Project Proposal

CS 3340

Connor Robertson, Evan Snyder, Jacob Digman, Dylan Bahl

Project Title: PowerSpike

Summary: For this project we will be making a tool for players of the game League of Legends which utilizes the developer’s APIs to track game data and contextualizes it to help the player see which areas they are strong in, and which areas they need to improve. The data that is accessed from the APIs is stored in a database to ensure that data can be tracked over long periods of time to show long term improvement. The user can create custom settings to track and contextualize only specific data that is relevant to their play, and these settings are stored to a separate database to save them between uses of the tool.

Required Object Oriented Concepts:

1. Inheritance – inheritance is demonstrated with the “Summoner” class, which is an interface that represents a generic player of the game. It’s three children are “Player” which is the main user, “Opponent” which is a specific opponent that the player is playing against. This child is mostly identical to “Player” except that all of its data is encrypted so that it cannot be saved. The third child is “Average Player” whose data is the average of x number of players taken from a specific rank, champion, etc.
2. Polymorphism – polymorphism is demonstrated in the “ApiController” class, where its requestFromApi method will have multiple implementations based on what specifically is being requested.
3. Abstract Class – The “DatabaseAccessor” class is an example of an abstract class, where it’s functionality for accessing the settings database is implemented abstractly, and its child class will keep that functionality, and add other functionality to access the Stats database. (This is also an example of polymorphism)

Required GUI Concepts:

1. Menu bar will be used to navigate through tabs (player profile, match history, stats, compare, settings, etc.)
2. There will be lots of buttons
3. Dropdown list to select which game server is being accessed
4. Radio Buttons to select which data the user wants tracked
5. Use controls of radio buttons to determine which are checked/not checked

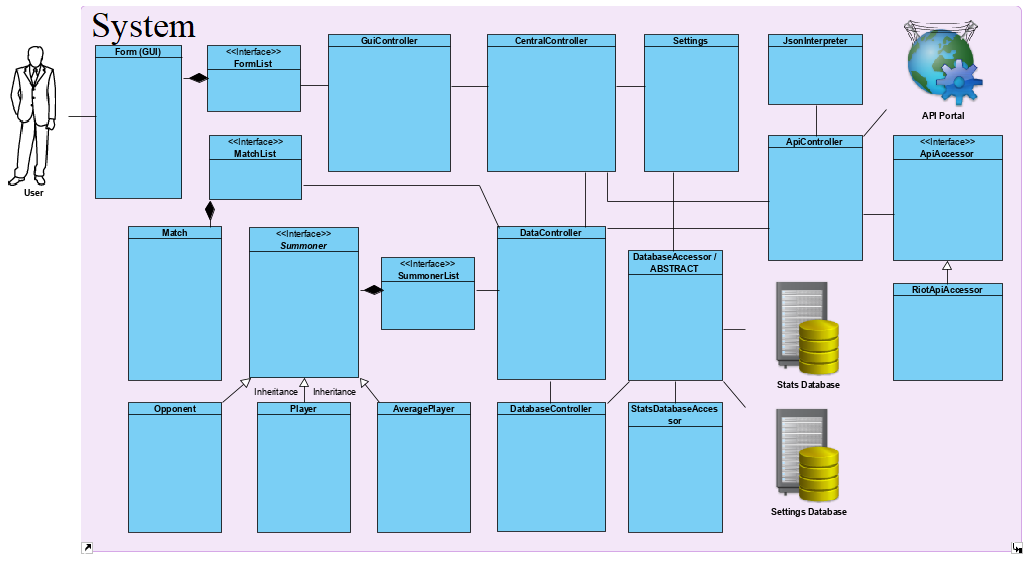
Required Database Concepts:

1. Settings database, Stats database
2. Unique settings for each game roll/champion table, generic system settings table. Match stats table, Player stats table.
3. Insert data to database upon receiving it from the API, done in DatabaseAccessor class
4. Delete data from database if requested, after x amount of time, or if corrupted, don’t in DatabaseAccessor class
5. Update button on Database tab of GUI, done in DatabaseAccessor class
6. Logic formulated in DatabaseController class and actual action done in DatabaseAccessor class

Future DLL Requirements:

1. Will restructure ApiController and JSON interpreter classes into Library. May do the same for Database classes instead/as well

Class Diagram:



ER Diagram:

A screenshot of a cell phone

Description automatically generated

Better versions of diagrams available on Github or upon request: <https://github.com/Robertsoncon/PowerSpike/tree/Class-Project>