Testing

User Stories / Acceptance Criteria for Acceptance Test

- A new user should be able to create an account (signing up) to use our app
- An existing user can login and use the app
- A user can click / tap “Start Quest” in order to start the quest creation process
    - After hitting “Start Quest” the user should be able to name the quest they are about to create
    - After hitting enter on the naming screen the user should be able to move around, when they do this a trail will begin to be created behind their movements, the user is now creating a trail!
- A user can click / tap an existing quest to try and run that quest.
- A user should be able to stop doing a quest without actually finishing it.
- A user should be able to look at their statistics.

Robert:

I tested code related to running and completing trails using androidX and the JUnit framework. This covers the maps activity and fragment modules, as well as the viewmodel. The test creates a mock trail, named “Dummy Thick,” and attributed to an arbitrarily generated user id. It then calls the functions necessary to start running a trail and checks that the state of the viewmodel has updated appropriately. Then, it fakes a location broadcast by calling the broadcast receiver’s “onReceive()” method, with the broadcasted location being the endpoint of the generated trail. This should trigger the trail completion code and update the state of the viewmodel again, which we can verify by checking a couple flags.

Maverick:

I tested code using the emulator to send location updates to an emulated android device. The location service class and the maps fragment were able to be tested through Log files presented by android studio. Not technically a unit test, but was integral for functionality testing. Also performed tests to verify functionality of saving activity and fragment states. Provided debugging statements for lifecycle attributes so other team members can visualize the maps activity life cycle.

Eric:

I tested the Firebase user database and authentication by creating multiple users throughout each of our Sprints and making sure that sign ups and data was working properly. I

also continuously bounced through the UI navigation of our application to make sure no bugs arose.

Aaron:

I created unit tests that check that our firebase functionality is working properly. Using androidX and the JUnit framework I made tests that independently test features free from how a user would use our app. These tests can be run individually and they have all needed setup and cleanup built in.

My tests looks at the Firebase class and looks at the scenario of a new user signing-up to use our app. The tests follows these steps:

1. Test connects to our firebase database
2. Creates a new user to be added to the database
3. Adds the user
4. Verifies that the user created is now in our database (checks that the specific ID made in step 2 is now in the database)
5. Reports test results
6. Does cleanup (deletes user out of the database, etc).

Additionally I did a lot of "field" testing where I actually used the phone app and walked around to create / attempt quests. To do this testing I would focus on one function at a time and make sure it was working properly. Additionally I would have new users use the app to see if the design made sense and that the app was working on different phones. The feedback from these tests was very valuable and helped the project a lot.

Sean:

In terms of testing, I tested certain parts of the Firestore Database on Google Firebase. Ensuring that the database collected data correctly, I've tested the source code and Firestore to make sure that repeat users are not stored as new users on the database. In the scenario that an existing user creates a new account on the register page with the same pre-existing account credentials, i.e. valid email and password, then the system will not generate a new account nor log the user in. It will display the register as unsuccessful. On the database side, since the login credentials are already associated to a unique UID, no new data will populate within the users collection in Firestore.