**CHAPTER FOUR**

**RESULTS AND DISCUSSION**

**4.1 Introduction**

This chapter describes the implementation and analysis of speech enabled airline reservation 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 630 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:

**4.3 Component (Architecture) Interaction of the Developed Reminder System**

The speech enabled mobile airline reservation system applications was developed using android studio for software design on android mobile devices. Java is used for the front-end and SQLite server used for the database back-end. Security, which is the most paramount factor, is ensured in such a way that only the authorized user is allowed to have access to usage of this application on its mobile phone. Figure 4.1 to 4.14 show the steps involved in operating the developed speech enabled mobile application booking a flight with the developed speech enabled application as screenshots.

The first stage is the login page of the designed software. This is the point of authentication, the authenticated user is then approved and home page comes up, that is the booking page for the designed software where the user selects a particular flight that suites her schedule. The payment page where the user will have to enter the payment card number and a confirmation page will be sent if authentic (although, this page is left as dummy because it is just a prototype). If not, a message will be sent to indicate a wrong entry and the booking will not be successfully completed. The receipt for the airplane seat reservation is then generated, which will be used at the point-of-entry in the airport when travelling when the seat reservation process has been duly completed. The user is then allowed to save and print out the confirmation page which will be used at the point-of-entry of the airplane.

The overall user flow of the speech enabled mobile airline reservation system can be summarized as follows:

1. An admin can add flights, delete flights and declare a flight status as closed.
2. An admin logs into the app using the administrator credentials. The admin can add crucial flight information like:
3. Merchant name e.g. Dana airline
4. Departure: e.g. Lagos
5. Destination e.g. Abuja
6. Time of Departure
7. Journey time: 130 minutes
8. Amount
9. The admin can also declare a flight as closed probably after the time of departure has elapsed.
10. The user logs into the app using their credential. They can view all the flight list and select a flight to book.
11. A user can also filter or search for a flight by using the speech to text function in the search part of the app.
12. A user can view the history of their bookings and delete anyone they see fit.
13. When a user books a flight. A sit number is automatically generated for them. This is why a user cannot book the same flight twice.

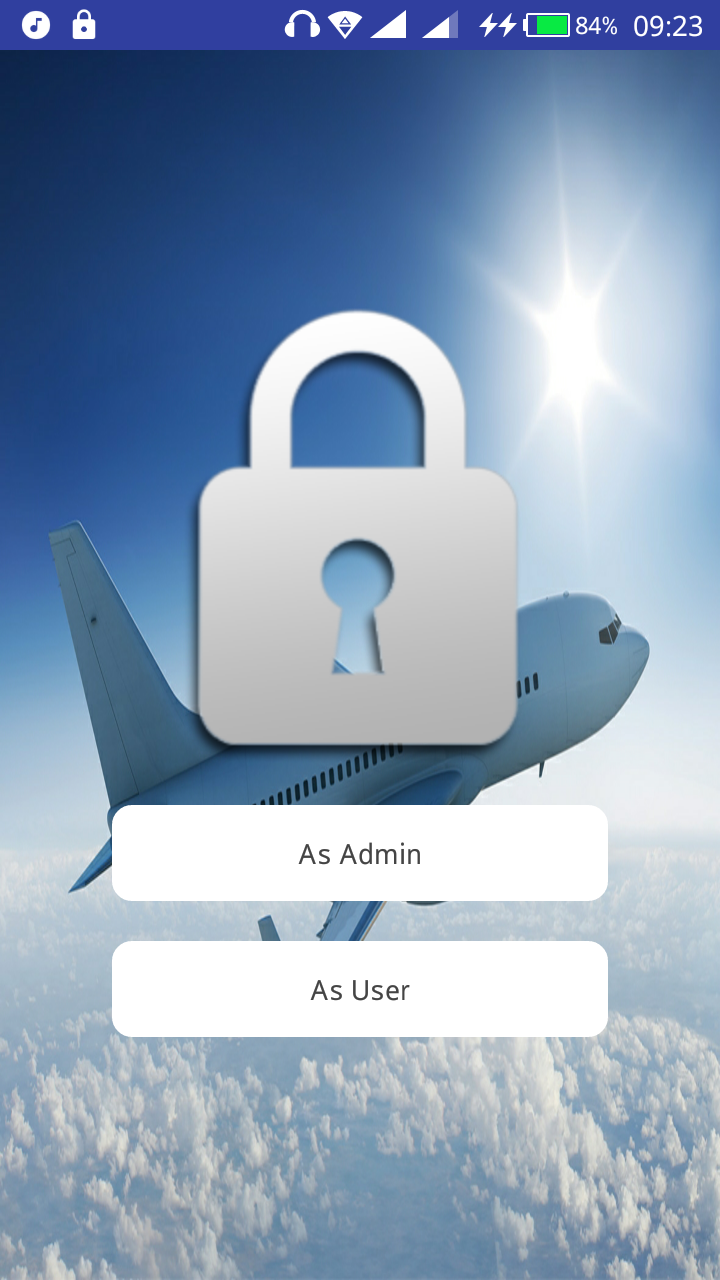
****

Figure 4.1 The user role selection page

****

Figure 4.2 Admin login page

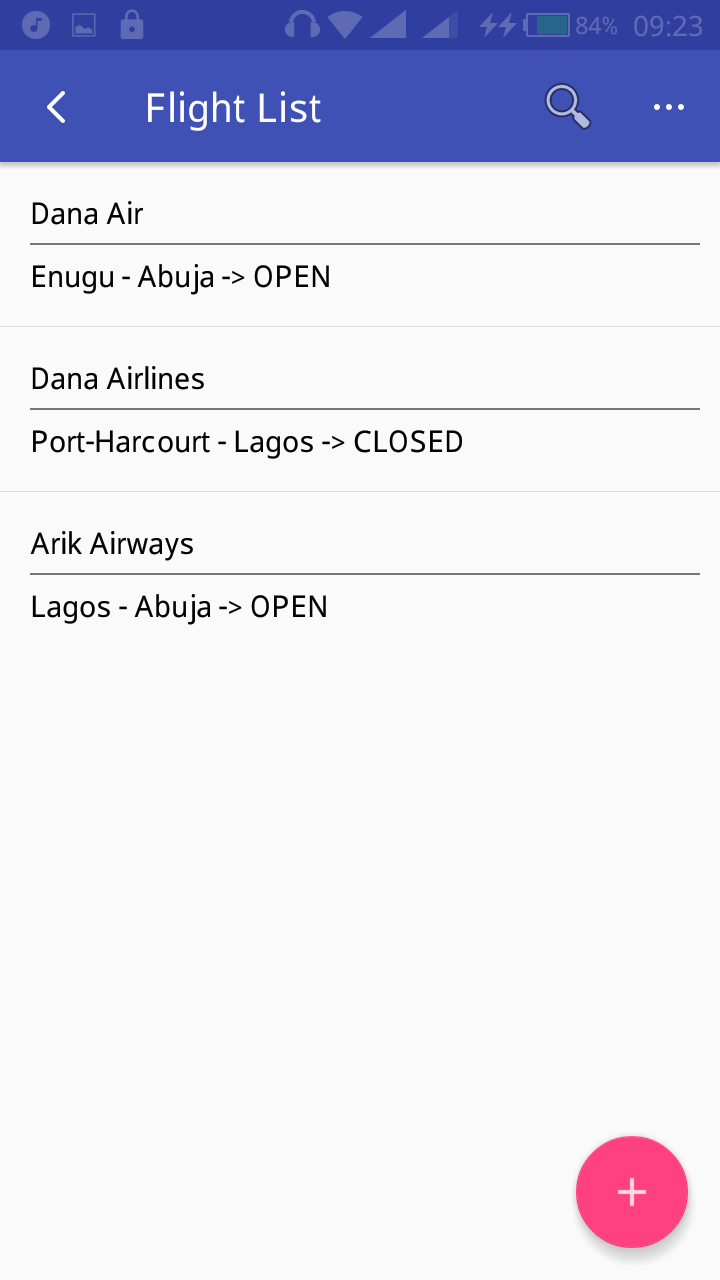
****

Figure 4.3 Flight list page

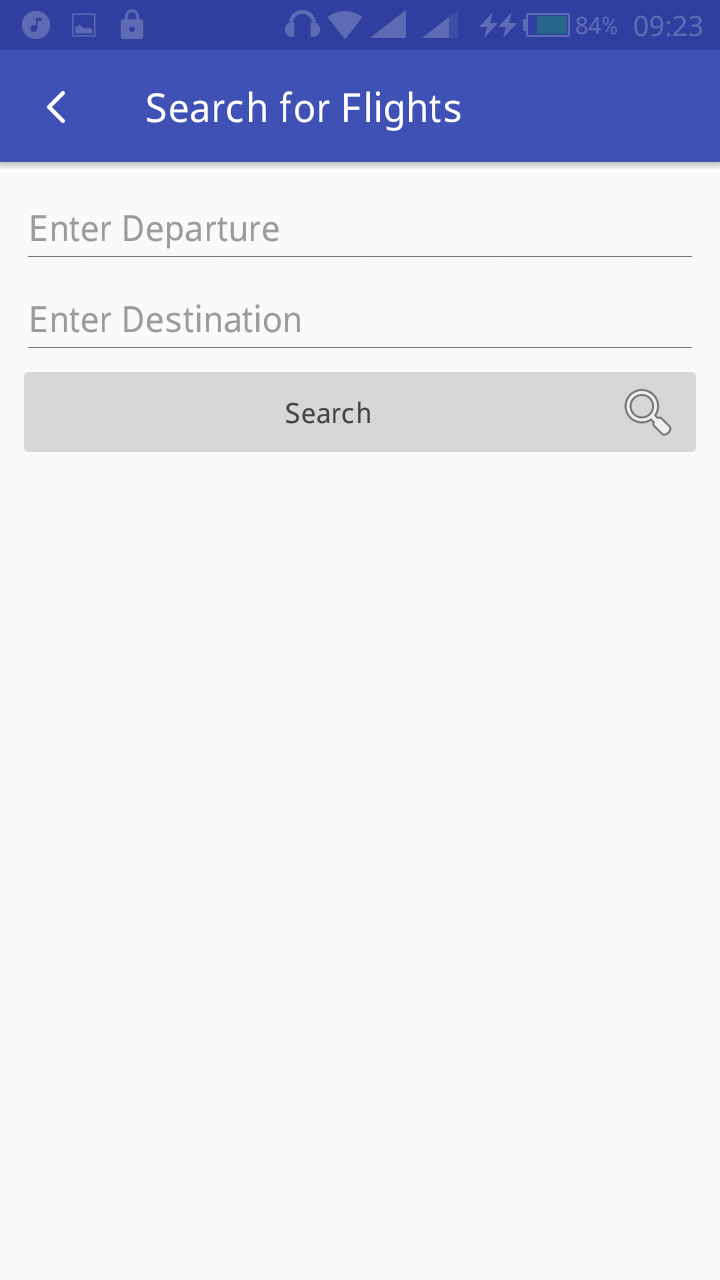
****

Figure 4.4 Searching for a flight

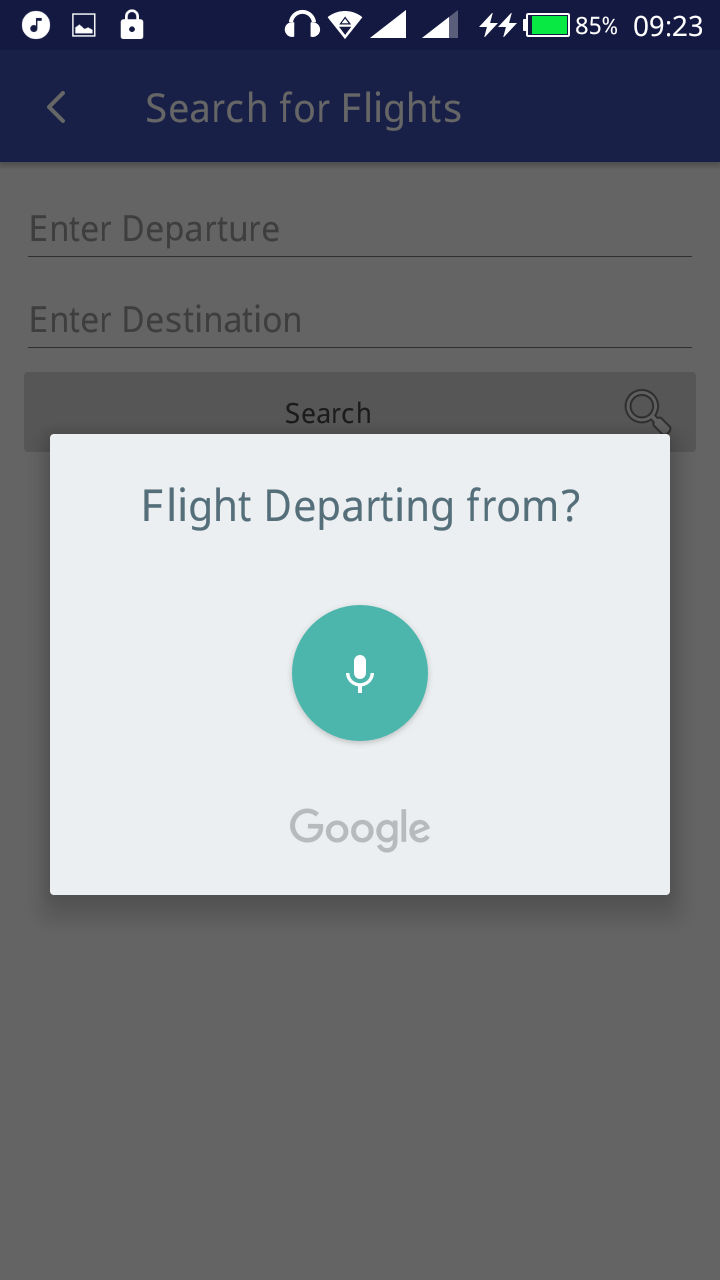
****

Figure 4.5 Using voice to input flight departure location

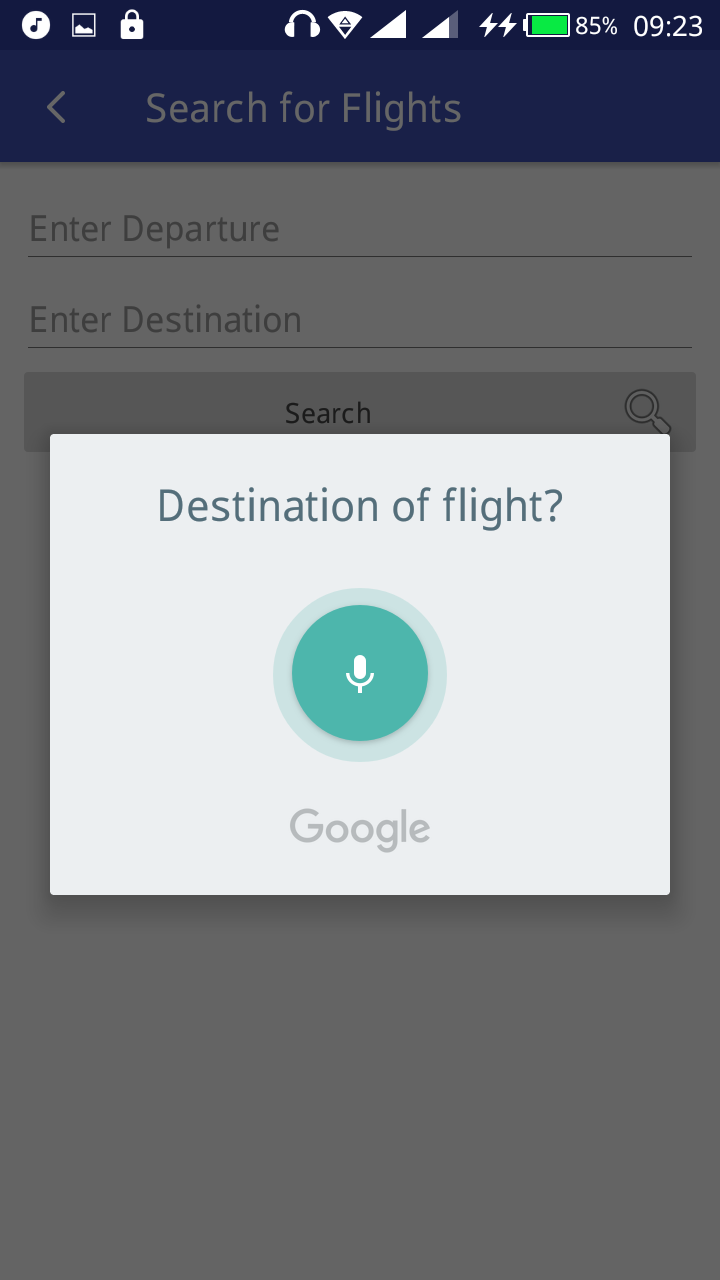
****

Figure 4.5 Using voice to input flight destination location

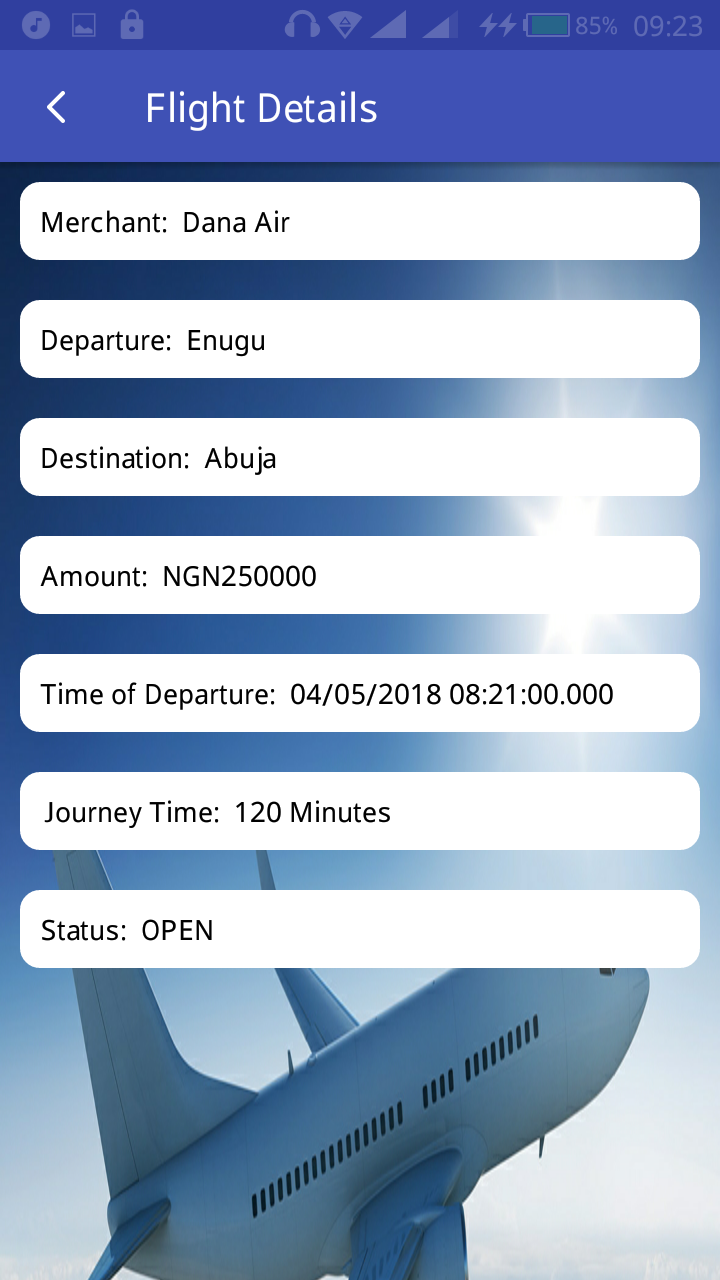
****

Figure 4.5 Showing flight details after a successful voice search

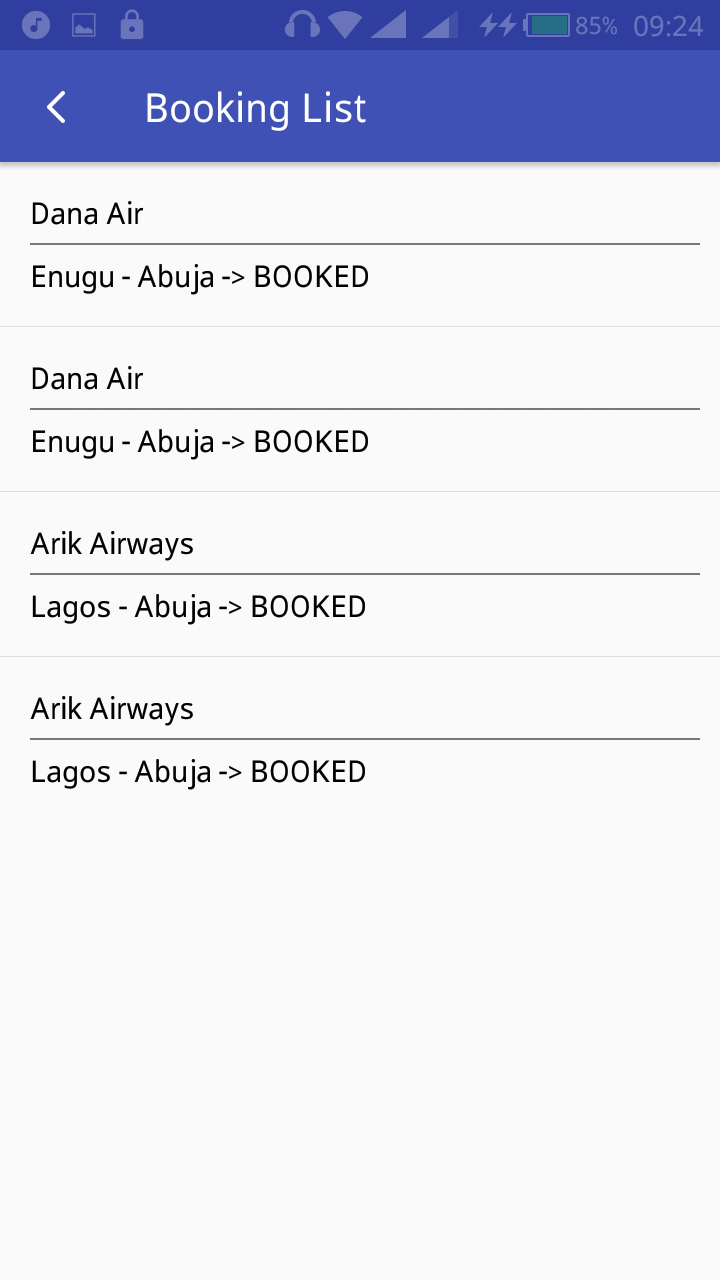
****

Figure 4.7 Showing booking list of a user

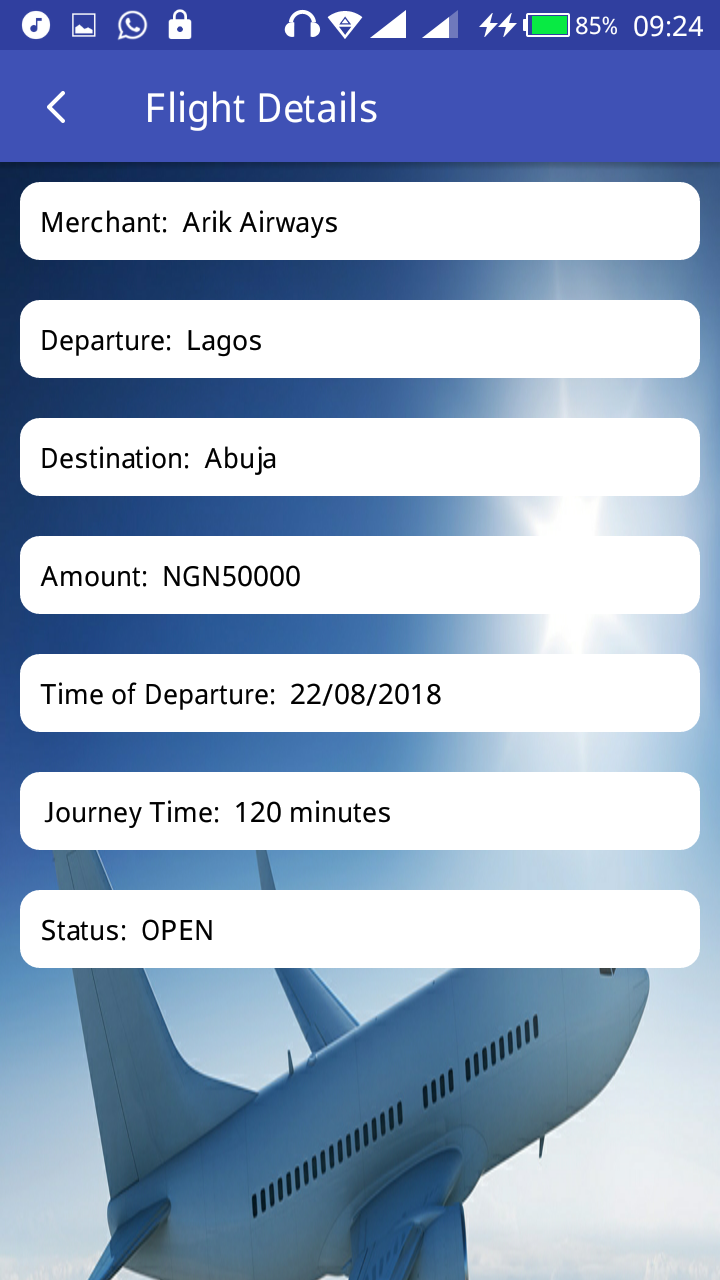
****

Figure 4.8 Viewing the details of one of the booked flights

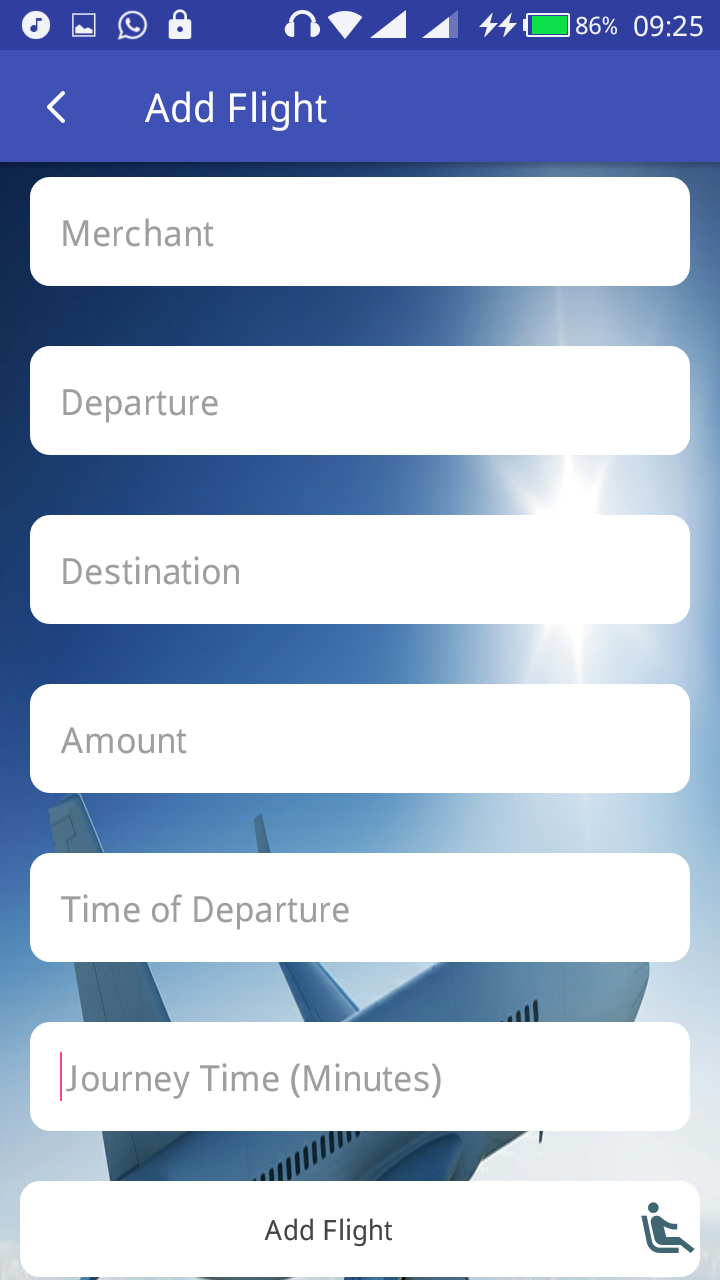
****

Figure 4.9 Adding a new flight by Admin

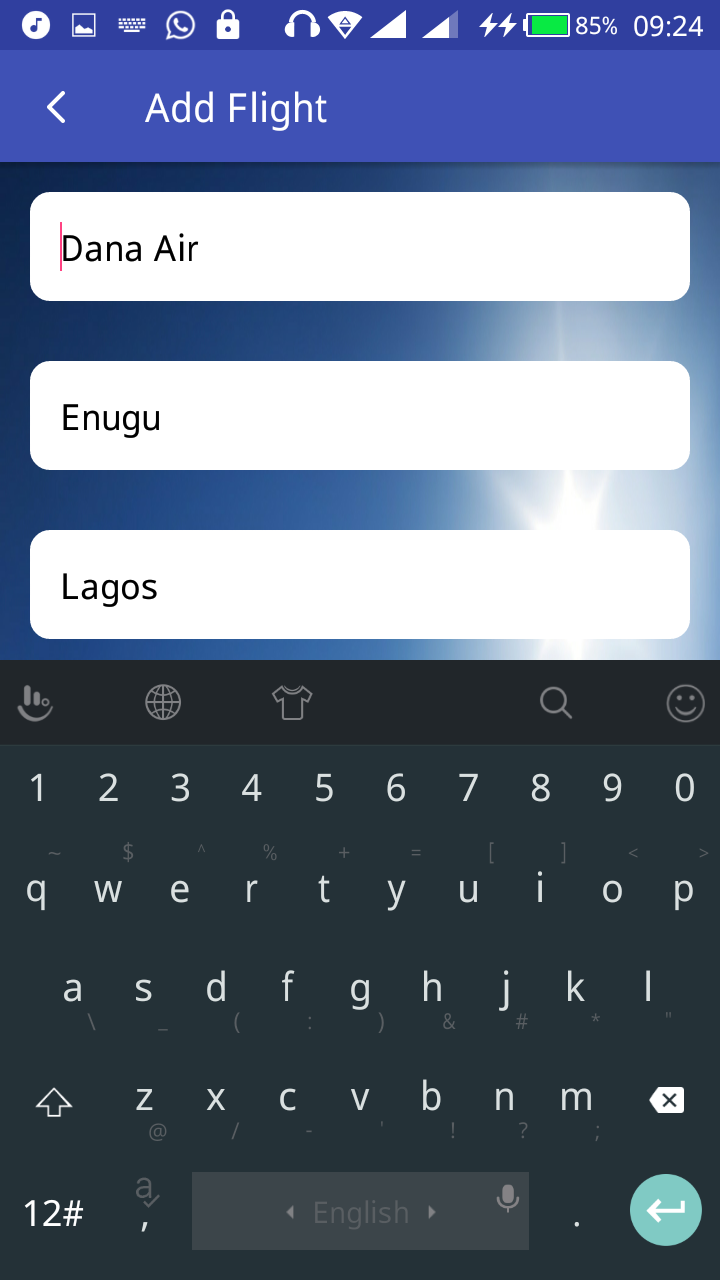
****

Figure 4.10 Admin adding a new flight

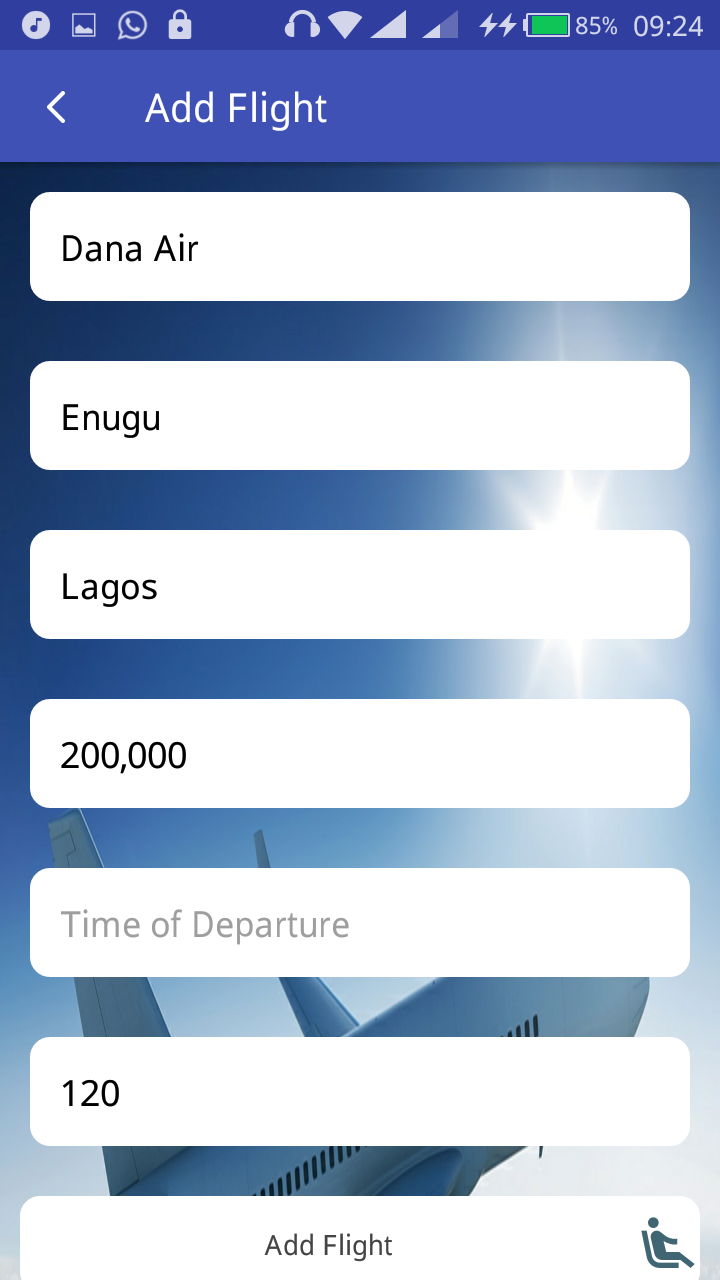
****

Figure 4.11 Admin adding a new flight

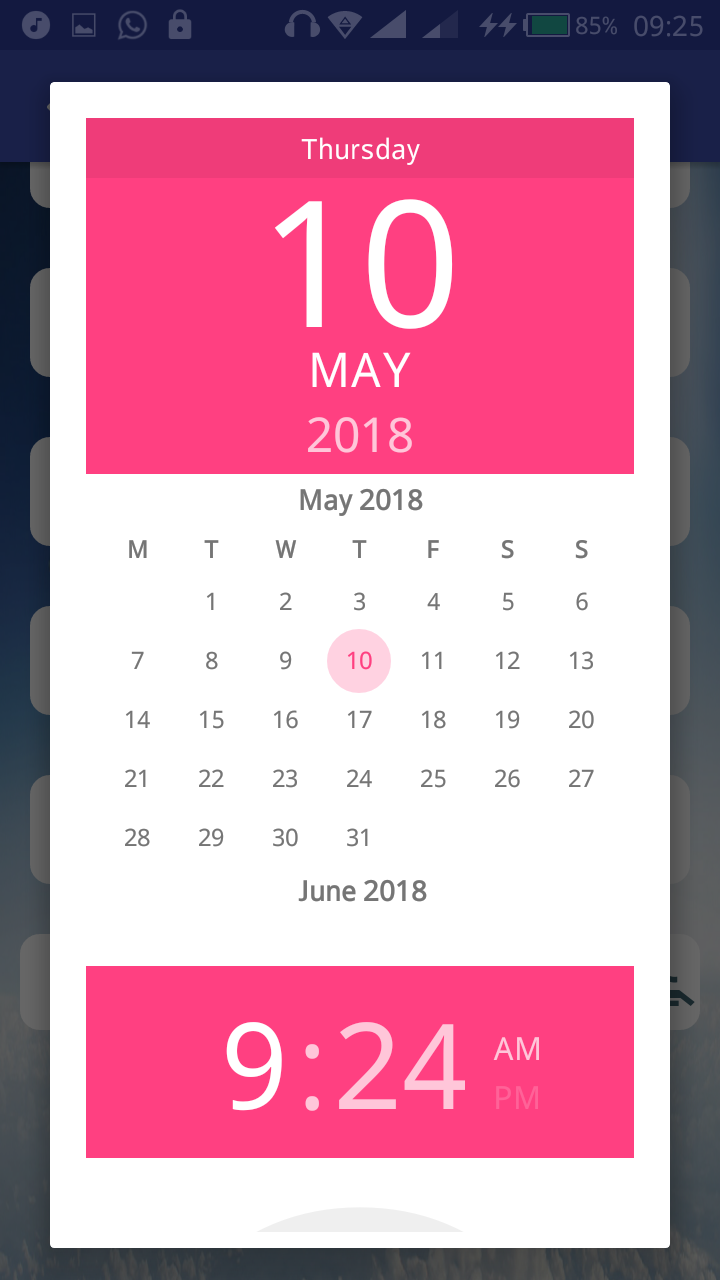
****

Figure 4.12 Admin selecting flight schedule

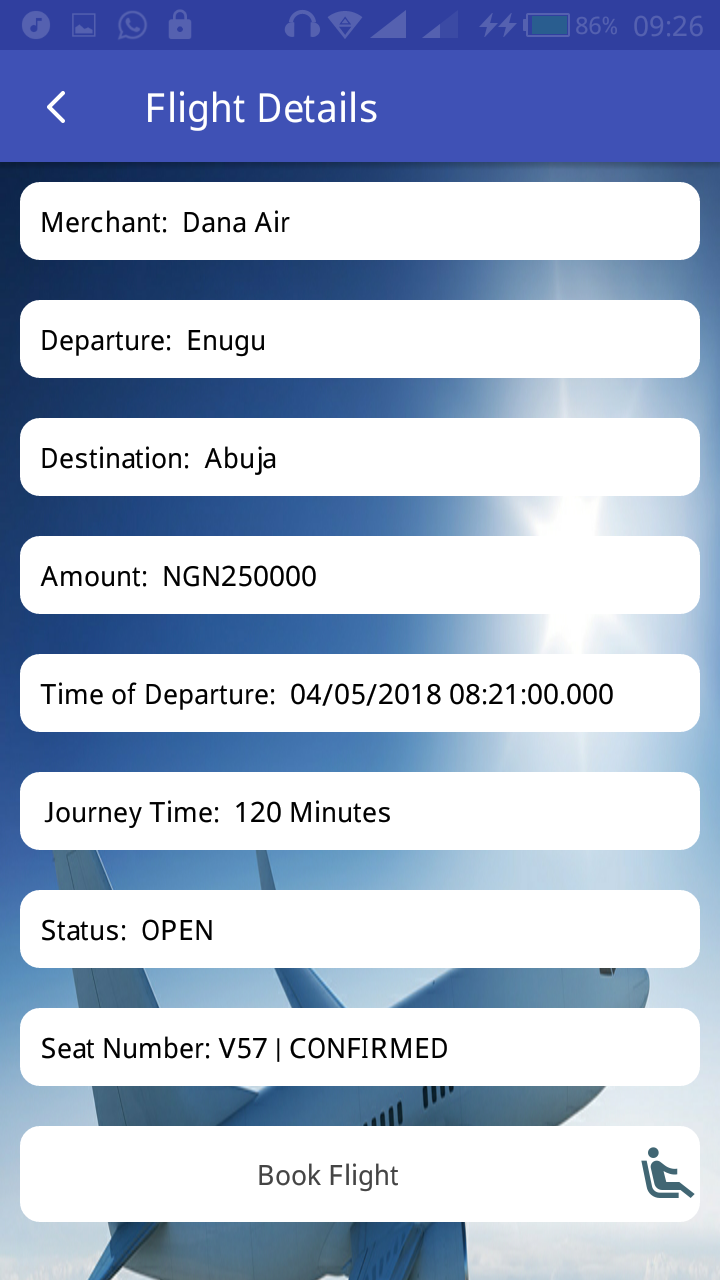
****

Figure 4.13 User’s booking confirmed

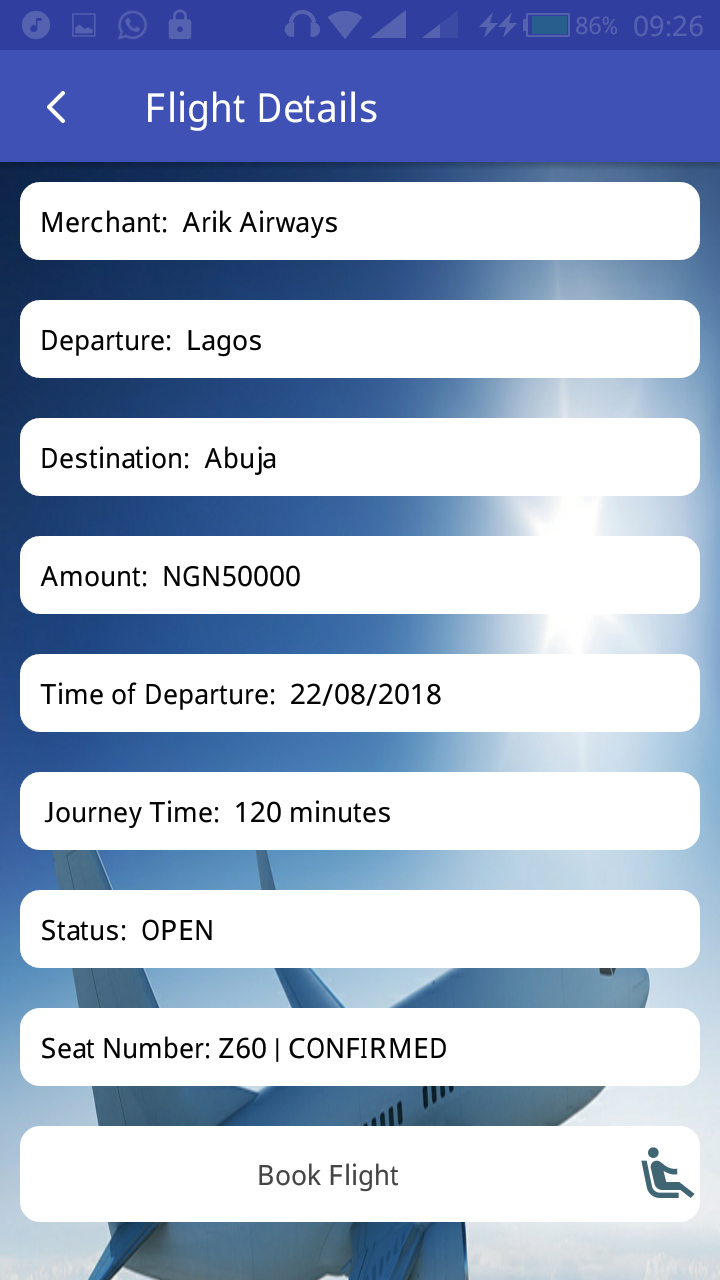
****

Figure 4.14 User booking another flight

**4.4 System Testing**

The accuracy of the program was tested with various data. This gives the assurance that the new system with achieve its purpose and objectives. The airline reservation mobile application was duly tested and certified working perfectly, no error report was given on any page of the mobile application. The system consists of pages stylishly linked together to deliver a good project management analyses experience to the users. It opens well on virtually all versions of android OS. All tabs were tested and certified okay, making sure none is broken, the system was also tested on various personal computers, 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.

**4.5 System Maintenance**

Having a mobile app has its responsibilities; proper and regular maintenance will make it run smoothly always. Updating of database is the top on the priorities. Then to the issue of device security, the app administrator that manages the backend must be watchful, to notice any malicious activity of code hackers, early discovery of hackers’ activities will aid in blocking them off by simply changing password of the device. Tabs get broken over time, so the tabs should be consistently checked using Java mobile tab checker function to test all tabs. Backing up the app database contents is another important maintenance task, this will make it impossible to lose data especially when the server (main computer) crashes or attacked by hackers. The system developed will stand the test of time if properly maintained.