**Developer List (name and computing IDs)**

Annette Chun (amc4sq)

Andrew Yang (ahy9ng)

**Device Name / Platform**

Spearow/Android

**Project Title**

Treasure Hunt

**Project Pitch (one paragraph describing what the app does)**

In our application, the user logins to their account and then walks around in the real world to collect treasures placed in different locations nearby. The treasures’ locations and the user’s current location, which is provided by the GPS system, are shown on the phone using a map. If the user is within a certain distance, the nearby treasure will be collectable. Once the user is close enough, the user must click on the treasure marker on the map to open the treasure. Afterwards, the user must shake the device to open the treasure, which will give them a point. Once a treasure is opened, the treasure will move to a different location nearby. The score is then updated to a leaderboard, which only showcases the top five highest scoring players.

**Platform Justification (a discussion as to why you chose the platform you did)**

Both of us had PCS instead of Macs, so we did not want to have to go to lab to work on the project. Also, we were both familiar with Java, but had to learn Swift for the first time for the iOS project, which meant that we would have less trouble with the syntax of the code in Java. We also had very slight Android experience before with CS 2110, which certainly helped us lean towards developing on Android instead of iOS.

**Key Features (include short descriptions of each)**

Our key features include our location detection feature using GPS, our shake detection feature, our login authentication system, and our leaderboard. The location detection feature pinpoints the user’s current location onto a Google map with the help of GPS. This location is used to see if the user is close enough to a treasure to open it. The user’s current location is updated every five seconds to accurately show the user’s changing position when they move. We also implemented a button using the location detection feature that the user may click to automatically center the screen back to the user’s current position. Our shake detection feature uses the accelerometer within the device to detect shake behavior from the user. The application uses this motion to open a treasure box. If a treasure marker is clicked on but the user fails to shake the treasure and instead presses the back button to return to the previous screen, the user will not receive a point for opening that treasure. That same treasure will still move to a different location even though the user did not receive a point for it. Login authentication is used to register a new user to our system with a unique username and a password that is at least five characters long. It also checks in already registered users, making sure they have the correct username and password. Everything is case-sensitive to increase the security of our system. If a user inputs an incorrect username or password, the system will not let them in and will notify them of the error. The login authentication system stores all information in a local database. Our final key feature is our leaderboard. The leaderboard lists the top five players’ usernames and scores with the highest-ranking scores. The top five users’ usernames and scores are retrieved from a web service we created. Every time a user’s score is updated, the new score is sent to the server’s database to maintain the leaderboard’s accuracy.

**Testing Methodologies (what did you do to test the app)**

To test the GPS functionality, we sent different GPS coordinates to the emulator to ensure that the GPS was accurately pinpointing the correct positions and that the map was showing these changes instantly. We changed the coordinates to be near treasures that were initially too far to confirm that treasures that are initially far can be open when the user eventually moves close enough. To test shake detection functionality, movement of the tablet was also emulated. We also tested it through the device in both orientations. After every shake test, the treasure was correctly opened and a point was given to the user each time. We tested the leaderboard by each making a new user separately and trying to garner as many points as possible to get onto the leaderboard, which we both were able to do even though we were on separate devices. We tested the overall flow and function of the application by taking turns starting from the beginning of the process and playing around with the application for long periods of time.

**Usage (including any special info we need to run the app (username/passwords, etc.))**

To start playing, the new user must first enter a username and password that is at least five characters long. Then the user must click on the register button to create an account. After this, the user may log in to the account on the same device from then on by inputting their username and password like before but instead, clicking on the sign in button. When walking around the map to collect the treasures, the user must click on the gray markers to try opening a treasure. There is no other special information required to run the application.

**Lessons Learned (at least half a page)**

For this application, we learned a lot more from this than we did from our initial android project earlier in the semester. We learned many different things for developing in android, especially with storing local data, sending GET and POST requests to a web service, using GPS and utilizing the accelerometer or shaking detection. There were many problems we faced at time and solving these problems helped us understand the lessons we learned in class more, and how Android lifecycle works better. For example, in the GPS portion, we learned that getting last known location would not give us a location for a long time so we had to set a default location until the current location was calculated. Also, we realized that for image resources in R.drawable, the image needed to have a transparent background, or it would just show up as a gray fuzz. We were not able to fix the image showing up as gray because colors were apparently too complicated. As for the web service, we learned that it takes a lot more effort implementing it than we thought. To post data to the web service, we merely needed to query the database for where we wanted to insert the data, determine what our inputs were, and submit the request. However, the get method proved to be trickier and took much more time. This was also because our web service displayed its data in JSON format. Thus, to retrieve the JSON data and read it, we had to utilize a JSONArray and use JSON methods, which were new and confusing to us. Also, many methods for retrieving data from a web service were deprecated, so we had a tough time finding and implementing the correct methods. Storing data locally did not prove to be as much of a challenge. However, it was still a new function that we had to learn. We utilized a SQLite database for our local storage. We did not reach that many problems with it; however, when we wanted to clear all the data from the database, we could not figure out how to do it. We tried to call a method within our application when we ran it to truncate the table to no avail. Finally, we realized that deleting the application on the device would automatically clear all data since the database was used for local storage on the device. Although it was good learning how to utilize local data storage functions, we determined that it would make more sense to store all data on an online database in order for users to login to their same account on different devices if they desired to. Overall, we learned a great deal and very useful functionalities that are crucial in building the foundation of a great application.

**Wireframe**

The wireframe is similar to the wireframe submitted for milestone 1 except that the prize, shop, logout and achievement screens were not implemented and the menu instead became a floating button pointing to a leaderboard.

