**TEST REPORT FOR MOBILE TEST AUTOMATION**

GROUP PROJECT: CMPE 287- SOFTWARE QUALITY ASSURANCE

**SUBMITTED BY:**

|  |  |
| --- | --- |
| **Ann Rose Cyriac** | **(010095267)** |
| **Apoorva Davu** | **(010027004)** |
| **Mounika Muga** | **(010763324)** |
| **Nagashree Hiremagalur Suresh** | **(010731942)** |
| **Priya Shah** | **(010114598)** |
| **Rohith Reddy Vootla** | **(010752014)** |
| **Sirichandana Gaddampally** | **(010316865)** |
| **Venkata Bharadwaj Senapathi** | **(010745605)** |

TEST REPORT

**TABLE OF CONTENTS**

1. INTRODUCTION 5

1.1 Purpose of Test Report 5

1.2 Application Overview 5

1.3 Scope of testing 5

2. TEST ENVIRONMENT AND TOOLS 6

2.1 Criteria for choosing the test tool 6

3. TEST STRATEGY 7

3.1 Test Approach 7

3.2 Preconditions 8

4. TEST SUMMARY 8

4.1 SIGN IN FEATURE 8

4.2 SEARCH FEATURE 9

4.3 PROMO CODE FEATURE 9

4.4 UI FEATURES 10

5. TEST ASSESSMENT 10

6. TEST INFERENCES AND SUGGESTIONS 11

7. CONCLUSION 11

8. APPENDIX 12

**TEST REPORT ON TEST AUTOMATION OF MOBILE APPLICATION - POSTMATES**

## INTRODUCTION

### Purpose of Test Report

This document is a test summary report of the automation testing for Postmates, a mobile application and elucidates the various aspects of tests conducted on the app’s functional and non-functional features.

### Application Overview

Postmates is a logistics company, which work via a mobile/web application, enabling the users to have anything delivered on-demand by the local couriers or available ‘postmates’ who can carry anything from any restaurant or store in a comparatively lesser time. It provides an API to the businesses to offer delivery. Postmates has android as well as IOS mobile application versions.

### Scope of testing

* **In scope**

The following functionalities of the application are in scope of the testing undergone:

|  |  |
| --- | --- |
| **TEST ITEMS** | **PRIORITY** |
| SignIn Functionality | High |
| Search Functionality | High |
| UI Elements (Buttons, Menu Items) | High |
| Promo code feature | High |

* **Out of scope**

The tests addressed below are out of the scope of the testing conducted:

* + User acceptance testing
  + Compatibility testing

## TEST ENVIRONMENT AND TOOLS

This section describes the tools and environment used for performing the automation testing.

|  |  |  |
| --- | --- | --- |
| **TOOLS** | **VERSION** | **PURPOSE** |
| Selenium Web Driver | 3.0.1 | To write automation scripts using appropriate driver |
| Java SDK | 1.7.0 | To run Java programs in the system |
| Eclipse IDE | 4.2 | IDE to code |
| TestNg | 6.9.12 | Run Tests |
| Android SDK | 5.1 | Create emulator for testing using AVM and support Appium |
| Appium | 1.5.3 | Automates mobile application |
| Application (apk) tested |  | com.postmates.android |

The tests on various features were carried out with the help of physical device or an emulator, created using the Android Virtual Device Manager, whose specifications are as follows:

|  |  |  |
| --- | --- | --- |
| **DEVICE** **SPECIFICATIONS** | | |
| **Physical Device** | Device Name | ZTE-Z958 |
| API Level | 22 |
| Android Version | 5.1 Lollipop |
| **Emulator** | AVD Name | Nexus5X |
| API Level | 22 |
| Android Version | 5.1 Lollipop |

### Criteria for choosing the test tool

After much deliberation, Appium was chosen to be the tool for conducting test automation on Postmates. Even though Robotium was one of the tools under consideration, the reasons mentioned below attributed to the selection:

* Appium is an open source tool which is readily available and standalone in the sense that it doesn’t mandate an SDK or an app recompilation
* It does not require an access to the source code or the libraries of the application tested which in turn would allow the testing of any application of choice whose is apk is available online.
* Appium can be used in conjunction with any of the testing framework available for example the Selenium Web Driver. This would aid the process of testing since the group is more familiar with Selenium Web Driver.
* The ease of setup has also contributed to selecting Appium as primary tool.
* The tests could be invoked from any of the IDEs such as Eclipse or IntelliJ IDEA which most of the users are comfortable with.
* Besides offering a wide set of programming language choice, the record and play feature of Appium is another highlight of the tool.

## TEST STRATEGY

### Test Approach

The test makes use of java client for Appium and Selenium Webdriver. The testing mainly focuses on the functional side of the elements on the application to be tested. An **object-oriented** approach is taken in the automation testing of the mobile application as in:

* The creation of objects would make the test script modular and reusable
* Java methods are used to define redundant actions hence providing reusability of code

A **data-driven** approach is made use of in testing the promo code feature which in turn takes the input from a .txt file. This increases the versatility of the testing which allows trying different combinations of inputs as required.

The purpose of automating UI testing is to reduce the manual effort for multiple test cases thereby saving lots of resources and time. Also, with UI testing a strong foundation to automation tool and interactive testing can be built. This includes very challenging situations like checking for the state of multiple UI elements for every action and validate if they are working accordingly. Since UI is most important part of the application through which user interacts, it is important to test it thoroughly, covering all the possible cases. Data- driven approach is used for UI testing too.

The Sign in and search features testing make use of the Webdriver protocol to enter the keywords and explore the test results.

### Preconditions

To run the automated test error free, the following points are to be kept in mind:

* Add Selenium and Appium jars in the build path before running the scripts in Eclipse
* An appropriate JAVA JDK is essential to run both Eclipse and the scripts. Very few versions of Eclipse support Java 1.8, which is the latest version available.
* TestNg plugin installation is essential to write a TestNg class in the Eclipse

## TEST SUMMARY

**Project name:** Automation testing report of the mobile application, Postmates

**System tested:** Postmates application on android platform

**Version Number:** 1.0

**Type of testing:** Automation Testing

**Testing method:** Black box

Following are the details of the various tests carried out on the Postmates application:

### SIGN IN FEATURE

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **TEST CASE PLANNED** | **EXPECTED OUTCOME** | **PREREQ.** | **PASS/ FAIL** | **SEVERITY** |
| Test for invalid email address | An error message will be displayed | Sign-in button works and user is redirected to sign-in page | PASS |  |
| To test for invalid password | Error message will be displayed | Sign-in button works and user is redirected to sign-in page | PASS |  |
| To test for valid email address and incorrect password entries | SignIn should fail and error message will be displayed | Sign-in button works and user is redirected to sign-in page | PASS |  |
| To test for valid email address and password entries | SignIn should be successful | Sign-in button works and user is redirected to sign-in page | PASS |  |

### SEARCH FEATURE

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **TEST CASE PLANNED** | **EXPECTED OUTCOME** | **PREREQ.** | **PASS/ FAIL** | **SEVERITY** |
| To test if search bar fetches results appropriately for invalid input | Appropriate message such as “No results” should be displayed for invalid input | User should have logged in to the application | PASS |  |
| To test if search bar fetches results appropriately for valid input | Appropriate search results such as restaurants with Sandwich should be displayed for Sandwich as an input | User should have logged in to the application | PASS |  |

### PROMO CODE FEATURE

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **TEST CASE PLANNED** | **EXPECTED OUTCOME** | **PREREQ.** | **PASS/ FAIL** | **SEVERITY** |
| To test entering invalid promo code input field in the payment page. | Entering a wrong promo code must display an error | User must have logged in to the application and the shopping cart should not be empty. | PASS |  |
| entering valid promo code input field in the payment page. | A correct promo code has to make changes to the total. | User must have logged in to the application and the shopping cart should not be empty. |  |  |

### UI FEATURES

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **TEST CASE PLANNED** | **EXPECTED OUTCOME** | **PREREQ.** | **PASS/ FAIL** | **SEVERITY** |
| Validating the presence of relevant UI elements such as search bar, tool bar, Menu bar, action bar | The user should be able to view the elements and access the elements to navigate to required page | The app should be installed and open | PASS |  |
| verify whether al the UI elements are present in the menu bar by clicking on the menu items and navigating through them | All the required menu items should be present and must be able to navigate to the respective menu item | User must be logged in | PASS |  |
| Testing the account info using the username and phone fields and validate if the UI elements are behaving accordingly with the user input | • User should be able to enter username  • User should be able to give phone number  • If data is not modified, then the save action should not be enabled  •If data is modified, save action should be enabled  •If strings or invalid input is given the app should prompt an error message | The user need to signed in first in order to enter the account info. | PASS |  |

## TEST ASSESSMENT

Out of all the test cases none of the tests failed. Following is the test assessment at a glance.

|  |  |
| --- | --- |
| Percentage of defects from the above test cases | 0 |
| No. of defects identified | 0 |
| No. of defects with High severity | 0 |
| No. of defects with Medium severity | 0 |
| No. of defects with Low severity | 0 |

## TEST INFERENCES AND SUGGESTIONS

The automation testing of the application has drawn the following suggestions which the tester think would contribute to the improvement of the application and user experience:

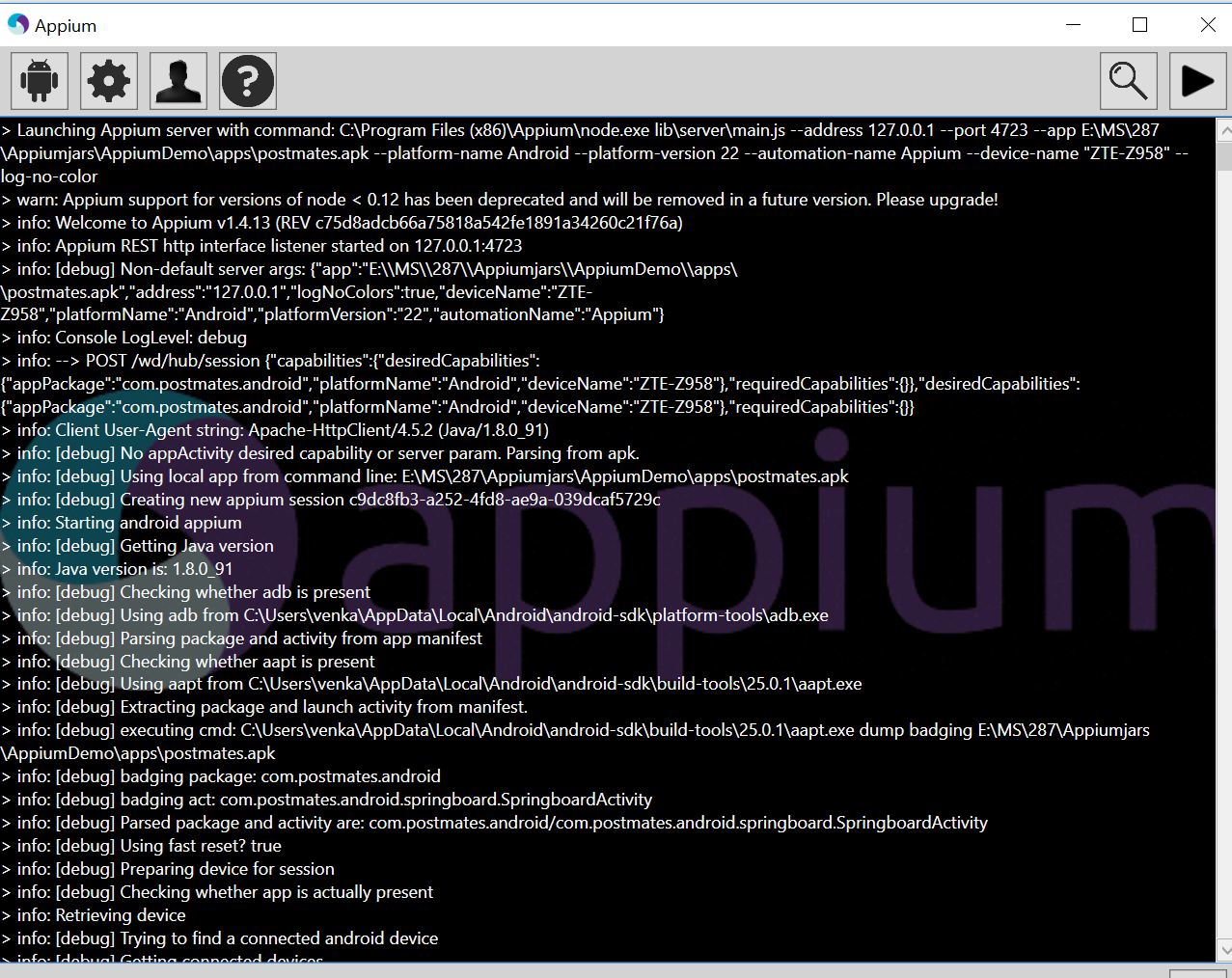
* The UI can be a bit more user friendly by arranging the available features in a convincing manner and making them self-explanatory.
* Navigation pattern on the application could be improved by including back buttons on screen and a home button on every page for redirection.
* The search for an item shows what is unavailable too which in real time is not mandatory

## CONCLUSION

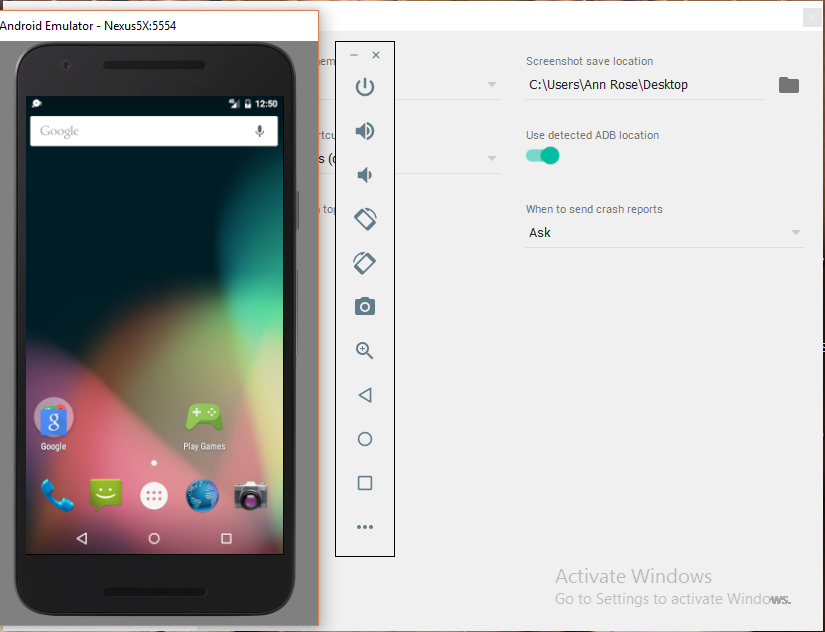
Automation testing was conducted for the mobile application, Postmates and a cent percent success rate was observed for the test cases designed. Suggestions for improvement of user experience, which are drawn from the test are enlisted in the Test Inferences and Suggestions section of this report. The tests mostly focused on the functional aspects of the application and were performed based on the test cases designed. Relevant test deliverables include the test report and the automation script for testing the application. In addition to the hands on experience in mobile test automation, the project facilitated the study and understanding on the concepts of automation testing, the need for proper planning and tool selection, Appium, Android Studio, Selenium Web driver and the use of emulator in testing.

## APPENDIX

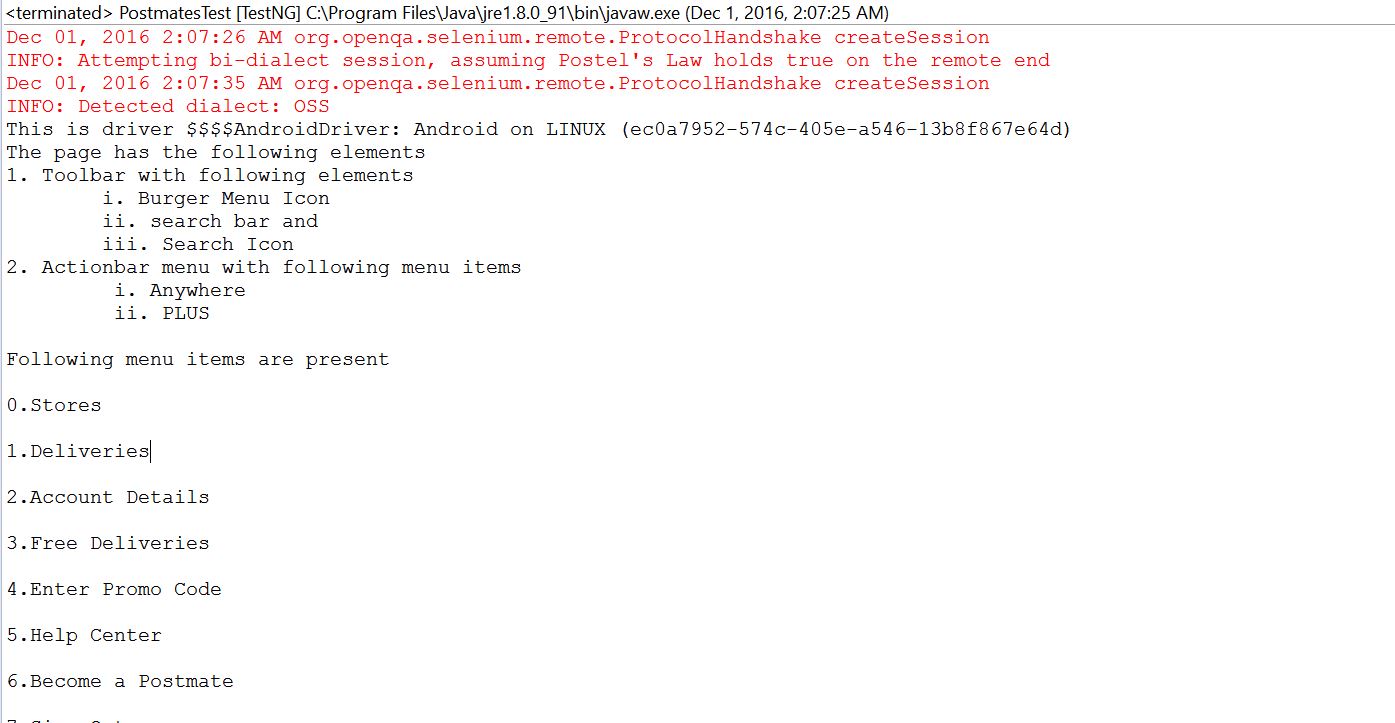
### APPIUM AS A TESTING TOOL



### SCREEN SHOT OF THE EMULATOR USED



### SCREEN SHOT OF CONSOLE OUTPUT FOR UI LEMENTS FEATURE TEST/



### SCREEN SHOT OF APPIUM INSPECTOR

