Maddox Scott

Mr. Pierson

IB Computer Science, Period 1

21 March 2022

Criterion E: Evaluation

Evidence of Final Product Implementation: See Appendix (B.6-7)

Success Criteria from Criterion A

1. The application must be able to read GPX activity files recorded from *Strava*, which the application will then be able to assign important statistics, such as route, date, duration, distance, and speed, which will be saved locally.
2. GPS routes that are imported should be visibly displayed in a map when requested, with tools such as being able to zoom in or out.
3. Users should be able to pick two imported activities, and the app will visually compare important statistics such as duration, distance, and average speed. Visual elements should be used to make comparing easier Appendix (B.1).
4. The application must prompt users to sign in when launching the app. User credentials must be saved locally, and activities imported must be linked to a specific user
5. The application must be able to display total, accumulated stats, such as “total distance biked” or “total time spent biking”. The app must also be able to view every imported GPS route in a single, cohesive map.
6. The application must have the ability to delete imported activities, and have the deleted activities not affect accumulated stats.
7. Implements an easy to use UI, with high-contrast fonts and visuals, and avoids small and light-gray text. Also implement simple animations for visual elements Appendix (B.1).

Evaluation of Success Criteria

1. Success - The application was able to read the route information from Christopher's rides, and the text entry system could record the activity statistics. Christopher stated that "the app can clearly get the routes from the strava files and map them. I could restart the app and data persisted so local saving worked as intended" (B.6).
2. Success - Activity routes were displayed in the Google maps widget. The user had navigational tools such as zooming in and out. "The google maps were great. I loved to see location markers linked to the roads I've biked on. Zooming worked as intended" (B.6).
3. Success - Two activities could be displayed in a single map. Their statistics, including distance, duration, and speed were compared below, with color coordination to signify which activity is superior. "It functioned just like I hoped it would, color-coding statistics let me see which ride was better just at a quick glance" (B.6).
4. Success - Christopher could make a new user, which all of his activities were linked to. Users could be saved locally, because Christopher "was able to log in with it after restarting the app" (B.6).
5. Success - Christopher could see all of his activities within a single, cohesive map. The sums of all activities properly displayed below. The accumulated stats page did not include deleted activities. According to Christopher, "being able to see all my rides in a single map was neat. The distance and duration totals also helped set all my biking into a proper perspective" (B.6).
6. Success - Christopher was able to delete the first activity he added, titled "Morning Ride". After Deletion, it did not affect the total stats page. My advisor stated " your delete

activity algorithm worked well, handling different contexts sufficiently" (Sean, Appendix(B.7)).

7. Success - All text was legible. Only high contrast colors were used. All Text was above 12px.

Recommendations for Future Development

Currently, the user must type in values for an Activity's distance and duration upon Activity creation. This meets success criteria 1, because the application uses text input to assign statistics to an Activity. While difficult, it is possible to circumvent user input, because these variables can be extracted directly from the GPX file. Much like the "constructPolyline()" method, a nested loop could be created to sum all the distances between each Track Point in order to calculate total distance. Furthermore, finding the difference between the initial timestamp and final timestamp would produce the ride's duration. Once such static methods are created, implementation would be simple, because the current Activity class allows for additional constructors. Methods for assigning and locally saving Activity statistics would still apply, so only acquiring the statistics would change.

Second, the addition of more data visualization would be a great addition to the app. Every Activity instance has getter methods to acquire their statistics, so charts or graphs could be easily made using them. The getter methods are accessible from anywhere, which allows for quick implementation.

Word Count: 493 Words