**CHAPTER FOUR**

**RESULTS AND DISCUSSION**

**4.1 Introduction**

Chapter four describes the implementation and analysis of drug reminder system whose architecture has been 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 the implementation and other design issues, which are common to mobile applications, and the way they have been tackled in this reminder application software. The system developed ensures that the user is notified of the set reminders and that the details of the reminders as well as other user’s data are secure from unauthorized access.

**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 Class path 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;

* 500Gb Hard disk
* 4 Gigabytes RAM (Random Access Memory)
* 4 Gigahertz Processor Speed (Intel Corei3).
* 64-bit Operating System (Windows 7)

The deployment and testing was carried out on Samsung S6 mobile phone with the following specifications:

**4.4 Component Interaction of the Developed Reminder System**

Medication Reminder is a two in one reminder app which allows you to set a medication alarm and a medical appointments also. The user of the app signs up/signs in on the app and the user is granted access into the app. The user can either choose the medical reminder or doctor’s appointment feature. When the user chooses the medication reminder, the list of all the medication that has be set for alarms is shown. The user can delete or edit them. The user can also add a new reminder for medication by clicking on the button at the bottom of the screen.

When the user chooses the doctor’s appointment, the list of all the appointments that has been set by the user is shown, the user can edit or delete the records. The user can also add a new appointment by clicking on the button at the button of the screen.

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 index page of the secure reminder app as shown in figure 4.1shows two options, new users are to first register while existing users are to login as shown in figure 4.3 to the created account. After a successful login, the user is taken to the homepage where the two key functionalities (drug reminder and doctor’s appointment) of the app are displayed as shown in figure 4.3. The logged in user choosing the drug reminder can set reminders for a drug to be taken a specific number of days, at different hours of the day as shown in figure 4.4. The user can view the list of all drug reminders created on the app as shown in figure 4.5 and can choose to update any of the reminder item on the list as shown in figure 4.6. The patient can create the doctor’s appointment reminder in a similar way to the creation of drugs reminders. The user can view the created appointment reminders as shown in figures 4.7 and 4.8 and update an existing reminder as shown in figure 4.9

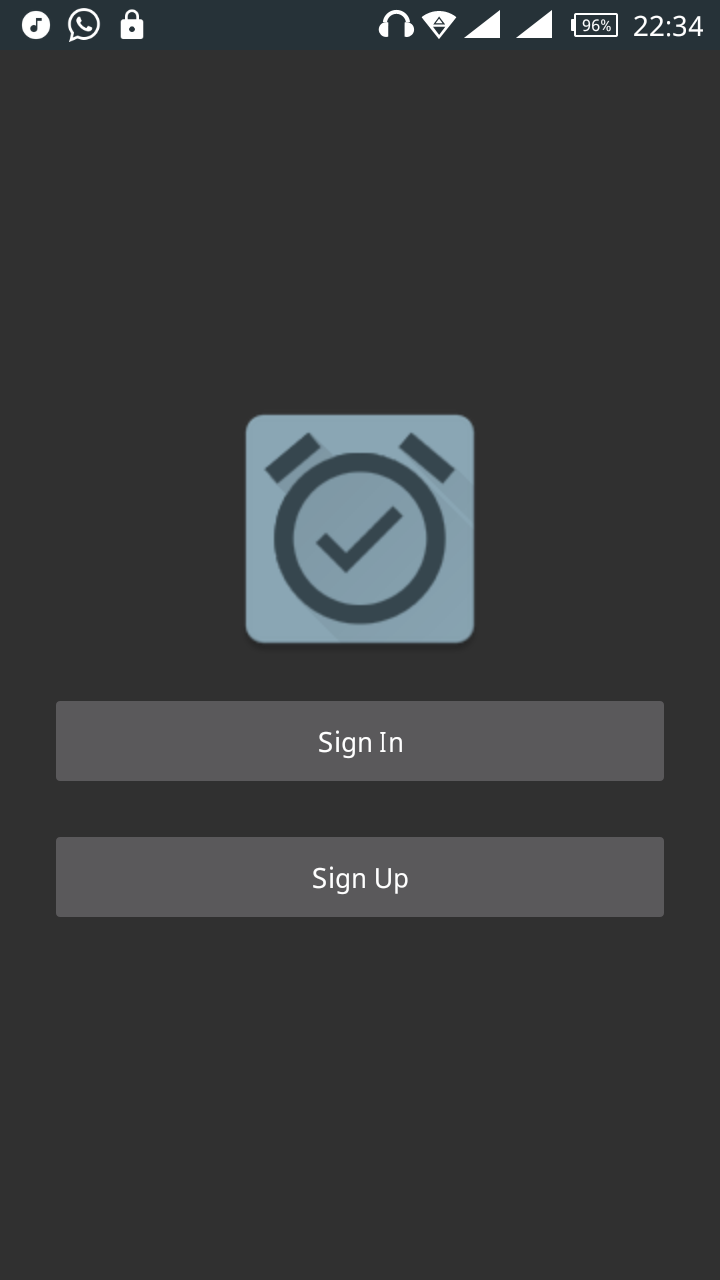


Figure 4.1 Activity Selection

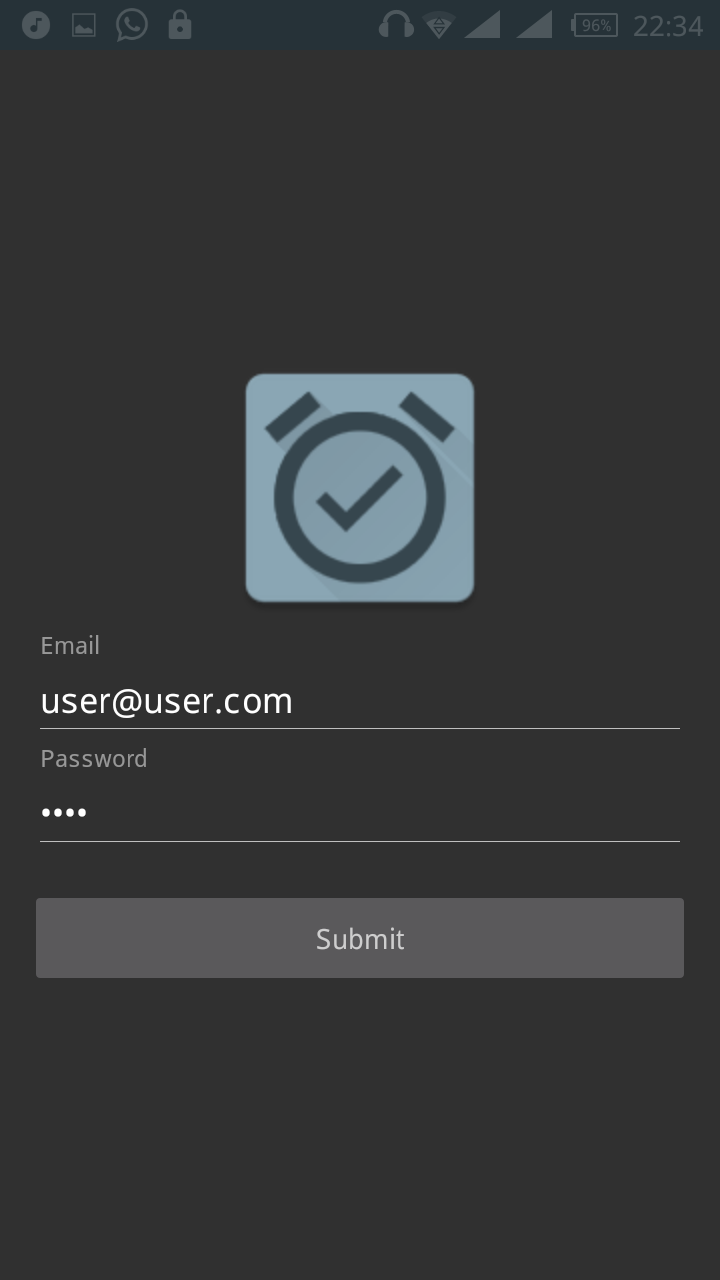


Figure 4.2 Registered user login page

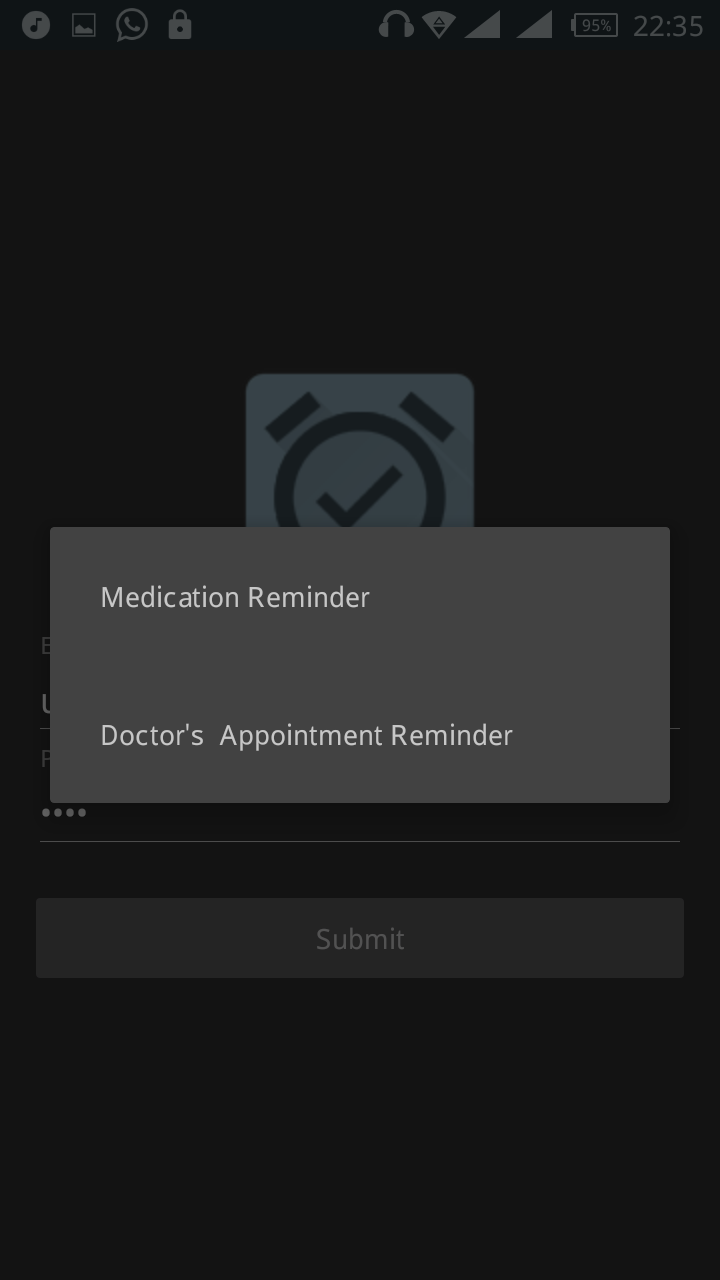


Figure 4.3 User’s page showing the key functionalities of the mobile app

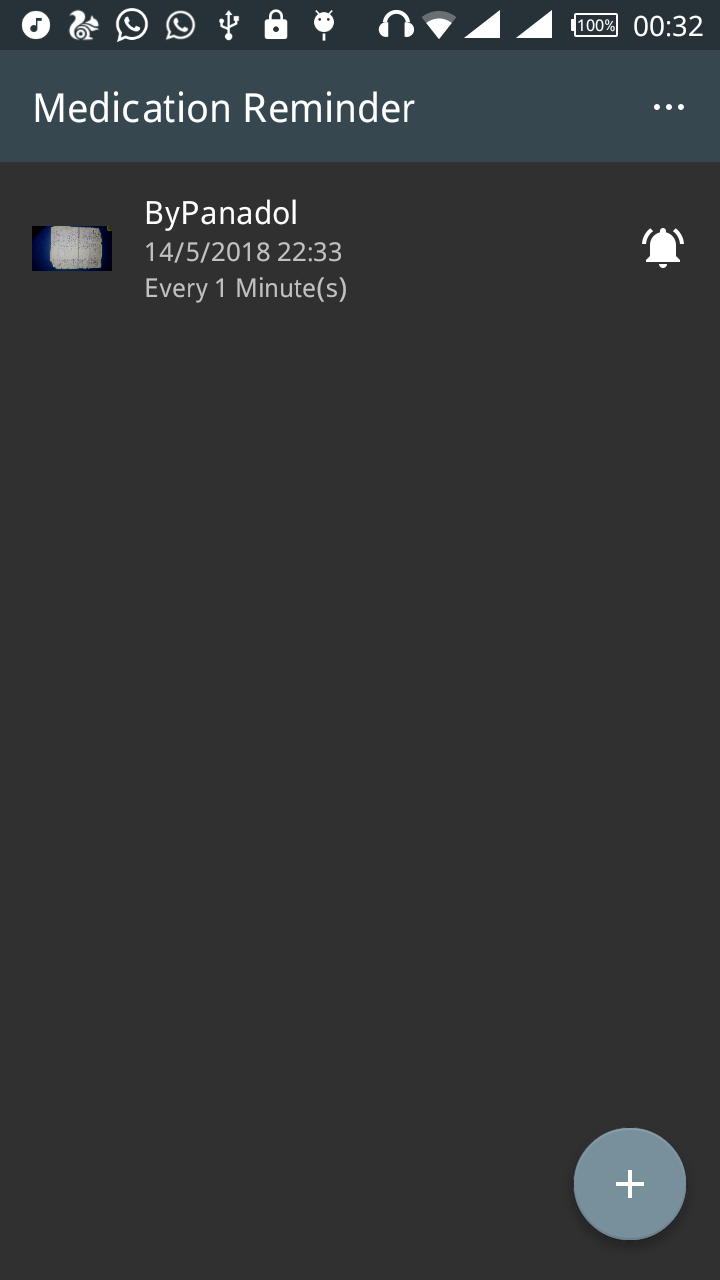


Figure 4.4 Drug reminder page

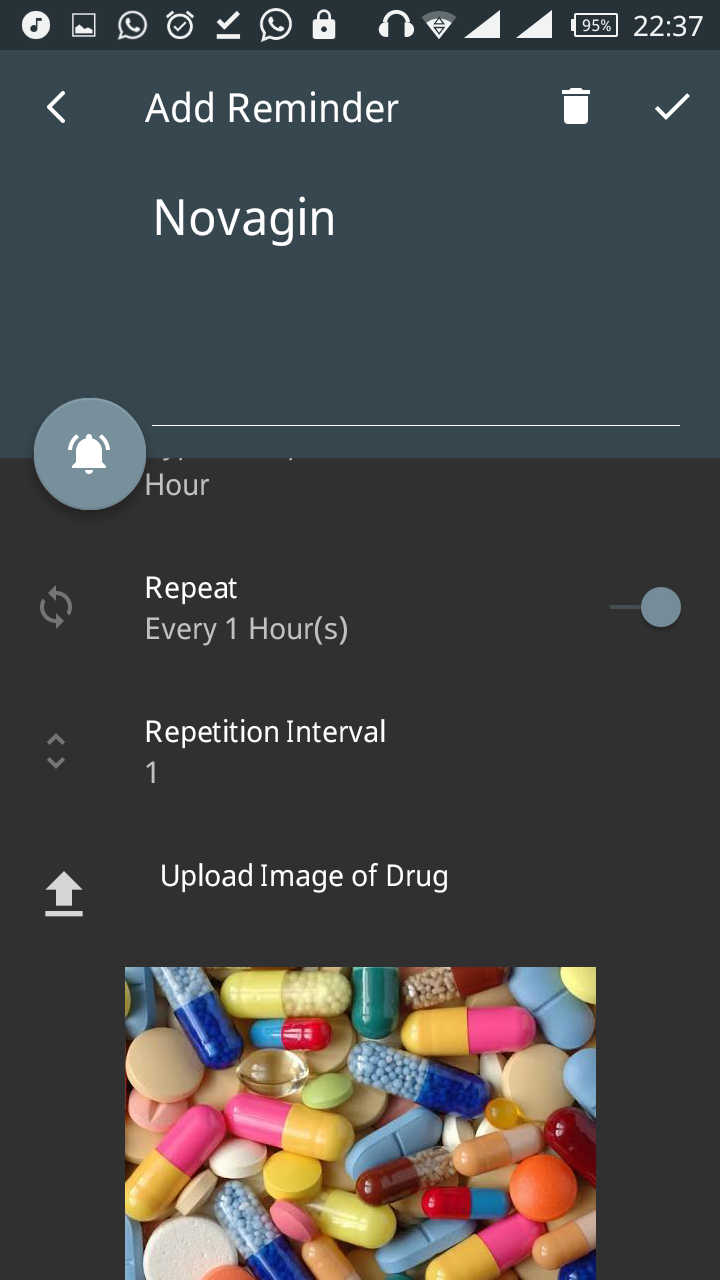


Figure 4.5 User viewing the list of drug reminders created

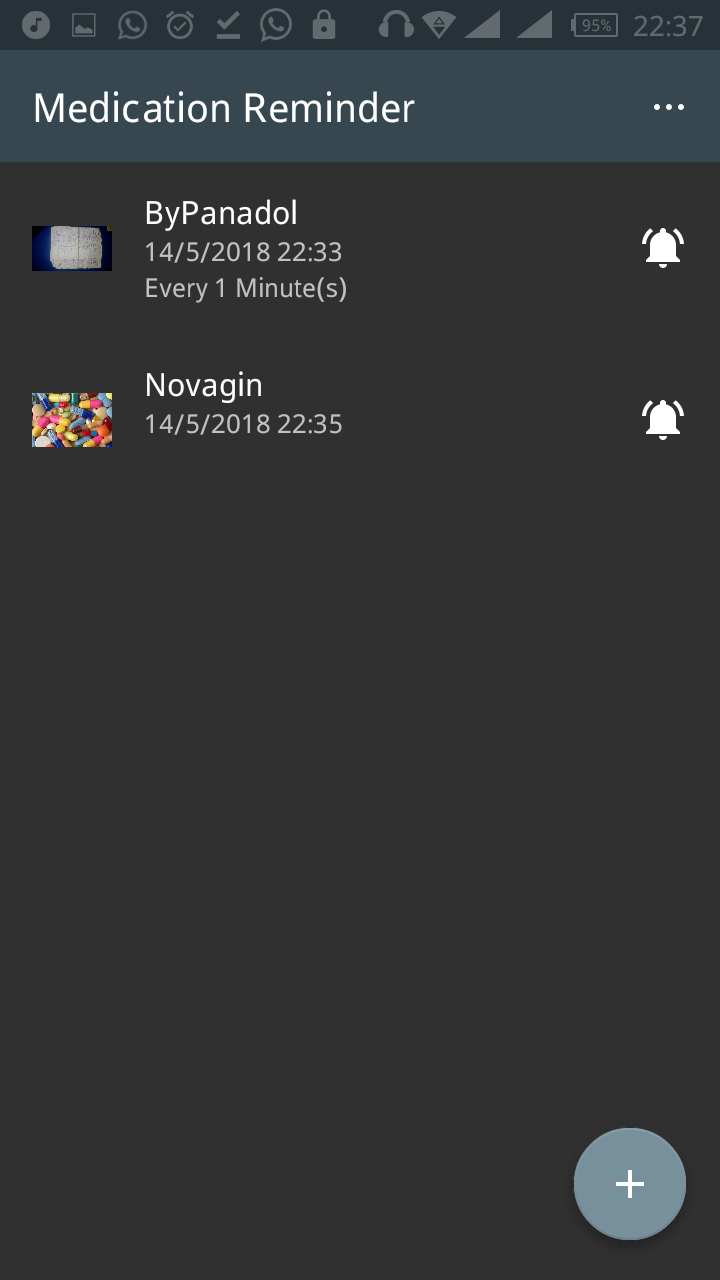


Figure 4.6 List of set reminders

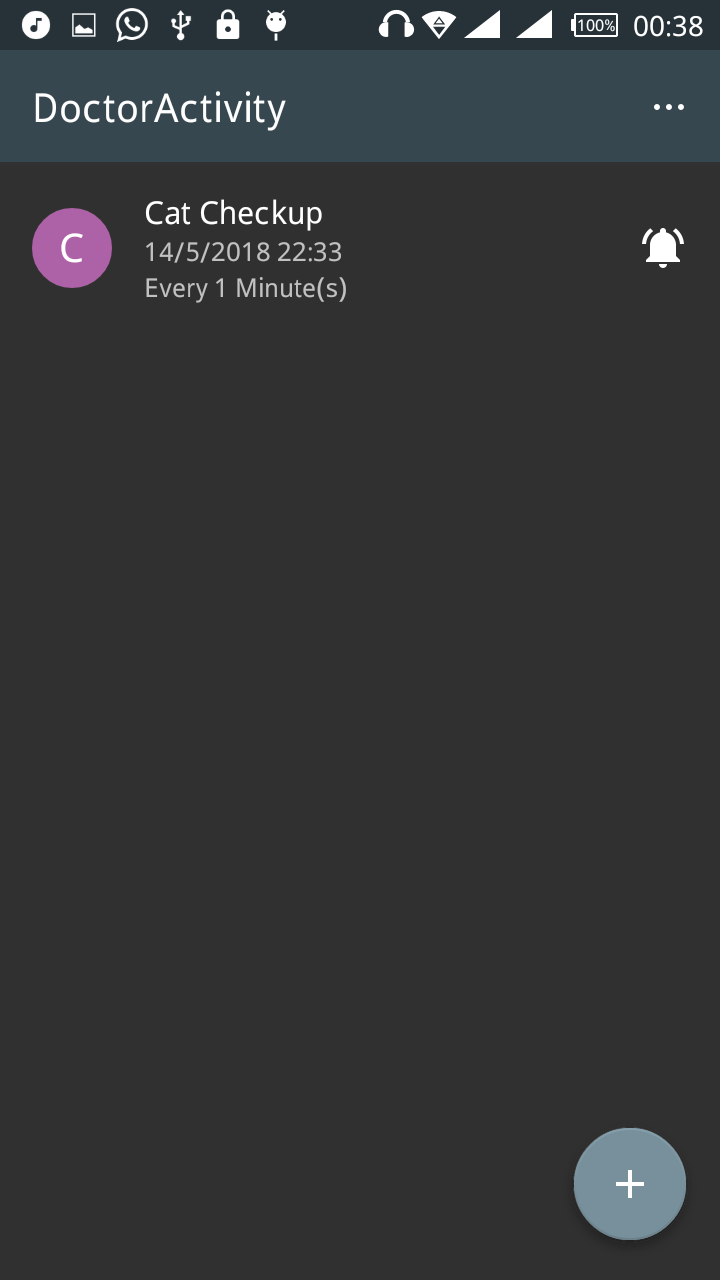


Figure 4.7 User viewing the doctor’s appointment reminder.

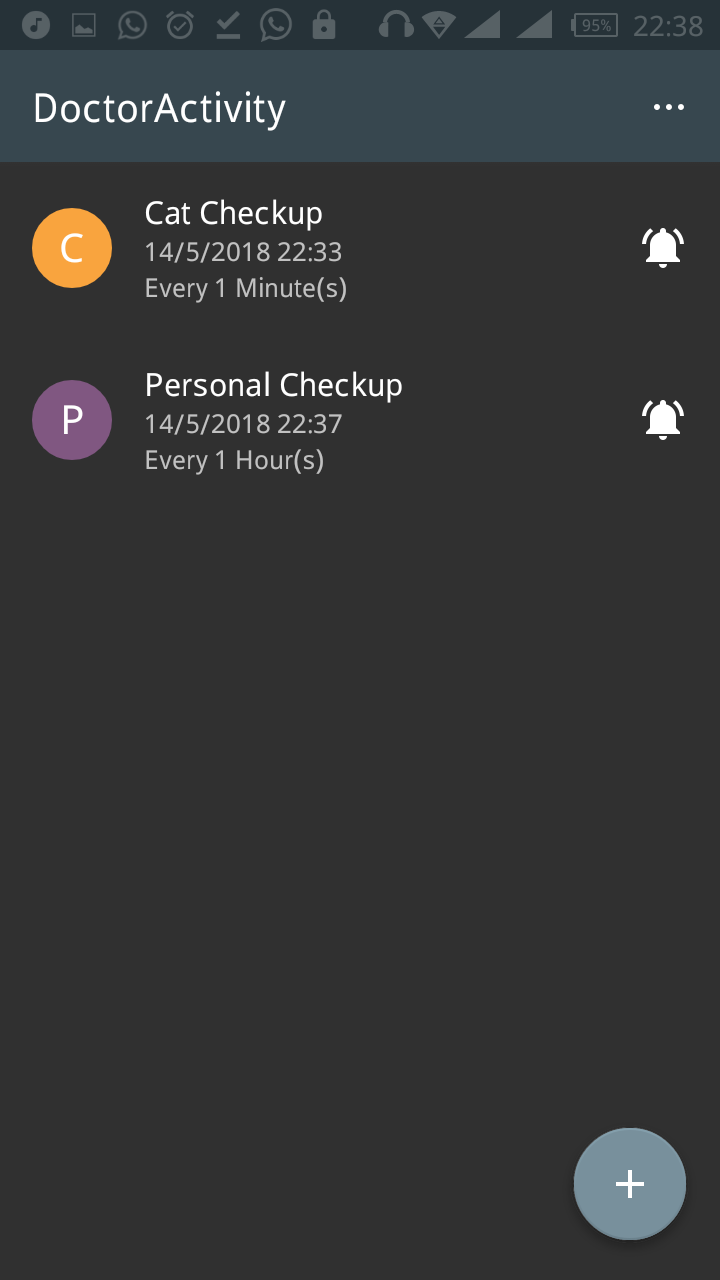


Figure 4.8 User’s appointment reminder list

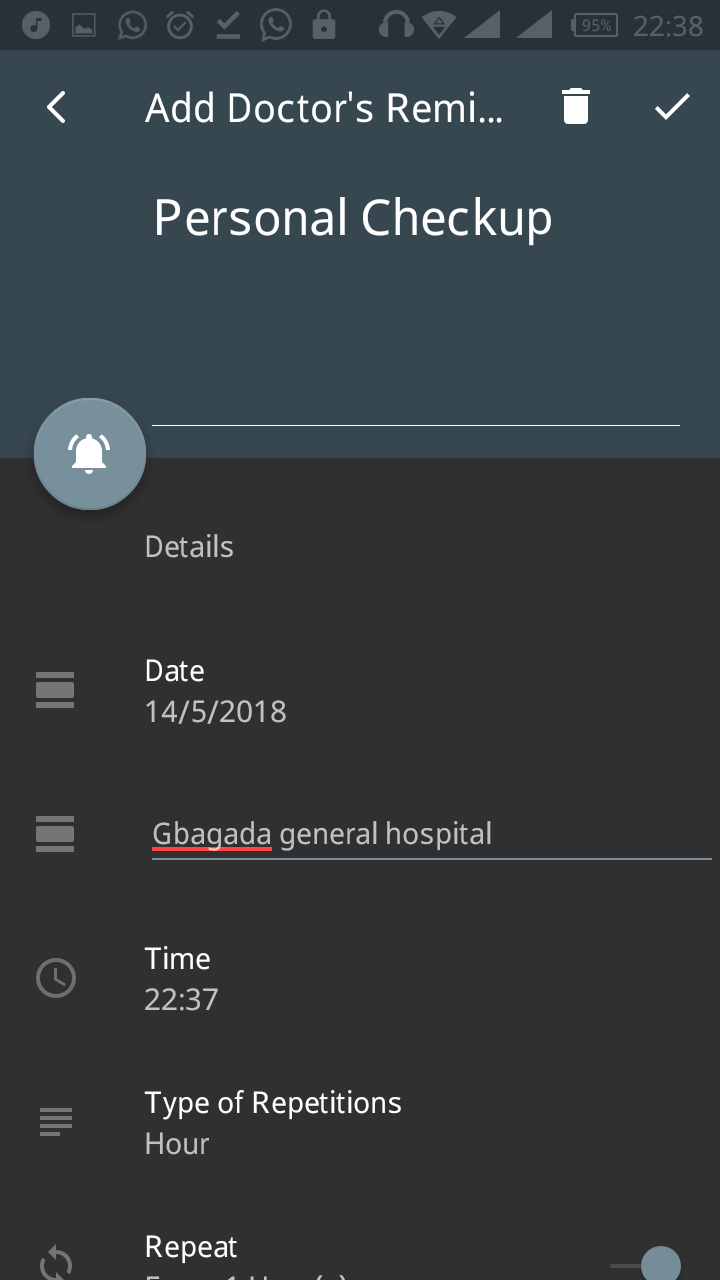


Figure 4.9 User updating a doctor’s appointment

**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 drug and doctor’s appointments reminder 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 reminder, alarm and other 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 reminder system mobile application (as described in chapter three) 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 reminder system mobile application (as described in chapter three) met the pass criteria.
      * **Testing Environment-** Samples of both in-patients (admitted) and out-patients (those taking medications out of the clinic) of Adeleke University health center (Students of Adeleke University) have helped in conducting this test on different android based mobile phones and tablets to make sure successful operation of reminder, alarm, notification and other 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 drug and doctor’s appointment reminder system was duly tested and certified working perfectly, no error report was given on any page of the android based mobile application. The system consists of pages stylishly linked together to deliver a good reminder 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 reminder 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 reminder 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 aside the details offered in this chapter four of the project write-up.