Team Final Destination

18 May, 2016

Semaka Malapane -
Matthew Nel -
Pierre Ngameni -
Ruth O. Ojo - 12042804
Sfiso Shabangu -

Contents

1 Test Plan Identifier

Final Destination: Testing for Alpha-Android

2 References

This document makes use of and refers to the Researcher Support System (RSS) Requirements and Design Specifications (version 0.1). This is the document that specifies what the entire system is supposed to do and specifies what each module is supposed to do.

3 Introduction

This is a Unit Test plan for the Research Support System (RSS) Android Application Framework (Android Module) and it will only address the RSS Android Module as implemented by team Alpha of the COS 301 mini project.
The RSS Android Module is being tested in isolation of all the other system modules, as such this test plan will cover one level of testing, namely Unit testing.
The purpose of this test plan is to document the testing requirements of the RSS Android Module as well as outline the testing procedure of the RSS Android Module. Only the functional and non-functional requirements will be considered. The basis of the requirements to be tested are in the RSS Master Requirements and Design Specifications (version 0.1).
The RSS Android module consists of the Android User Interface to the RSS, as such to test the functional requirements automated UI tests will be done. There are eleven non-functional requirements to be tested.

In addition to the above requirements, the Android Module will be assessed in line with the following criteria:

- Front-end process demo
- Usability
- Use of external mocks
- Gradle build with appropriate dependencies
- Use of Dependency Injection
- Use of Fragments
- Use of REST

Details on the testing of the aforementioned requirements and criteria will be given in the succeeding sections.

4 Test Items

The RSS Master Requirements and Design Specifications (version 0.1) did not give enough information on the expected functionality of the Android Module, as such assumptions will be made regarding the functional requirements of the module.

Assumed test items for the unit testing of the Android Module will include:

1. Functional Requirements
   (a) The Android UI should allow for access to all system modules that require user interaction.
   (b) Each link and button should work
   (c) All input boxes should allowed for input
   (d) User input should be validate with respect to the required input
   (e) The Android module should interact with a REST API to retrieve all necessary content that should be displayed on the UI.
   (f) Usability

5 Software Risk Issues

6 Features to be Tested

7 Features not to be Tested

8 Approach

8.1 Testing Level – Unit testing

In the given code, no testing constructs were found, as such the Final Destination testing team will carry out the necessary tests. Unit tests will be automated by the use of established Android UI testing frameworks. At this level of testing, mock objects will be required to ensure that the testing process can be carried out.

8.2 Test Tools

8.3 Meetings

8.4 Measures and Metrics

Metrics for the testing of non-functional requirements will be collected. These will be made available by the Lint Report tool.

8.5 Configuration Management

8.6 Special Requirements

The Android module will be assessed according to the given evaluation criteria.

1. Front-end process demo

   The UI of the module should be representative of an Android application UI and should conform to the set standards.

2. Usability

   With each click of a link or a button a user should be given appropriate feedback, in that a result of that click should be displayed or an appropriate error message should be thrown. The feedback should be timely as well.

3. Use of external mock objects

   As a result of a modular system design, the Android module should be able to fulfil all requirements in isolation. As such the use of mock objects is suited as a substitute for the real objects provided by the other system modules. REST API mock objects are needed for this module.

   The implementation will be checked for the use of a Mock REST API.

4. Gradle build with appropriate dependencies

   Dependency management of the module should be handled with Gradle to allow for seamless project build. All Android dependencies should be kept in the Gradle configuration.

5. Use of Dependency Injection

   As previously mentioned, the specification for the RSS requiress a modular design, ensuring for less inter-module dependencies. Each isolated module is meant to use Mock objects in place of concrete objects from the respective system modules. These mock objects are meant to be injected into the Android module.

6. Use of Fragments

   The

7. Use of REST

   The Android module should interact with a REST API in order to receive the necessary objects to be displayed on the UI. REST http requests should be made depending on the required data and a back-end server should respond with the appropriate data as long as no error occurs.

9 Item Pass/Fail Criteria

10 Suspension Criteria and Resumption Requirements

11 Test Deliverables

This section consists in enumerating with a brief description all what we have to deliver after testing the Android module which involves the USER INTERFACE OF THE RESEARCH SUPPORT SYSTEM (RSS). The deliverables of this test will be as follow:

- The Test plan document which is the document containing all the aspects of the testing; it describes the different activities that was done during the testing as well as the scope of testing , and the approach we had to take in testing the UI of the android module of the RSS.
- The Test cases: It is based on the functional and non-functional requirements. The functional requirement consists of using mock object of Junit to apply the unit testing; and Android lint report is used for the quality requirement (non-functional requirement).
- Tools and their output: Two tools are used in the testing process. Robolectric Android which is a framework for unit testing that uses the Android SDK jar in order to test-drive the android module and Android Lint Report for non-functional requirement.
- Errors logs and executions logs: are some screenshots of files that records some errors that occurs during the testing.
- Problem reports and corrective action: a corrective action was applied in the graddle file (file that manages dependencies in android studio) when opening the source code.

NB: the only thing that will not be part of the test deliverables is the Android module.

12 Remaining Test Tasks

the module we were given is not a multi-phase process and won't be released in an incremental way hence does not require additional attention in the sense that there will not be any future functionnality that will be added on the android module we have been given to test. If it was the case we would have answer affirmatively that there will be some remaining test but in our case there will not be any remaining test that will be carry on the android module because in our context this module is a stand alone application.

13 Environmental Needs

This section of environmental needs explains and give a description of the type some needs we will consider

In this section we look at some points that can appear to be environmental needs during the testing process and answer some of these environmental questions such as:

- Will the system (the android module) be restricted of use during the testing? since the android module(user interface) we are testing his a stand alone module and not integrated to the entire system. it is not therefore an issue to verify wether or not the system should be restricted of use. but in the meanwhile for testing purpose the android module should be live runing.
- As an environmental needs we needed to run the android code of the user interface on the latest version of Android Studio (Android Studio 2.1.1).
- We do not required any special power, but just the normal electric power that could make a laptop to be on.
- As another environmental need we had to look on how our the test data will be provide. in the testing process we will be using ock object to test against the functional requirement and we will also use input from the keyboard.
- In this project we are not using simulators(special hardware), either static generator
- The component of the android module are tested thoroughly as a whole.

14 Staffing and Training Needs

The android module is to be tested. Since most of our team members (3 out of 5) have never used programmed an android module, they would need to be trained and taught how to use android studio which would take up our already limited time. Due to the aforementioned reason, it was decided that it would be better if the members that are already proficient programmers in android should focus on testing the module and the other members focus on writing the test plan and assisting with test reports and non-functional requirements.

15 Responsibilities

16 Schedule

13 April: Have a look at the assignment specification and the code to be tested 14 - 18 April: Look up non-functional requirements testing tools and test plans 19 April: Meet up to discuss progress and findings. Divide the work according to our strengths and experience 20 April: Try to get the code running and see what the code does 21 April - 5 May: Assess the given code against requirement

specification. Start writing the test plan 6 May: Meet up to discuss progress 7-14 May: Continue writing test plan 13-15 May: Test the functional requirements and non-functional requirements 16-18 May: Finish writing test plan and report on results 18 May: Submit report

17 Planning Risks and Contingencies

Nowadays risks planification and contingencies are part of the planning of any project. In this particular project we had to consider some aspects which appear as risks such as:

- The availability of the human ressources, software, hardware and tools that we needed before carry on with the testing. We arranged all of these before the 10 May 2016 .
- In order to avoid late delivery, progress meeting was scheduled to know the level of each member and the difficulties he/she may have in the process.
- Avoiding delay due to training on a specific technology or tools: we made that each member should be assigned a task according to his level of knowledge and training and prerequisites.
- We do not have a case of contingency due to a change of requirements.

18 Approvals

This section states the parties who are responsable to approve the differents processes as fully done and give the go ahead for the continuation of the project. Parties involved in the approval are considered, depending on the magnitude of the project . In our case, we are applying the white box method, using mock Object of the JUnit framework; for testing the user interface module. Being a unit test it is different from other type of project such as integration; Therefore we only needed a group of software testers directed by a senior Software tester in charge of the whole design and plannification of the testing project and responsible of giving the approvals on each stage of the project.

19 Glossary

- RSS – Research Support System
- Android Application Framework – Android Module
- UI – User Interface

References