ABSTRACT

Air travel is one of the most prominent and increasingly sought-after forms of transportation. Technological advances have helped in the evolution and advancement of air travel over the past decade. The airline industry is continuously flourishing and prospering, significantly due to customer satisfaction. The goal of this project is to enhance customer (flyer) satisfaction by providing valuable insights associated with air travel, within an android application.

In the android market, the most common apps belong to high profile airlines, which generally assist users in flight search, booking, mobile check-in and access to their respective reward/frequent flyer programs. This project aims to address the unconventional areas of customer service within one android app, comprising of three different modules. Each module will aim to provide a unique functionality, such as; providing a comprehensive view of all reward programs of a user in one view, seat maps of popular airlines to assist in flight search and gate maps of eminent airports and its terminals.

This app will aim to provide the user with an appealing and user-friendly interface. The app will be supported by an Oracle Express Edition 11g database server. Development environment includes, but is not limited to Eclipse, Java SDK and Android SDK.
# TABLE OF CONTENTS

Abstract .................................................................................................................. ii

Table of Contents ................................................................................................. iii

List of Figures ....................................................................................................... iv

1  Introduction and Background .............................................................................. 1
   1.1 Delta Air Lines, Inc. ................................................................................... 1
   1.2 American Airlines ...................................................................................... 2
   1.3 United Airlines ........................................................................................... 3
   1.4 Kayak ........................................................................................................... 4
   1.5 Flight Aware Flight Tracker ....................................................................... 6
   1.6 Skyscanner .................................................................................................. 7

2  Travel Wise ......................................................................................................... 8
   2.1 Module 1: Reward Programs ..................................................................... 8
   2.2 Module 2: Seat Maps ................................................................................ 8
   2.3 Module 3: Gate Maps ................................................................................ 9

3  System Design ................................................................................................... 10
   3.1 Development Requirements ...................................................................... 10
   3.2 Architectural Design .................................................................................. 11

4  Testing ............................................................................................................... 17
   4.1 Test Plan .................................................................................................... 17
      4.1.1 Test Strategies .................................................................................... 17
      4.1.2 Test Types .......................................................................................... 18
   4.2 Test Results ............................................................................................... 19

5  Conclusion and Future Work .......................................................................... 23

6  Annexure – 1: Test Case Example .................................................................... 24

7  Bibliography and References .......................................................................... 28
LIST OF FIGURES

1.1 Delta Airlines Android App........................................................................................................2
1.2 American Airlines Android App ..................................................................................................3
1.3 United Airlines Android App ........................................................................................................4
1.4.1 Kayak Android App for Flight Booking ...................................................................................5
1.4.2 Kayak Android App for Hotel Booking ..................................................................................5
1.5 Flight Aware Flight Tracker Android App ....................................................................................6
1.6 Skyscanner Android App for Flight Booking and Price Comparison .............................................7
3.2.1 Architectural Design ..................................................................................................................11
3.2.2 TravelWise Use-case Diagram .................................................................................................12
3.2.3 TravelWise Sequence Diagram for Registration/Login ..........................................................13
3.2.4 TravelWise Sequence Diagram for Reward Programs Module ..............................................14
3.2.5 TravelWise Sequence Diagram for Seat Map Module .............................................................15
3.2.6 TravelWise Sequence Diagram for Gate Map Module ............................................................16
4.2.1 Overview of screenshots after completion of test .................................................................20
4.2.2 Results of login page screenshots ..........................................................................................20
4.2.3 Stress test results ....................................................................................................................21
6.1 Testing results of Motorola Moto X ............................................................................................24
6.2 Before and after screens related to MonkeyExercise program execution ..................................25
1. INTRODUCTION AND BACKGROUND

Over the past few decades, mode of transport has changed on a rapid scale. Constant efforts have been made to decrease travel time while maintaining the comfort factor. Air travel is the obvious choice for satisfying any time constraints. With increasing interest in air travel, the need for technology to support one of the busiest forms of transportation has been intense. Technological advances have definitely benefitted the airline industry, which has been continuously expanding at a significant rate. One of the primary factors that helped air travel prosper is customer satisfaction.

Since the longevity of air travel is directly proportional to customer satisfaction, the airline industry strives towards maintaining its customers/travelers. Almost all airlines look forward to rewarding their customers with points/miles that can be earned on every flight ticket. These rewards are usually marketed in the form of reward programs, which are directed towards frequent travelers/flyers. Frequent flyer programs assist in promoting customer loyalty and have been used as a marketing tool by numerous airlines. While focus has always been to ensure customer satisfaction and loyalty, priority is also been given on engaging the customer with the airlines by facilitating them with mobile applications.

In the android market, the most popular and frequently used apps belong to high profile airlines and their frequent flyer programs. Some of them are illustrated below:

1.1 Delta Air Lines, Inc:

Airline: Delta [1]
Frequent Flyer Program: SkyMiles [1]
Fleet Size: 743[2]
Destinations: 247 [3]
1.2: American Air Lines:

Airline: American Airlines
Frequent Flyer Program: AAAdvantage
Fleet Size: 630
Destinations: 273 [4]
1.3: United Air Lines, Inc:

Airline: United Airlines
Frequent Flyer Program: MileagePlus
Fleet Size: 709
Destinations: 373
As seen in figures 1.1, 1.2 and 1.3 above, major high profile airlines generally assist users by providing the options of flight search, flight booking, mobile check-in and also access to their respective frequent flyer programs.

Other apps in the android market include those which focus on areas such as itinerary management, broader flight search/booking, flight tracking, hotel search/booking, car rental search/booking, etc. Some of these apps are illustrated below:
1.4: Kayak:

Figure 1.4.1: Kayak Android App for Flight Booking

Figure 1.4.2: Kayak Android App for Hotel Booking
1.5: Flight Aware Flight Tracker:

Figure 1.5: Flight Aware Flight Tracker Android App
1.6: Skyscanner:

Figure 1.6: Skyscanner Android App for Flight Booking with Price Comparison
2. TravelWise

This project is motivated by the need to enhance user experience and restrict travel hindrances faced by travelers. The objective of this travel app is to provide users with key insights and user oriented analytics involved with air travel and assist them in making an informed decision.

TravelWise aims to address the unconventional areas of customer service within one android app and achieves this through three different modules. Each module will aim to provide a unique functionality, such as; providing a comprehensive view of all reward programs of a user in one view, seat maps of popular airlines to assist in flight search and gate maps of eminent airports and its terminals. A user-friendly interface will stimulate user interaction and experience, while promoting user engagement.

2.1 Module 1: Reward Programs

The first module is dedicated to providing the user with analytics pertaining to users’ frequent flyer programs. This module presents users with a comprehensive view of frequent flyer programs they are enrolled in. Each reward program also presents the user with the number of miles earned (mile balance). This app also shows threshold levels for each reward program, i.e. the membership level/status of that particular program. Additionally, the app will also communicate with users when a threshold level has reached in the form of notifications.

2.2. Module 2: Seat Maps

The second module has been designed with the sole objective of providing users with an inside view of an airline and its seat analytics, to assist them in making an informed decision. An informed decision can possibly be flight booking, price comparison or personal preference. This module allows users to search for airlines and their carrier/fleet.
Upon selection of an aircraft, this module shows recommended and non-recommended seats to further aid users.

2.3 Module 3: Gate Maps

The third and final module aims to provide users with the option of searching for airports. After appropriate filtering, the search results display the airport that has been queried. Users will have the feature of viewing airport maps which will guide users in locating themselves on an airport. Furthermore, terminal maps will also be presented to the user for a more informed view.
3. SYSTEM DESIGN

This chapter discusses the components that make up the Travel Wise android travel app. This chapter is divided into two sections, namely development requirements and architectural design. The development environment and associated essentials are listed in the development requirements section. A brief architectural design is illustrated in the architectural design section, to demonstrate the work flow of the app.

3.1: Development Requirements:

This section provides information on hardware, software and database related development requirements.

**Hardware Requirements:**

- **Operating System:** Windows XP/Vista/7/8
- **RAM:** 512 Mb
- **Hard Disk Drive (HDD):** 40 Gb
- **Internet Connectivity:** >126 Kbps

**Software Requirements:**

- **Development Tool kit:** Eclipse IDE 3.4/3.5
  - JAVA SDK.6
  - Android SDK
- **IDT Plug-In (Eclipse):** Version 0.9.7
- **Programming Language:** JAVA
- **Database Server:** Oracle Express Edition 11g
- **Server Setup:** Apache Tomcat Version 6.x/7/x
3.2 Architectural Design:

This section presents an overview of the architectural design of the Travel Wise android app, which can be layered into four tiers. Figure 3.2.1 below, illustrates the architecture of this app.

The presentation tier includes Servlets. These servlets communicate with the business tier. The business tier consists of the Application Service classes, such as the Wrapper Classes and Transfer Objects. Application Service classes implement business logic. The Wrapper Classes act as a medium between the presentation and the business tiers. The Application Service classes also use Data Service classes to interact with the persistence tier. This tier consists of Java Database Connectivity (JDBC) with which classes from the business tier communicate by fetching and manipulating data. This tier also consists of the Oracle Express Edition 11g server.

The mobile client uses the android app in order to gain access to the presentation tier (servlet) over a HTTP connection, to display the response.
A use-case diagram for TravelWise is illustrated in figure 3.2.2 to represent how a user will interact with the app.

![TravelWise Use-case Diagram](image)

**Figure 3.2.2: TravelWise Use-case Diagram**

Sequence diagram will be illustrated to present process interaction within the TravelWise android application. The sequence diagrams for login, reward programs, seat maps and gate maps have been provided individually in figure 3.2.3, figure 3.2.4, figure 3.2.5 and figure 3.2.6 respectively.
Figure 3.2.3: TravelWise Sequence Diagram for Registration/Login
Figure 3.2.4: TravelWise Sequence Diagram for Reward Programs Module
Figure 3.2.5: TravelWise Sequence Diagram for Seat Map Module
Figure 1: TravelWise Sequence Diagram for Gate Map Module
4. TESTING

This chapter covers various tests that have been performed on the TravelWise app to ensure users use the app in a manner it was developed to work as. This section is further divided into two sections, namely test plan and test results.

4.1 Test Plan

The strategy was to challenge the app with numerous tests that would determine its quality. Testing has been performed using a combination of widely recognized strategies, namely crowdsourcing, physical devices, simulators and emulators. These strategies are described briefly below.

4.1.1 Test Strategies:

Crowdsourcing

The crowdsourcing technique involves companies that specialize in beta testing software and mobile applications. The principle benefit of such a strategy is quick access to a wide variety of devices without the need to invest in those devices. Detailed diagnostics are also provided depending on the company.

Example: Eleks, A1QA, TestObject, etc.

Simulators and Emulators

Simulators and emulators are provided by vendors, to allow activities such as selection of different screen sizes, models and operating systems. This strategy is highly beneficial since it provides a feel for the app or simply the user interface.

Example: Android emulator, Windows android emulator, Google emulator, etc.
Physical Devices

Physical device testing technique involves manual testing of the application on various devices. This kind of testing is important before an application is released to the user. Physical device testing helps detect any issues that an end-user may face, since it involves a real device.

Example: Samsung Galaxy SII, HTC One, Samsung Galaxy Tab, etc.

4.1.2 Test Types

To provide seamless user experience, the following tests were performed on the TravelWise app:

Hardware Compatibility Testing

This kind of testing took into account differing screen resolutions and varying support for features to ensure correct functionality on all target devices. The app was tested on numerous devices to verify compatibility and optimal performance.

User Interface Testing

UI testing took into account the varying user experience from device to device. In addition to emulator testing and device testing, user testing is important for thorough UI testing. Some uncovered aspects of the application such as ease of use and coloring formats can be provided by end-users.

Location and Network Service Testing

Wireless network, location services and data connectivity services have been tested to ensure reliability and provide a seamless user experience. Emulation and physical device testing has been performed to mimic conditions where data connections or location services may be unavailable.
Usability Testing

This type of testing includes text visibility, navigation between screens and verification of functionality of the app.

Performance/Stress Testing

This kind of testing was conducted to ensure that user input gets appropriate feedback/output from the app. The MonkeyExerciser was used to generate pseudo-random streams of user events such as clicks, touches or gestures.

4.2 Test Results

Screenshots

The TravelWise app was installed and tested on a number of devices with various screen resolutions and sizes. The test results would enable review of the layouts on each device.

The overview screen of screenshots in figure 4.2.1 shows the following metrics:

- Successfully passed devices
- Screenshots needing approval
- Failed screenshots
Figure 4.2.1: Overview of screenshots after completion of test

Figure 4.2.2 below shows specifics of devices on which tests have been performed. In this case, screenshots have been taken for the login screen on all devices. The devices shown in figure 4.2.2 are real-world devices which are connected to the internet.

Figure 4.2.2: Results of login page screenshots
The stress test results are included in figure 4.2.3. The MonkeyExerciser program generated random clicks and yielded the results seen in figure 4.2.3 below.

![Stress Test Results](image)

**Figure 4.2.3: Stress test results**

The above mentioned tests have been performed using real devices and google emulators. Real devices are android devices hosted in real data centers, connected to the internet via Wi-Fi. Google emulators are official emulators provided by Google which has ARM and X86 support and are OpenGL accelerated.

TravelWise has been tested on 26 unique devices. Some of the real devices that tests have been conducted on are:

- Asus Google Nexus 7, LG Optimus L3 II E430, Samsung Galaxy Note N7000, Samsung I8190 Galaxy S III mini, Samsung I9300 Galaxy S III, Samsung Galaxy Pocket S5300, Samsung Galaxy Y Duos S6102, Samsung Galaxy Note II N7100, Samsung I9000 Galaxy S, HTC Desire, etc.
Detailed testing specifics such as steps performed, scripts run and device logs are included in Annexure 1 at the end of the document.
5. CONCLUSION AND FUTURE WORK

This project realizes the need of a mobile application to facilitate growing complexities related to air travel. An android application named TravelWise has been developed to assist users and provide them with a user interactive interface.

TravelWise has good scope for scalability and growth. Future work/development that can be done to upgrade the project and challenge innovative areas are listed below:

1. Real-time update of miles upon successful completion of a flight
2. Addition of airports and their respective airport maps to the Gate Map module for scalability
3. Inclusion of restaurants, hotels and other airport amenities in the Gate Map module to increase user interaction
4. User location display on different views of airport maps in the Gate Map module
5. Flight booking and cost comparison functionalities
6. Users’ comments/feedback on individual seats of aircrafts in the Seat Map module.
7. Addition of airline fleet to Seat Maps module
8. iOS and web apps to complete the ecosystem

With constant innovation and technological advances, the air travel scenario is bound to change continuously and grow adversely. Several partnership opportunities are available for further expansion and development of the TravelWise android application. All these factors will definitely take the quality and customer satisfaction metrics of this travel app to an entirely different level.
TEST CASE EXAMPLE

Tests have been performed on numerous devices for quality assurance. This annexure provides detailed insight one such device - Motorola Moto X.

Figure 1 below shows a successfully completed test on Motorola Moto X, which included details of the device that the test was performed on and hardware specifications of the device.

![Automated Tests]

Replay #38

Started: April 30, 2014 - 11:47 pm  
Test: #3 Stress Test  
Suite: #4 Stress Test  
App Under Test: #1 1.0

Device

Motorola Moto X  
Android 4.3

Resolution: 720 x 1280 (hdpi)  
Screen size: 4.7"  
CPU: Intel | single core  
RAM: 2GB  
Internal storage: 16GB  
SD card: 1024 MB

Script

1 launchDefaultAppAndWait()

2 monkeyExercise({eventCount: 3000, throttle: 30, seed: 0})

Figure 6.1: Testing results of Motorola Moto X
`Replay#` simply denotes the device number in the list of devices tested. Other test-related details such as start date, test name, suite name and app name are also specified. Device details include device name, android version, resolution, screen size, processor, storage, etc. The scripts and their related methods used are specified under the script section. For further illustration, the two steps performed are shown in figure 2 below:

![Login Screenshots](image)

**Figure 6.2: Before and after screens related to MonkeyExercise program execution**

Apart from test specifics, device details and scripts, a log called as device log is maintained for all events that occur during testing on the device. The device log for this Motorola Moto X is attached below:

**Device Log**

```
```
START com.android.internal.os.RuntimeInit <<<<<<
3 2014-04-30 23:48:35.902 DEBUG: AndroidRuntime : CheckJNI is ON
4 2014-04-30 23:48:35.902 DEBUG: dalvikvm : Trying to load lib libjavacore.so 0x0
5 2014-04-30 23:48:35.912 DEBUG: dalvikvm : Added shared lib libjavacore.so 0x0
6 2014-04-30 23:48:35.912 DEBUG: dalvikvm : Trying to load lib libnativehelper.so 0x0
7 2014-04-30 23:48:35.912 DEBUG: dalvikvm : Added shared lib libnativehelper.so 0x0
com.android.commands.pm.Pm
9 2014-04-30 23:48:35.952 WARN: ActivityManager : No content provider found for
permission revoke:
file:///data/local/tmp/950e8d64-58b2-4960-b05a-08596d7aeba4542052619537366384.tmp
10 2014-04-30 23:48:35.962 WARN: ActivityManager : No content provider found for
permission revoke:
file:///data/local/tmp/950e8d64-58b2-4960-b05a-08596d7aeba4542052619537366384.tmp
/data/app-lib/vmdl-131643160
1047420 bytes
info.tabsswipe appid=10050 user=-1
free 7472K/10488K, paused 13ms+1ms, total 25ms
2195 2014-04-30 23:49:57.143 INFO: Input : injectKeyEvent: KeyEvent { action=ACTION_DOWN, keyCode=KEYCODE_BACK, scanCode=0, metaState=0, flags=0x0, repeatCount=0, eventTime=126411, downTime=126411, deviceId=-1, source=0x101 }
2196 2014-04-30 23:49:57.143 INFO: Input : injectKeyEvent: KeyEvent { action=ACTION_UP, keyCode=KEYCODE_BACK, scanCode=0, metaState=0, flags=0x0, repeatCount=0, eventTime=126411, downTime=126411, deviceId=-1, source=0x101 }
2199 2014-04-30 23:49:57.163 INFO: WindowManager : Screenshot Window{ad3aeb58 u0 org.testobject.android.toolkit/org.testobject.android.toolkit.HeartBeatActivity} was all black! mSurfaceLayer=21105
minLayer=21000 maxLayer=21105
2201 2014-04-30 23:49:57.173 DEBUG: dalvikvm : GC_CONCURRENT freed 92K, 17% free 462K/556K, paused 0ms+0ms, total 0ms
2203 2014-04-30 23:49:57.203 WARN: InputMethodManagerService : Starting input on non-focused client com.android.internal.view.IInputMethodClient$Stub$Proxy@ad7bdb18 (uid=10049 pid=2203)
(END OF LOG)
7. BIBLIOGRAPHY AND REFERENCES


