Project Submission Three

Group 7 - Sundance Stats

Dylan Docherty, Ryan Lebeau, Calvin Moras, Marcus Toldo

1.

a) **Execution Instructions:**

1. Download/Install node and npm
2. Load our project ‘sundancedestinystats’ into Visual Studios Code (or a similar program)
3. Run: npm install
4. Then run: npm run start
5. Select one of the http links to our web server to view

Alternatively

1. Complete steps 1 - 4
2. Cancel server with ctrl+c
3. Open ‘Index.html’ from package with web browser

b) **User Story Review:**

Our initial Sprint 1 was altered to ease the development milestones required for our first complete prototype. Its user story was altered to remove tracking performance and simply include the viewing of certain stats, primarily items. This was done because while it was originally believed that statistics and other metrics would be more simple. In reality, the displaying of a user’s items is a very basic procedure, and was a simple feature to code compared to some of the other sprints. Due to these reasons we elected to use item lookup and display as a first sprint goal.

c) **Design Review:**

Our initial sprint was modified a bit from the stage two submission after some serious research done by our group. We realized that the first thing to display would be the users character information such as clans, subclasses, and items. The entire design as a whole has not changed much, although displaying the user’s character information was not the primary focus and has been shifted to be more of the main goal. The app will still track a user’s statistics, but this can’t be implemented until the account information and items are displayed. In the future development of this app the first sprint will be based around obtaining and display the characters items and such, while the second sprint will focus more on displaying a user’s character statistics.

d) **Implementation Review:**

While some tweaks were made on a small scale, no overall changes were made to the technology stack. The simplicity of the loaded webpage is proof that a mobile version will easily be possible in the future, so we have no plans to remove this from any sprints. Our client-side has remained web-based as intended so the development process still used the same languages: HTML, CSS, JavaScript and TypeScript. Although we never reached a late enough stage of development to require a full-scale server-side component to this application, we did have a rough proof-of-concept as part of our development process. This was achieved by maintaining the Bungie manifest on our local machine, which was also running the code to display the webpage, thereby acting as a pseudo-server. The Bungie database manifest remained a key component in the development requirements as all hash lookups required for displaying proper content on webpages had to be done through it.

e) **Team Velocity:**

Since there was no direct user-story from the original project that corresponded to what we wished to achieve for our first sprint, our velocity estimations were unclear, just that they were lower than what was originally expected for the original sprint 1. The original estimated time to complete this sprint was well over two weeks, primarily due API research and feature coding. After changing sprint goals, the development time shrunk to less than half a week condensed; team velocity was estimated for approximately half of the original velocity, at around one week, the final result being around a quarter of the original estimation. This was likely due to both API research taking far less time than expected, as well as item based system being less complicated to code compared with the statistics system.