Software Development Plan

Ballpark Bookie

4.1 Plan Introduction

This software design document provides the details of the planned design for Ballpark Bookie which provides an application to provide users with general and specific predictions about Major League Baseball games. It will have a simple design that will prominently feature two ways in which a user can interact with the application.

The application’s final output will be the predicted winner of an upcoming Major League Baseball game. The user will be able to search through a list of all the MLB teams and then select two to go head to head in our regression algorithm to determine a winner. The application will provide this information through a series of steps performed in the backend. Also, on another page, there will be a big predictions section that allows for the user to see the likelihood that a selected team will make it to the playoffs or world series.

On the left of both of these sections there will be a list of all 30 MLB teams with links to their schedules.

4.1.1 Project Deliverables

Our team has access to all the data from the mysportsfeed.com API. There are detailed stats for all major American sports and we will use their baseball data. The user will get access to accurate predictions on MLB games.

The main function of this project gives us the percent chance that one team wins over another team. We are working on updating this to be more accurate by taking in more statistics. The improved algorithm should be in its second form before the start of this upcoming 2020 season on 3/28.

After we have our improved algorithm will test it among past and current games and see if it needs more attention. By 4/12 we should know if it needs to be more accurate or if we have met our goal of 63% - 65%.

Stretch goals include implementing a way to more accurately take into account whether players are hit / pitch right or left handed. This is the first stretch goal that we will approach because it is beneficial whether or not our accuracy goal has been met.

Simultaneously, progress will be happening on the front end to look better and this will continue until the code freeze.

4.2 Project Resources

Ballpark Bookie requires very few resources to build and operate.

4.2.1 Hardware Resources

|  |  |  |
| --- | --- | --- |
| Table 1: Hardware Resources | | |
| Name | Specifications | |
| MacBook Pro (15-inch, 2019) | Processor: 2.3 GHz Intel Core i9, Memory: 16 GB 2400 MHz DDR4, Graphics: Radeon Pro 560X 4 GB Intel UHD Graphics 630 1536 MB | |
| Razor Blade Stealth 2015 | Intel® Core™ i7-6500U CPU @ 2.50GHz × 4, Intel® HD Graphics 520 (Skylake GT2), 8GB RAM | |

4.2.2 Software Resources

|  |  |  |
| --- | --- | --- |
| Table 2: Software Resources | | |
| Name | Specifications | |
| Visual Studio Code | Version: 1.38 | |
| Node.js | Version: 10.16.3 LTS | |
| Terminal | Version: 10.14.6 | |
| React-Native | Version: 0.59.8 | |
| Pop!\_OS | Version: 19.04 | |
| MacOS | Version: 10.14.6 | |
| Linux OS | Version: 16.04 | |
| React | Version: 16.8.3 | |
| Command Prompt | Version: 16.04 | |
| Eslint | Version: 6.5.1 | |
| Prettier | Version: 1.18.2 | |

4.3 Project Resources

* Front-End:
  + Derham/Persily: Designing the user-interface of the application in an easy-to-understand/operate fashion. Creating the relevant pages of the web-app to support the various functions and/or features associated with the application. Implementing user-input sections that will transmit user requests to the backend so the relevant results can be calculated and displayed. Taking results of the underlying algorithm and displaying them to the user in a clear and concise way on the page. Gathering feedback relating to the ease-of-use of the web-app’s UX and implementing necessary changes.
* Back-End:
  + Ochsner/Braekman/Arteaga: Developing the algorithm necessary to predict the outcome of any given MLB game. Using appropriate APIs to gather the necessary data/statistics that will be used in the aforementioned algorithm. Maintaining the collected data in a manageable way that can be easily accessed and reduces redundancy/unnecessary extra requests to APIs. Fine-tuning the algorithm to produce the most accurate predictions by determining which statistics are most relevant to the outcomes of games and assigning appropriate weights to those statistics. Sending the calculated predictions to the front-end to be displayed to the user.

4.4 Project Schedule

This section provides schedule information for the Ballpark Bookie project.

4.4.1 PERT / GANTT Chart

See 402 Gantt Chart in Documentation

4.4.2 Task / Reource Table

|  |  |  |
| --- | --- | --- |
| **Task** | **People** | **Resources** |
| Function to get Player Stats | Kea, Thomas, Riley | MySportsFeed API  Python 3 |
| Function to get Game Schedule | Kea, Thomas, Brett | MySportsFeed API  Python 3 |
| Function to call Lineups | Thomas, Andrew, Riley | MySportsFeed API  Python 3 |
| Store API data | Andrew, Brett | Python 3  pickle.py |
| Create winList Function | Kea, Thomas, Andrew | MySportsFeed API  Python 3 |
| Convert stats to float and remove extra | Kea, Thomas, Andrew | Python 3 |
| Calculate Difference Between Home and Away | Kea, Riley | Python 3 |
| Create Venue Win % function | Kea, Thomas, Andrew, Riley, Brett | MySportsFeed API  Python 3 |
| Create Average Player function | Kea, Thomas | MySportsFeed API  Python 3 |
| Account for null values | Kea, Andrew, Riley | MySportsFeed API  Python 3 |
| Incorporate multi page front end | Riley, Brett | Reactjs |
| Better front end look | Riley, Brett | Reactjs |
| Train Model | Kea, Thomas, Andrew, |  |
| API calls for Odds and incorporate betting | Riley, Brett | MySportsFeed API  Python 3 |
| Add Advice column and gambling tips | Riley, Brett | Betting Websites  Reactjs |
| Output Final Result | Kea, Thomas, Andrew, Riley, Brett | Python 3  Reactjs |