ECE 322
Progress Report

Arun Woosaree
Alex Rostron
Jacob Reckhard

November 19, 2019

1 The Code

The project we are testing is a diary like app, for tracking your moods. Users can add, delete, edit, and view their moods among other things.

2 What's been done

2.1 Unit Tests

Some unit tests were created in a white box testing manner. These tests were written using JUnit. `https://developer.android.com/training/testing/unit-testing/local-unit-tests`

Currently we have unit tests testing the constructor of MoodEvents, as well as several logical methods from converting database strings to moods. Most of the functionality of our app is displaying data from a database,testing how things are displayed is hard to do in unit tests. Additionally, the source code that was written, was not done so with testing in mind. Thus, our plans for high code coverage with unit tests is not on track. Instead, we look to have high statement coverage by the end of this project through a combination of unit and integration tests.

2.2 Integration Tests

Most of the functionality of our app is displaying data from a database,testing how things are displayed is hard to do in unit tests. So, for this we used integration tests. For the most part, a automated user interface testing tool called Espresso was used. `https://developer.android.com/training/testing/espresso`

Currently, we have tests for the app's functionality to add a mood event. We check each case of adding a mood event, making sure that an error dialog is shown if not enough information is shown, and testing the different optional inputs. For example, we have a test for just selecting the mood, another which does that and adds a comment. There are also tests for adding moods where the user's location is recorded, and where the test selects the number of people the user was hanging out with. These tests assert that the right data is shown on the main screen after entering the moods. For example, if the user entered 'HAPPY', that must be shown. Additionally, if the user selected 'ANGRY', and entered a comment, that comment should also be shown. If the location was entered, then NaN should not be shown as a coordinate. At the end of each test, the mood is deleted, so the delete functionality is also covered by these tests.

Another functionality our app has is filtering the UI to show only moods of a specific type. To test this, we have an integration test that does this. This test will create a 21 moods of various types then will return to the mood viewing screen. At this point it opens up the filter UI, and selects a filter state. It then compares the actual moods displayed to the number of moods it expected to be displayed. It will then repeat this 7 times will different filter states. It does make sure every single mood type is visible at least once.

2.3 User Acceptance Testing

The user acceptance testing took the form of a live demo. We have done a bunch of user acceptance testing. A large part of our manual testing has been in this form. The functionality for adding moods, deleting moods, editing moods, viewing moods, and filtering moods have all passed user acceptance testing.

3 What we plan to do

3.1 Unit tests

There is plans to write more unit tests for the small bit of actual logic in our application. However, since there is not a lot of logic going on, most of our tests are not unit tests. The places we plan on testing with unit tests are the getters, setters, and constructors of our classes. As well, as the functions that convert strings from the database into java objects.

3.2 Integration Tests

One thing that is still left to do here is to create tests for adding moods that have a photo, and checking if the photo is displayed properly. We also plan to make tests for following a friend, unfollowing, and viewing the moods of followed friends.

A very high condition coverage is our goal for this project. Even if that isn't achived through unit tests. We may also look into branch coverage of the User Interface flow. Overall we are slightly behind our original percent coverage goal due to the unforseen testability difficulties, however we feel on track to meeting our coverage goals. It is possible we may apply more black box methods of validation once the requirements of the moo-d project solidify further.