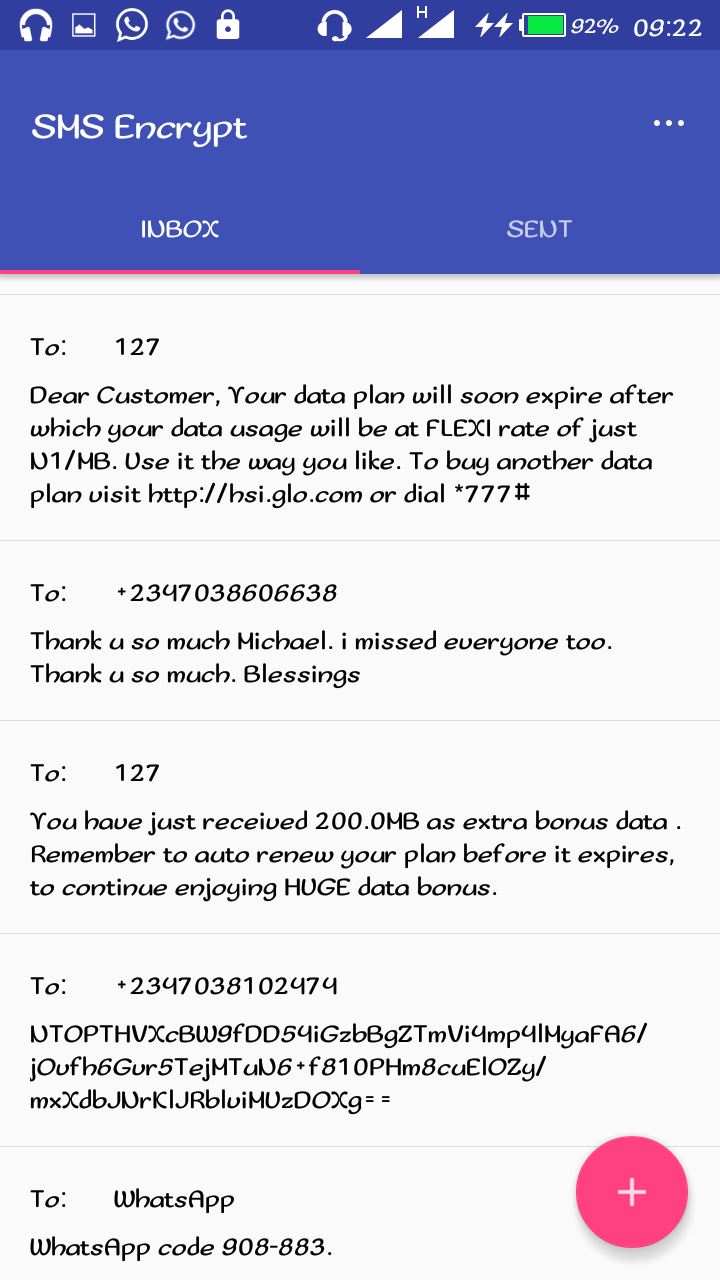


Figure 4.5 Message inbox showing encrypted and normal text messages on the phone

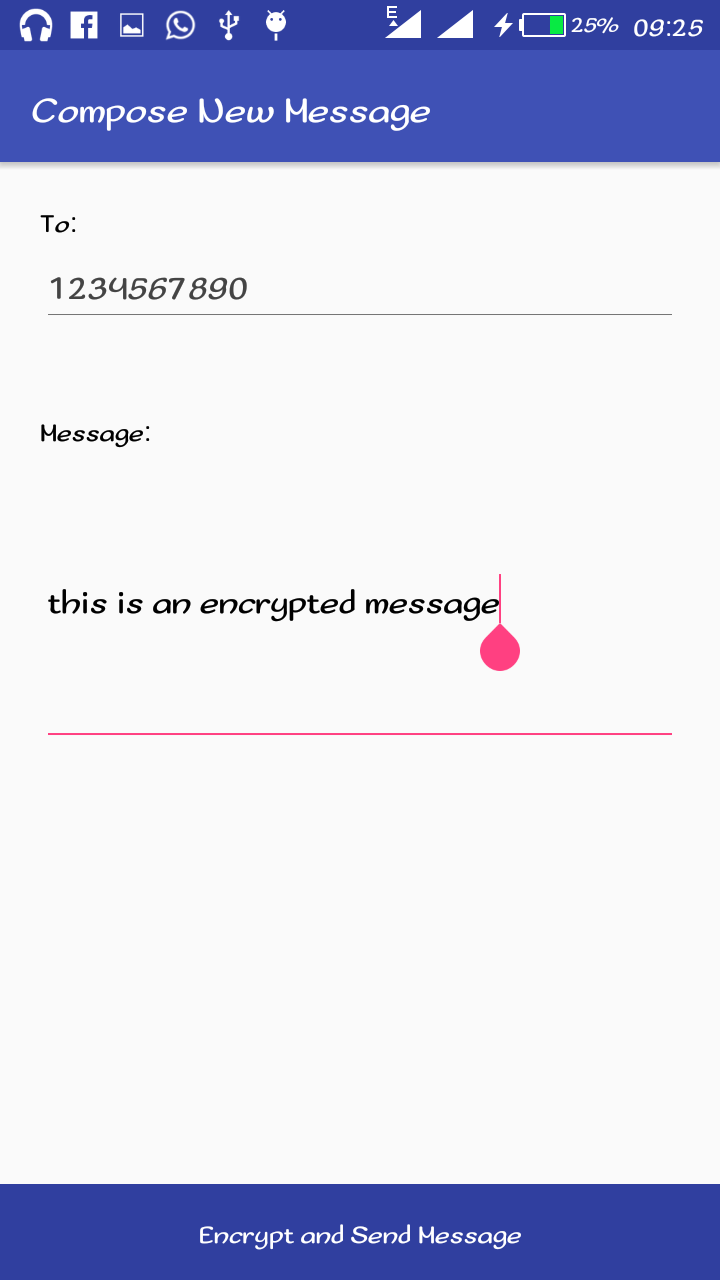


Figure 4.6 User composing new encrypted message

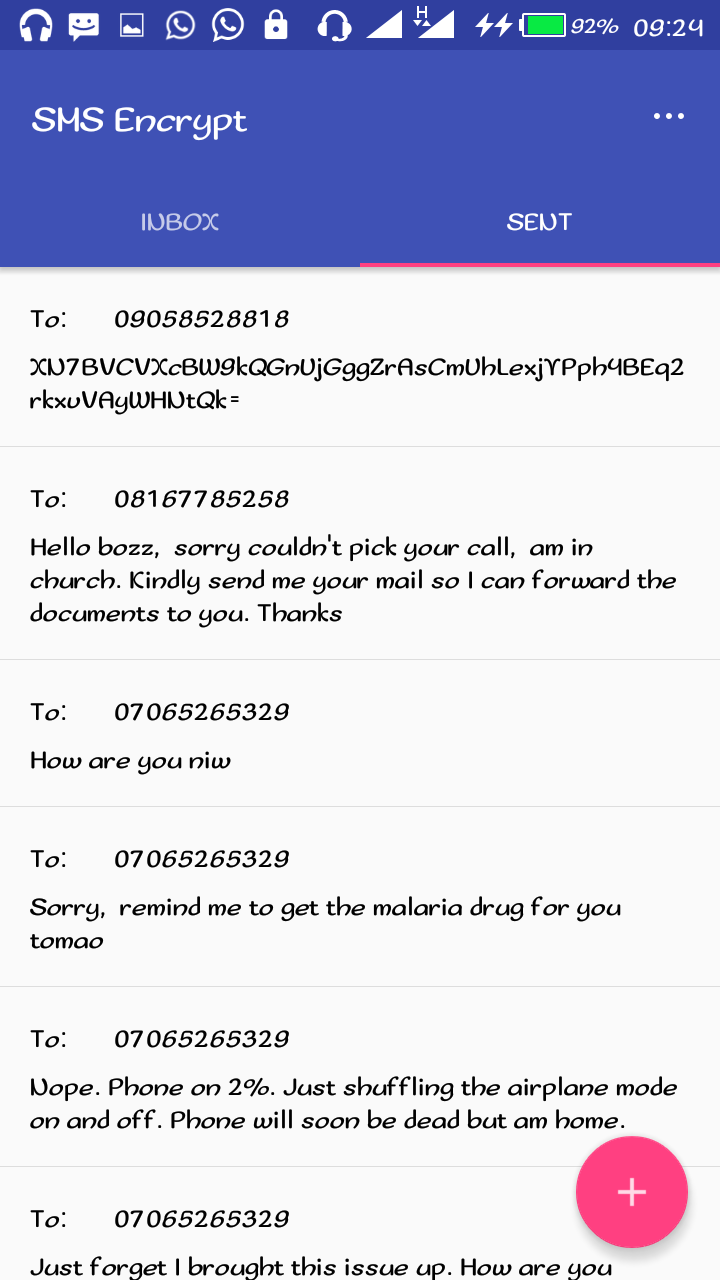


Figure 4.7 List of sent normal and encrypted messages on the phone



Figure 4.8 New incoming message from another user decrypted



Figure 4.9 Encrypted message from another user

**4.5 Testing of the developed System**

Testing evaluates a software product to ensure that it satisfies its planned purpose. This sub section discusses the testing process that is carried out to evaluate the successful implementation of this project. The main purpose of this test is to describe the testing details of the use cases of the secure SMS system and to identify the features of the system that were tested, to identify and define all the activities necessary to prepare for and conduct the testing process.

**4.5.1 Test Approach**

This section of the test plan describes the overall approach for testing of the system. The approach followed for testing the system ensures that the major features of the project are effectively tested. The system is validated and verified to make sure that the system fulfils the aim and objectives stated in chapter one of this report. The approach used for testing the system is unit testing and real life usability testing on different android based mobile phones with different screen sizes.

* + 1. **Unit Testing**

Unit testing tests each single services of the project to check for errors. This is mainly done to discover errors in the code, to isolate each part of the program and to check the correctness of the code.

**4.5.3 Test Case Specifications and Results**

The test cases specify the inputs, predicted results and execution conditions. Each test case aims to evaluate the operation of a service element of the system.

* + - * **Test Items**

The items tested are all of the developed secure SMS services of the system.

* + - * **Test case pass/fail criteria**

This includes criteria to identify decision rules that are used to determine whether a test case passes or fails a test.

* + - * **Pass Criteria**: The test cases executed on the system will pass if they meet all specific requirements of the system. All functional and non-functional requirements of the secure SMS mobile app met the pass criteria.
      * **Fail Criteria:** A test case is said to fail, if the desired functionality is not satisfied by the system. None of the functional and non-functional requirements of the secure SMS mobile app met the fail criteria.
      * **Testing Environment-** Students of the department of Computer Science of Adeleke University have helped in conducting this test on different android based mobile phones and tablets to make sure successful operation of services. The results were all positive.
      * **Valid Data-** Data is valid only when it is consistent with the defined data types in the data contract. Valid data keyed in on the mobile app were correctly processed.
      * **Invalid Data-**Data which is not consistent with the one defined in the data contract to

use the services operation. All invalid data keyed in were correctly rejected with the right exceptions/error messages are necessary. Both valid and invalid data were provided in the test in order to evaluate the service’s response in both cases.

The mobile android OS based secure SMS system was duly tested and certified working perfectly, no error report was given on any page of the web application. The system consists of pages stylishly linked together to deliver a good SMS experience to the users, it opens well on virtually all android based mobile phones and tablets. All links were tested and certified working, making sure none is broken, the system was also tested on various models mobile phones and versions of android OS, other than the one used for its implementation. The developed system was tested over and over, and every bug detected was corrected over and over until the system worked perfectly.

**4.6 System Maintenance**

This type of maintenance proves the lifespan and efficiency of the program. Having a mobile android based secure SMS system on one’s phone has its responsibilities; proper and regular maintenance would make it run smoothly always. The system developed will stand the test of time if properly maintained.

**4.7 Users’ Guide**

This SMS system has been developed to a level that will enable users to navigate around easy and quick. The system is menu driven whereby, for any particular operation carried out, a menu must be specified and there are few number of options for the user to select depending on what the user wants to do. Therefore, the application has been made simple enough for use, and no extra document needed as users’ guide.