Test Plan

Author: Noah Zhang, Lucas Loaiza, Shang Wang, Siegfred Madeghe

1 Testing Strategy

1.1 Overall strategy

Software testing is a crucial part of building a software system. And in this project, the formulated test plan inspected the quality of the developed mobile application. The testing plan for this project includes three stages. The first stage is unit testing; the second stage is integration testing; the third stage is system testing.

1.1.1 Unit Test

Unit testing was the first stage in testing the app. In unit testing, the performed tests ran after writing the functions making the implemented classes. This method made sure that the units developed were robust and worked smoothly.

1.1.2 Integration Testing

Integration testing is the type of software testing that follows after performing unit testing. In integration testing, the components tested in the unit testing phase are then glued together and then inspected together as a group. This logical flow was used in implementing integration testing when closely related, tested components from the unit testing phase were grouped and then tested to perform certain tasks. The results were then analyzed to ensure that the observed results were correct and the codes bug-free.

1.1.3 System Testing

In this stage, the complete application was tested. The tests ran in this stage made sure that the basic requirements were met and that the application was running error free. The tests were performed using the emulator because none of the teammates had Android phones.

1.2 Test Selection

- The unit testing was performed using White Box Testing. The system was automated using testing tools.
- For the integration testing, Black Box Testing was mainly used. In some test cases, however, Gray Box testing was implemented.
- System testing: Black Box Testing was used. The testing was done manually using the emulator.

1.3 Adequacy Criterion

The main criteria used to determine whether a test passed or not was whether the outcome from the test matched what was expected--in this case, the required functionalities. This key criterion was the same for all test levels implemented.

1.4 Bug Tracking

In building the software for the developed application, the testing emphasis was on unit testing. Because it is easier to track bugs at the unit testing level; nevertheless, this does not mean that the test levels were not carefully considered. For example, the plan for bug tracking on integration testing was planned to be: Test, understand the bug, refer to the component-groups producing the bug, go further to the unit components in the groups, and then fix the bug. This way of tracking bugs simplified the process of spotting the sub-systems and the unit components that might have caused the bugs.

1.5 Technology

The key technology used for the testing was JUnit. Other tools, like Serenity, were used, too.

2 Test Cases

ID	Purpose	Steps	Expected Results	Actual Results	Pass/Fail	Comments
Test_01	To verify that clicking the ascending and descending button changes the ordering of the app names	Select the setting button and then change the setting under the ordering section accordingly.	Ordering of the apps should change according to the chosen manner of ordering--whether ascending or descending.	The apps order changed to alphabetical order in ascending when the chosen setting was ascending order and to descending when the chosen method was descending order	Pass	The test passed and met the requirements set forth by the testing criterion

Test_02	To verify that the chart showing the battery level changes according to the changes in the actual battery level	Use the emulator to change the battery level and observe how the graph changes	The graph should dynamically change when the battery level changes	When the battery level was manually changed using the simulator, the graph also changed. This was done on the XXX device on the Android phone	Pass	The test should be done on an actual android device to ensure that the functionalities are working just fine
Test_03	To verify that push notifications are sent to users when battery level drops below a	On the setting menu, enable notifications and then	When the battery level is below the threshold value, there should be a push notification sent to the user	The notification setting was enabled and threshold value set to 30%. When the	Pass	Push notification works according to the criterion set by the testing document

	certain threshold	set the threshold value		battery level dropped below 30%, a push notification was sent		
Test_04	To verify that the apps that use the battery can be listed accordingly	Open several applications and let them run on the background	The apps running on the background and thus consuming battery power should be listed by the application	The apps consuming the battery energy were not listed as expected	Fail	The test case failed. The integrated units that make the apps listing features should be debugged
Test_05	To verify that the apps once in an arranged order--alphabetically ascending or descending--can	Arrange the apps in either ascending or descending order	The apps will retain its original order once the app ordering feature is disabled	The apps retained the original order when the ordering feature was	Pass	The test passed, but, again, testing it on a real android application would be great

	retain its original form once the ascending feature is disabled	, then change the settings to disable the app ordering feature		disabled		
Test_06	To verify that once the push notification feature is disabled, then no notifications will be pushed to users even when the battery levels goes below a set threshold	Set the threshold values on the setting but do not enable the push notification feature	It is expected that no notification will be pushed once the feature is disabled even when the battery level goes below the set threshold	No push notification was sent to users even when the battery level went below the set threshold value	Pass	Expected and actual results matched. The feature does work