Kevin Banze

SCS 301

November 27, 2020

I want to make an app that displays the rank and statistics of another player in the game League of Legends. League is a multiplayer online battle arena which consists of 2 teams, 5 players on each team. The goal of the game is to destroy the opposing teams nexus, which is essentially home base. There are champions (characters) that you pick to play, items that make your character stronger, neutral objectives like dragon and baron that give your team a boost to certain statistics depending on which on they take. I want this app to mainly be a function in the pregame lobby. When you connect to a lobby, I want the app to display the names of the people on your team as well as the other team, with their ranks, top played champions, win rate, and build paths (items the player usually buys.)

For the program management plan, I plan on spending A LOT of time meeting with the team, detailing what exactly needs to be in this app. One big part of this would be analyzing the API needed to allow this app to work. In my opinion, we would need to have an API which connects to a third-party website which sends all of the data retrieved on the player back to our app which can then display it. The ability to map out this API and which third party site we will use should set us up for success in the rollout of our app. When errors occur, I believe this is where we will have the most. Build paths are going to be pretty difficult to stay on top of as the most opportune builds change every new patch that is released and also change depending on who the enemy team is. On the first day of the patch, we would need to have a data analytics team go over what has the best win rate for items and so on.

The requirements document is a separate document titled FINAL requirements.

Use cases: Linking account, choosing different preferences, importing runes, importing runes more than once, search functionality, improper closure of app, using a different account.

Linking Account Use Case:

Player will need to link their riot games account to our app.

Failure to do so will

1. Not allow the app to work, it needs an account to link to
2. Link to the correct account. If a player has more than one account, they will need to sign into the account currently be used. Failure to do so will not bring up their info.
3. When a player is in game, the API will communicate with the third-party website sending back the info on the player’s game.
4. Upon selecting a champion, the user can select to import runes (think of runes as items that your champion comes prebuilt before the game starts) based on highest win rate of the top ranked players of the champ

UML of use case in a separate document titled FINAL use case UML.

Database UML in separate document titles FINAL database UML.

Layout in separate document titled FINAL layout.

I will be using an Object-Relational database management system as it would make the most sense for how the data will be used and transformed. The app will consist of about 2 classes at heart. User and tempuser. These classes work simultaneously and the whole app functions through these apps. User is the actual “Person” who has an account with our app. This class includes customizing your profile to match how you play or how you want to play. Tempuser is the class that all other players in your game will have if the player doesn’t have an account with us. We assign them the base data we need to fill the class and present the data with no added features. This allows the player using the app to still check who they are playing against, what champion they play, what role they play, etc.

For any special occurrences, I think I have one big one about the API. The API will need to be very quick and always be able to function with the classes or the app breaks. This would mean the API would have to be updated as the third-party website we access it from is updated to make sure we still are going to the right place to get the data. Failure to do so crashes the app by either not having the correct data to be pulled, or overloading with too much data causing the app to be slow and not function. Websites get updated all the time and the areas where certain classes, data, or variables are sometimes move with these updates. Best case scenario, we could get someone from the API we are using as part of our team. This would cost extra money but be necessary in the longevity of our app. This person could keep us in the loop about the updates and the specifications behind them, and how we might need to change how the API works with our app to have continue success with our player base.