SPM Final Project Report

League of Legends analytics

Group - 23

201801263 - Shreeya Godbole

201801402 - Bhargav Dave

201801404 - Smit Kumbhani

1. Project team details and roles of team members

1. Smit Kumbhani
   ID: 201801404
   Role: Team Leader, Developer, Documentation
2. Bhargav Dave
   ID: 201801402
   Role: Developer, Documentation
3. Shreeya Godbole
   ID: 201801263
   Role: Developer, Documentation

2. Project background

a. Purpose of project:

League of Legends(LoL) is an extremely famous MOBA game and the game includes a lot of mechanics including heroes, items, strategies and their combinations. The gameplay is fairly complex with towers, creeps and ancients giving players gold and XP in order to advance in level and better kill their opponents.

The sheer number of heroes, items and strategies makes playing it extremely difficult because there are too many possibilities and this can be overwhelming for new players because of the amount of combinations possible.

There is no one tool which provides a detailed repository of the game mechanics as well as a tool that shows descriptions of various heroes and items and can in fact show the various suggested combinations and strategies, ones that are computer recommended as well as ones that are used by the top players of the world.

We are setting out to create such a tool for players, old and new, of LoL so that they get the resources they need in an elegant fashion.

b. Scope of project:

We are aiming to do the following:
We are aiming to show details like win rates, hero win rates, leaderboards, comparison charts, as well as a log of recent performances in order for a player to better analyze and improve their playstyle and gameplay. We are aiming to add a functionality which allows the players to log in and check on the strategies that might be recommended to them and tailor made for their profiles. All in all, this would create a holistic software that houses tools that players of LoL, old and new, find useful in their daily ventures.

Deliverables:

- Create a website which has various functionalities which include: User sign-up, User login, Password change,Tier List, Real-time analysis, Comparison, Personal stats.
- Create a holistic software that contains tools helpful for the old and new players of LoL.

3. Perspectives

a. Who will use the system?

1. Gamers - People who are playing the games to find out which character is best for them and which attack - defense combination can lead them to victory.
2. Admin - To improve, remove or add new functionalities in the system.

b. Who can provide input about the system?

The following can provide creative inputs about the system:

- Investors: The people who are investing in the project will provide input as they are financing the project and will get affected by its outcomes.
- Users/Gamers: They can give suggestions and feedback about the software , this will help improve the project.
- Developers: They are the team members who can provide their thoughts and suggestions to better the project.

The following can provide data for our system:

- API: It will fetch data from the LoL database and provide raw data for our system on which we can perform analysis and generate reports and results.

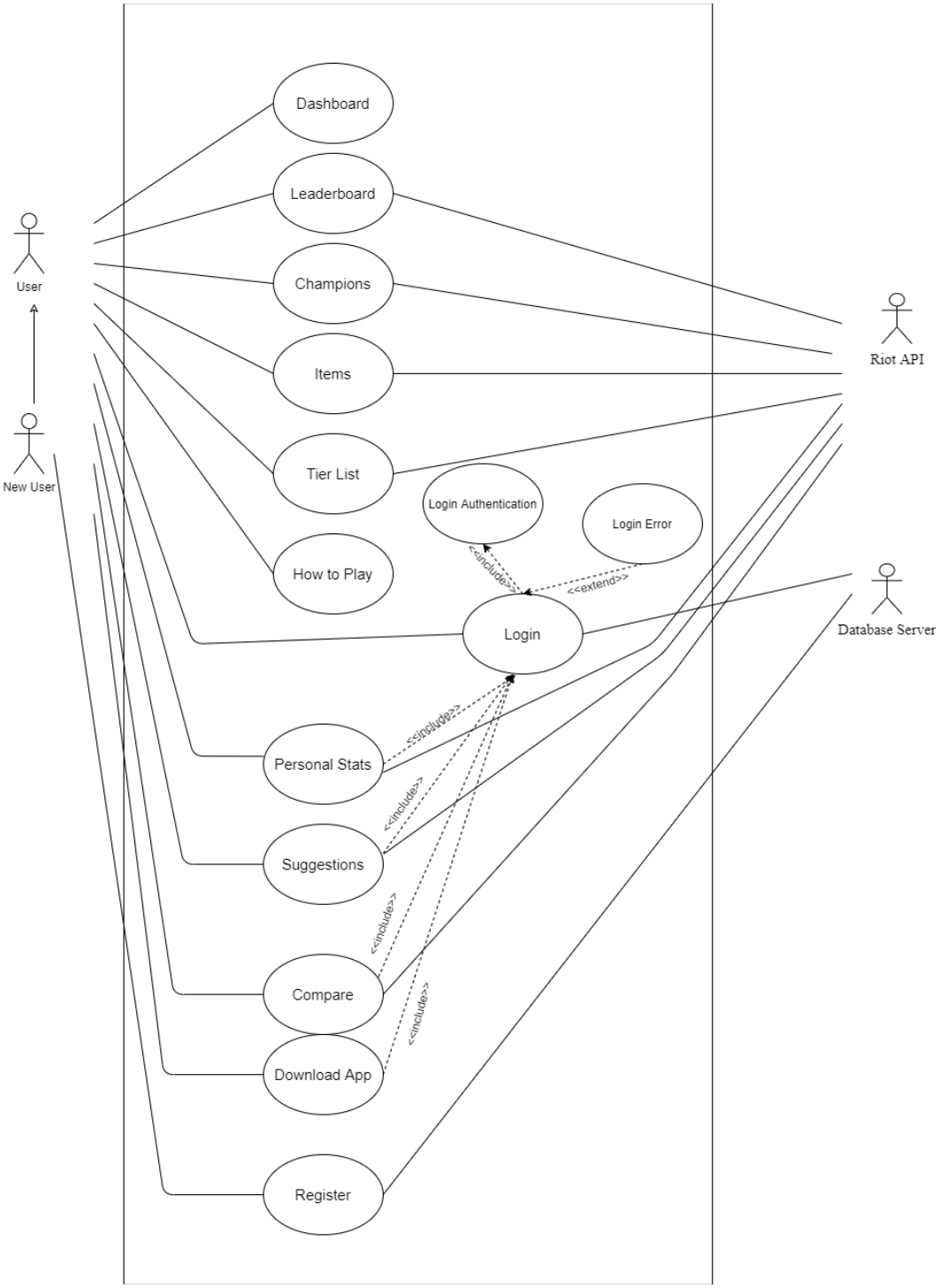
4. Project Objectives

a. Business rules for our system:

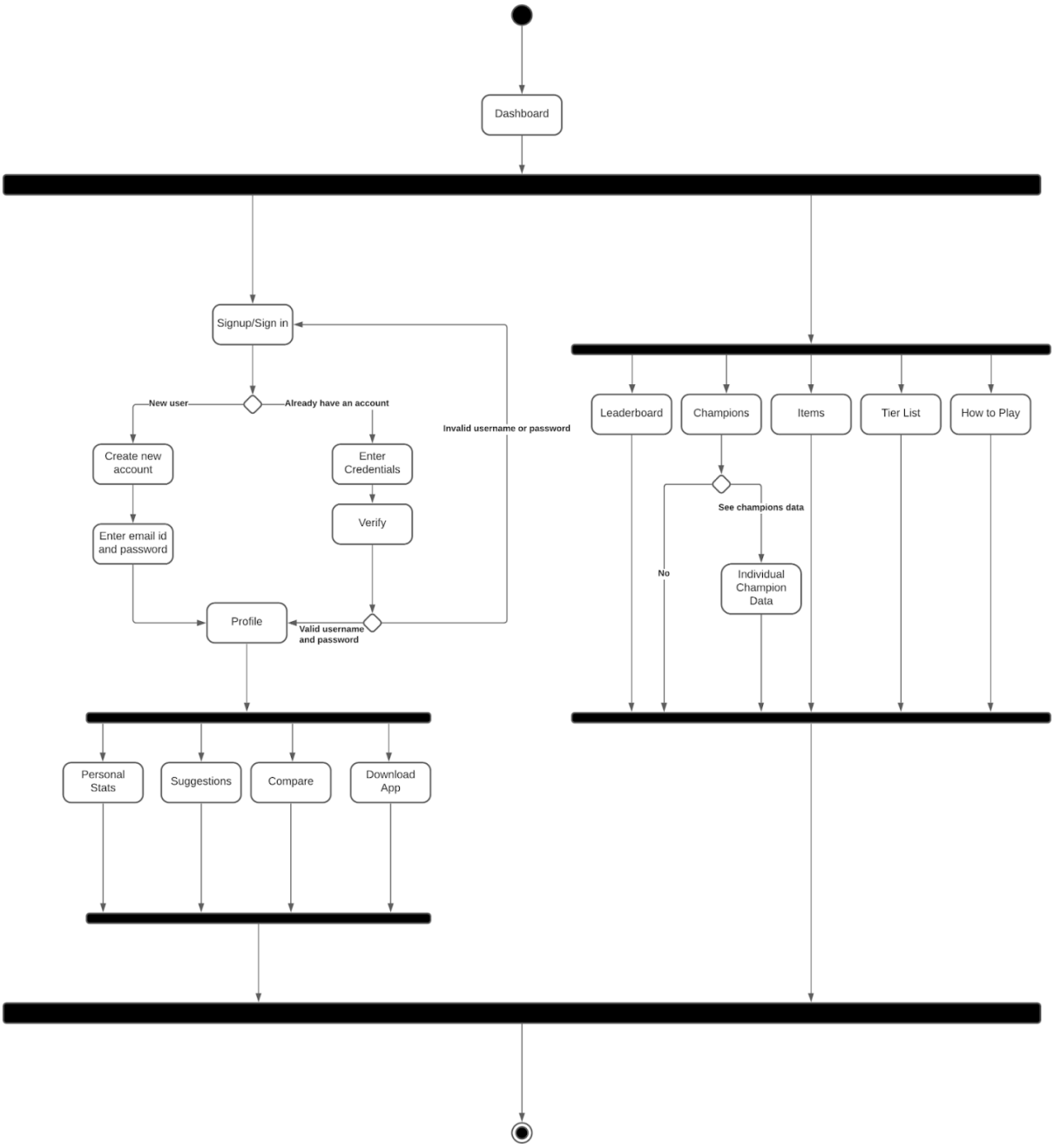
- All users are required to have a valid email address and password to access the system.
- Users data namely email address should be stored securely and will not be used by admin for any purpose.
- Data which is used by the system to calculate the stats should be up to date with the source.

b. System information and/or diagrams

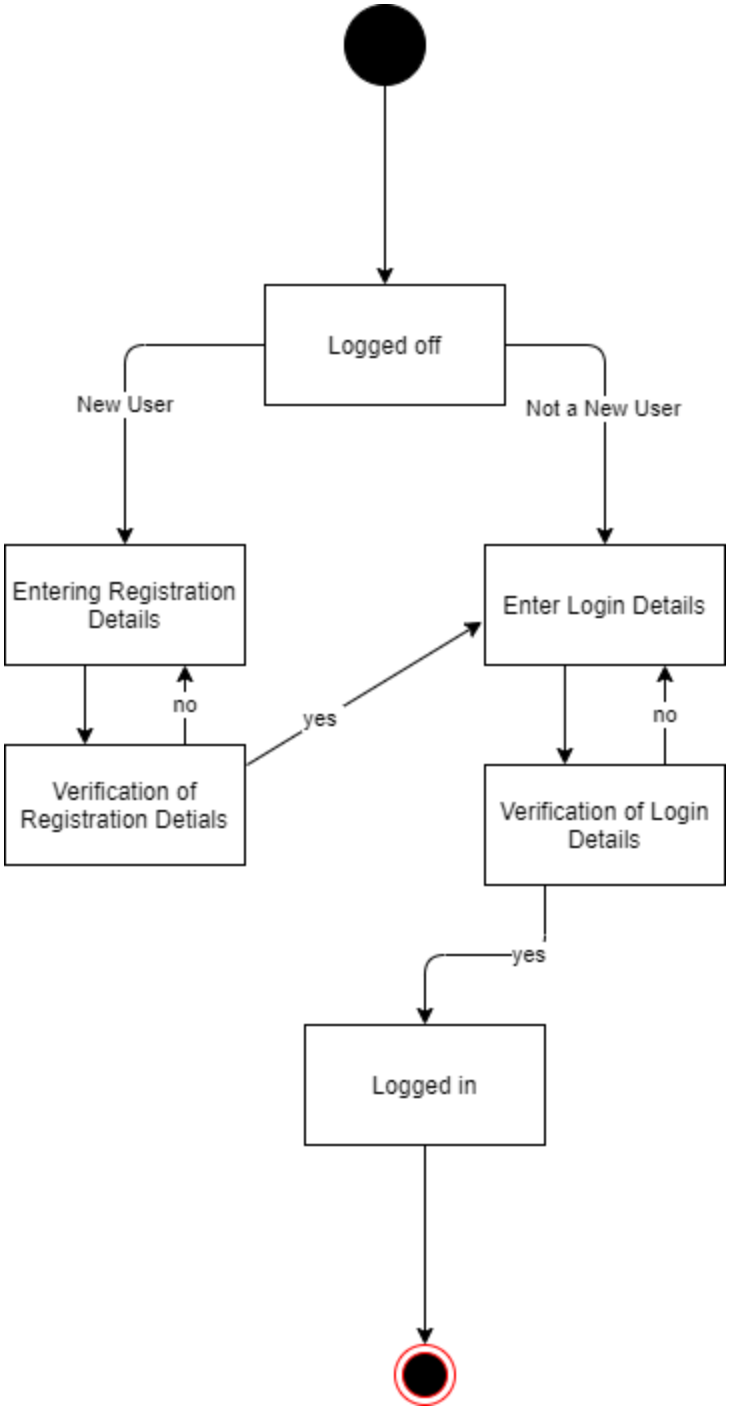
USE-CASE DIAGRAM:



ACTIVITY DIAGRAM:

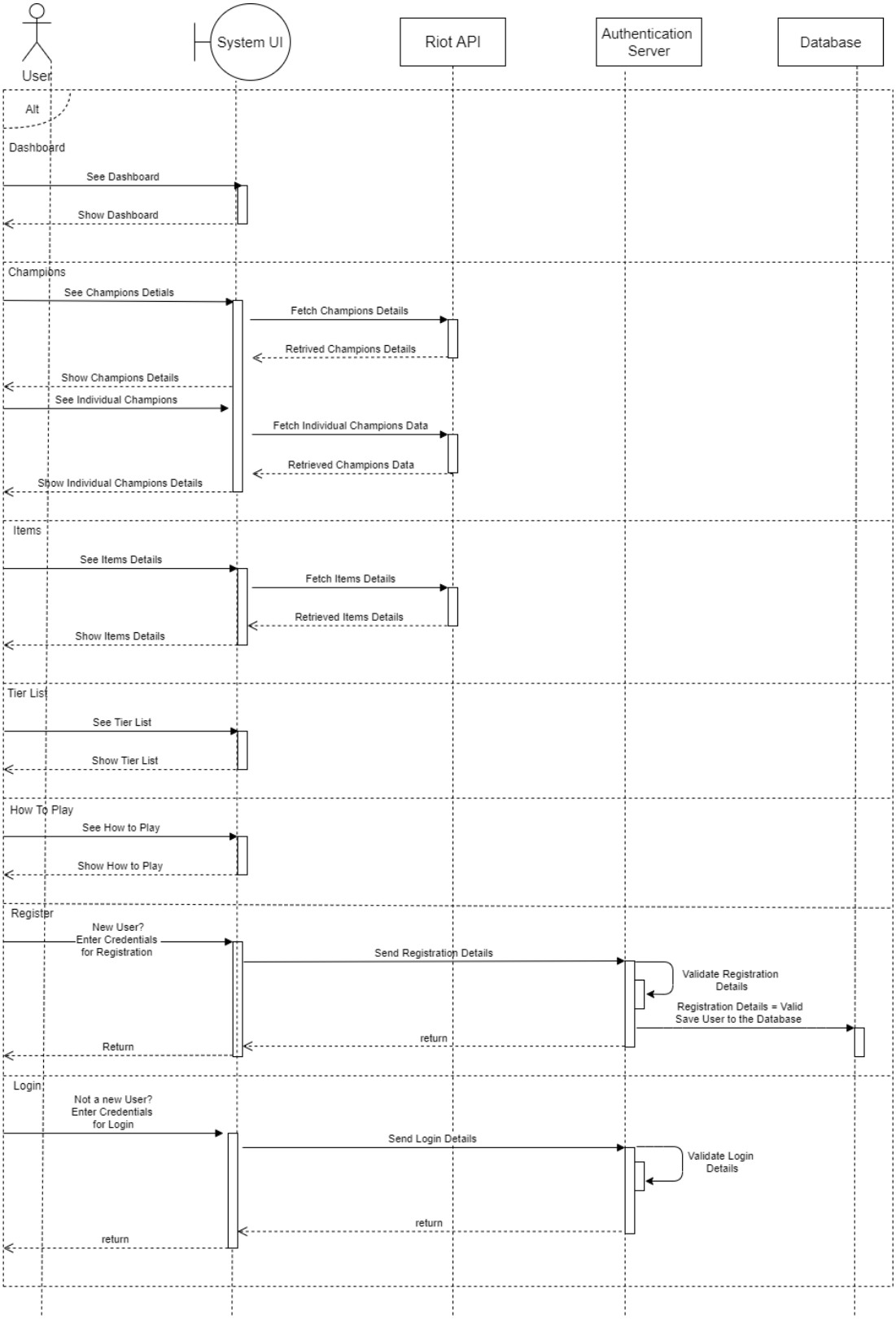


STATE DIAGRAM:

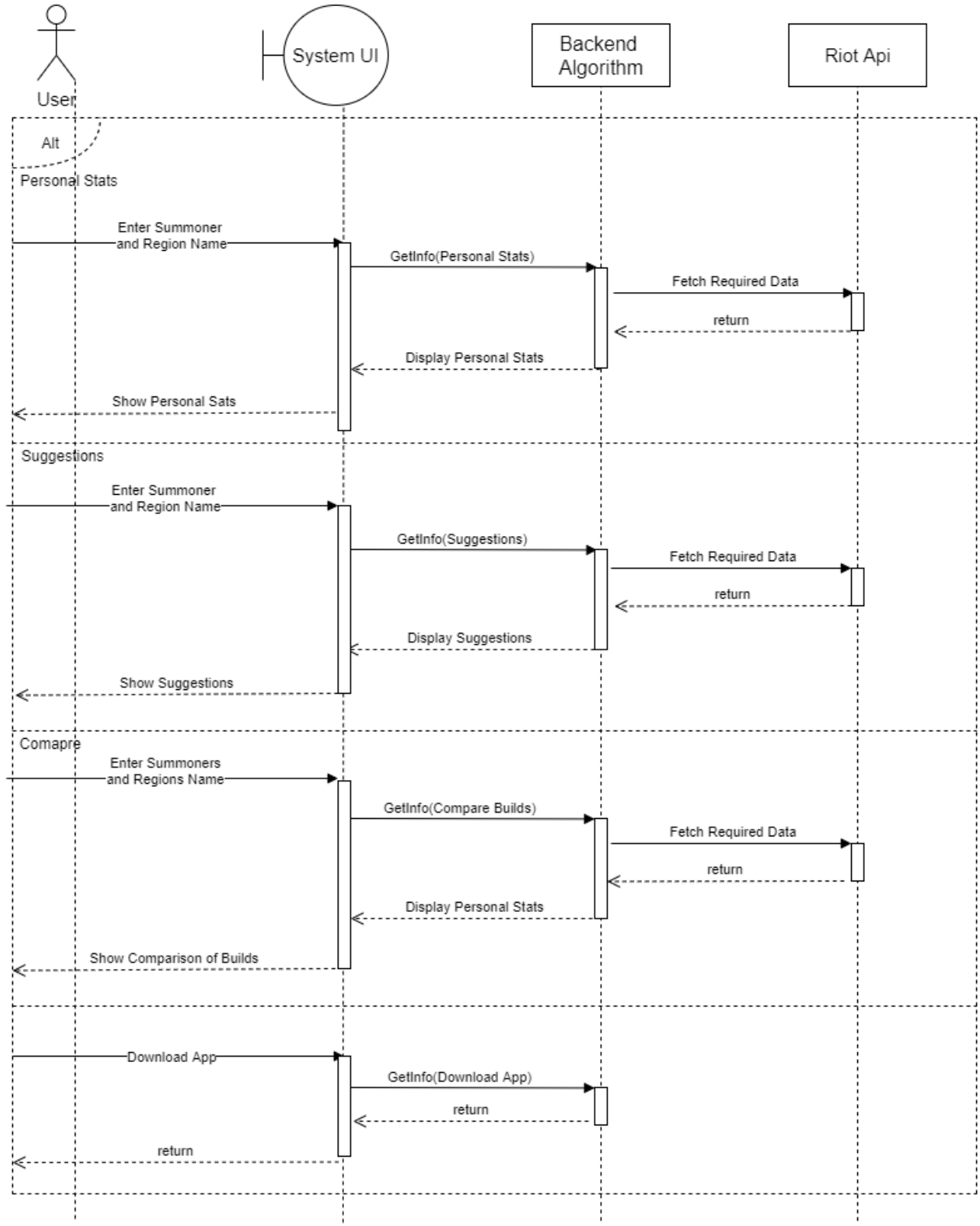


SEQUENCE DIAGRAM:

Before login:



After login:


User
System UI
Backend Algorithm
Riot Api
Alt
Personal Stats
Enter Summoner and Region Name
GetInfo(Personal Stats)
Fetch Required Data
return
Display Personal Stats
Show Personal Sats
Suggestions
Enter Summoner and Region Name
GetInfo(Suggestions)
Fetch Required Data
return
Display Suggestions
Show Suggestions
Comapre
Enter Summoners and Regions Name
GetInfo(Compare Builds)
Fetch Required Data
return
Display Personal Stats
Show Comparison of Builds
Download App
GetInfo(Download App)
return
return

c. Assumptions and dependencies, including time and effort estimates, details on tasks done by specific team members.

DEPENDENCIES BETWEEN THE TASKS:

Task name	Depends on / Required for	Task name
Gather requirements through the internet, surveys and analyze them	Required for	Write user stories
Write user stories	Required for	Note down the functional and non-functional requirements analyzed from the user stories.
Understanding what tasks to do and planning their flow	Required for	Creating a timeline for tasks ahead on Jira
Understanding what tasks to do and planning their flow	Required for	Making an activity diagram, Making a state diagram
Making a Concept Map	Depends on	Gather requirements through the internet, surveys and analyze them
Making a use case diagram	Depends on	Write user stories
Making a state diagram	Required for	Making a sequence diagram
Implementing Home page features using react	Depends on	Creating a timeline for tasks ahead on Jira
Searching for a data source to use and find API to get LoL data	Required for	Writing python scripts for data fetching at the backend
Linking the backend to frontend and testing for errors while running	Depends on	Implementing Home page features using react, Writing python scripts for data fetching at the backend
Linking the backend to frontend and testing for errors while running	Required for	Performing Black Box testing

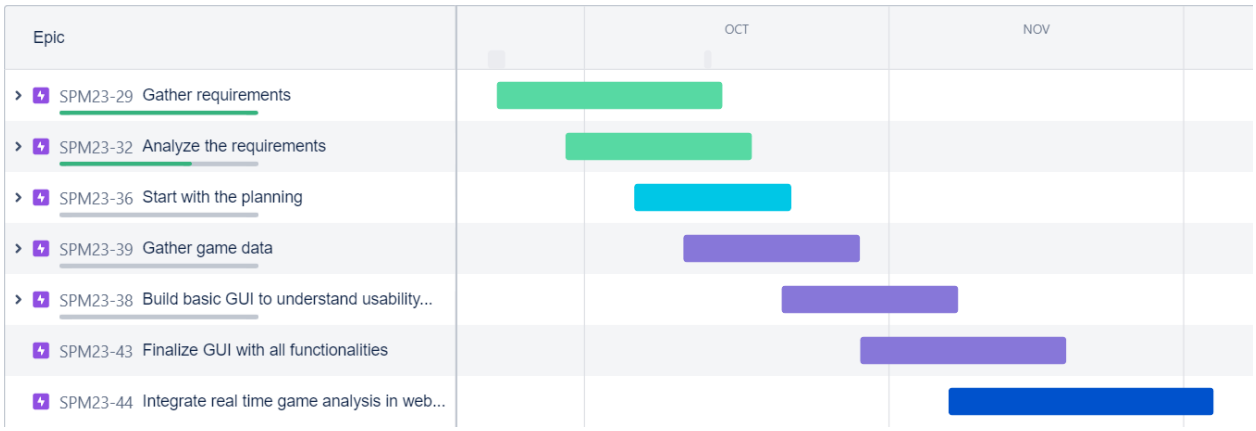
Performing Black Box testing	Required for	Performing GUI testing
Performing GUI testing	Required for	Deploy our code on the local computer
Deploy our code on the local computer	Required for	Try to deploy it on other servers

TIME AND EFFORT ESTIMATE FOR EACH TASK ALONG WITH WHO PERFORMED THE TASK:

Activity ID	Activity Name	Estimated times (hours)	Estimated duration (dates)	Done by:
1	Gather requirements through the internet, surveys and analyze them.	48 hours	25/09/2021 to 27/09/2021	Shreeya
2	Write user stories	12 hours	25/11/2021	All three
3	Note down the functional and non-functional requirements analyzed from the user stories.	8 hours	18/11/2021 to 23/11/2021	Bhargav, Smit
4	Understanding what tasks to do and planning their flow	10 hours	29/09/2021	All three
5	Searching for a data source to use and find API to get LoL data	72 hours	30/09/2021 to 3/10/2021	Bhargav
6	Creating a timeline for tasks	8 hours	5/10/2021	Shreeya

	ahead on Jira			
7	Making a Concept Map.	10 hours	23/10/2021 to 25/11/2021	Smit
8	Making an activity diagram	10 hours	23/11/2021 to 25/11/2021	Bhargav
9	Making a use case diagram	11 hours	24/11/2021 to 26/11/2021	Shreeya
10	Making a state diagram	9 hours	24/11/2021	Smit
11	Making a sequence diagram	12 hours	25/11/2021 to 26/11/2021	Bhargav
12	Implementing Home page features using react	80 hours	20/10/2021 to 27/11/2021	All three
13	Writing python scripts for data fetching at the backend	80 hours	20/10/2021 to 27/11/2021	All three
14	Linking the backend to frontend and testing for errors while running	80 hours	1/12/2021 to 5/12/2021	All three
15	Performing Black Box testing	8 hours	5/12/2021	Smit
16	Performing GUI testing	8 hours	5/12/2021 to 7/12/2021	Bhargav

d. Project plan based on 4(c) using Gantt chart



e. Design and implementation constraints:

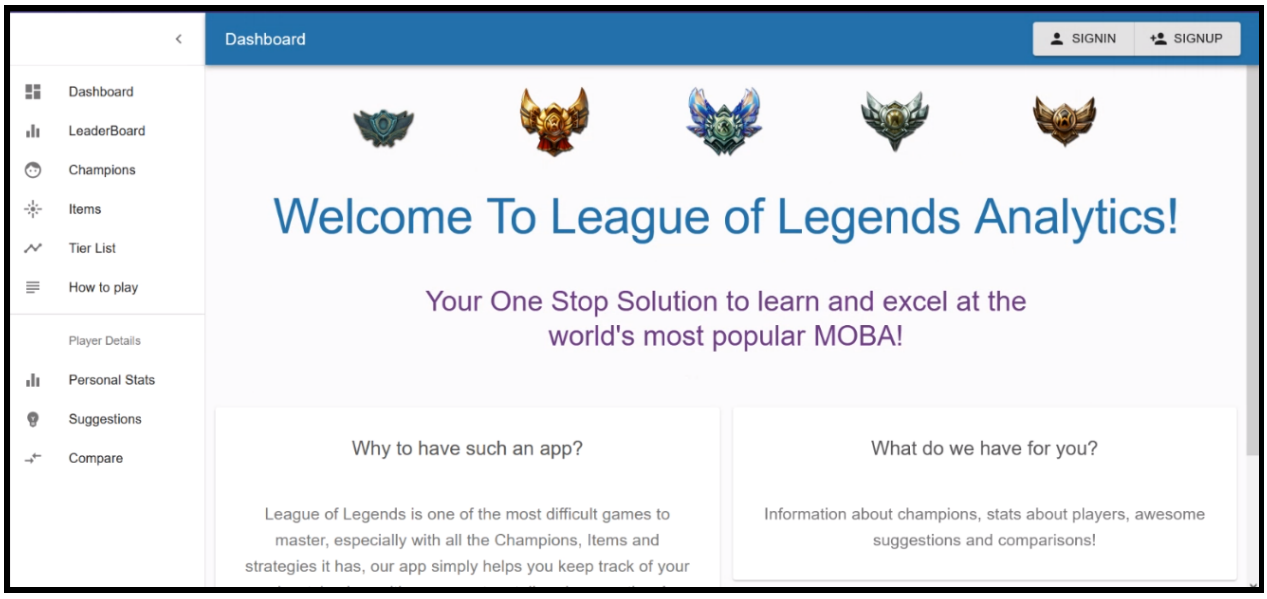
- Since we are depending on the LoL API to get data, it's constraints will be inherited to our system. One of the constraints is limitation of data. If data is limited then there will be less data to analyze and as a result there will be less results.
- Another API related constraint is that it allows limited calls per given time. Sometimes they are limited due to low latency of the LoL server.
- This constraint can also affect the scalability of the system performance. When there will be too many users accessing the system, it will be required to fetch data more frequently and hence have to make calls, which are again limited.
- The LoL database keeps updating as players/gammers keep playing the game. If we are to update our database in sync with LoL's then it will be resource consuming.
- SQlite database is a binary file and hence regularly updating that does not work with most version control softwares like github. Because updating something changes a value on a single line or row and it can not be sometimes recognized as change by the softwares like github because it is a binary file.

5. Risks associated with your project and its mitigation strategies:

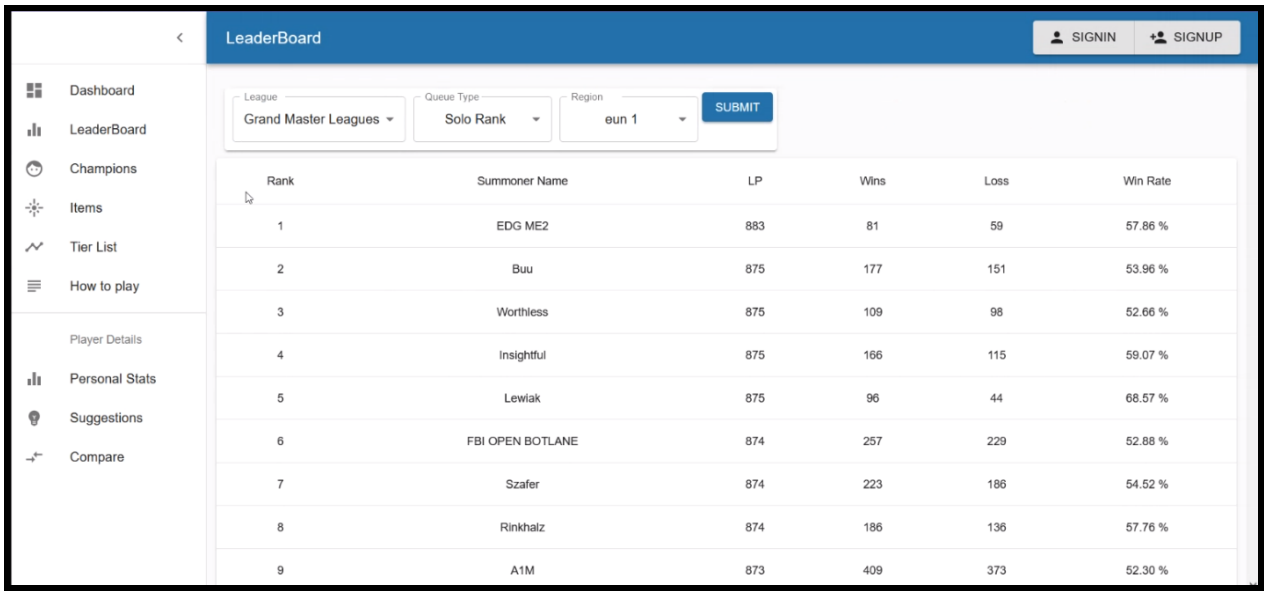
- We are using a free version of API to get required data. Since it is a free version it allows only a limited amount of calls. So data will be updated at a lower frequency and hence it will not be in sync with real time data. It will require the paid version or premium version of API to make more frequent calls to access data more frequently or in real time.
- It is currently hosted on a local machine so it can handle a limited number of users. If the system gets more traffic than it can handle, it will overload the local machine. In order to overcome this and make it more scalable the system has to be hosted on a server capable of handling traffic.
- We are currently not using authentication services to confirm email addresses. There is a risk that someone might make an account with a fake email address or use someone else's email address. From the perspective of security we must put a stop to this kind of behavior.
- We have not yet started to store data with encryption. We are still storing data unencrypted. This may leak personnel contact details like email addresses under cyber attack. To avoid data leaks we will have to find a suitable encryption algorithm and apply it on the data.

6. UI/UX design of the complete system

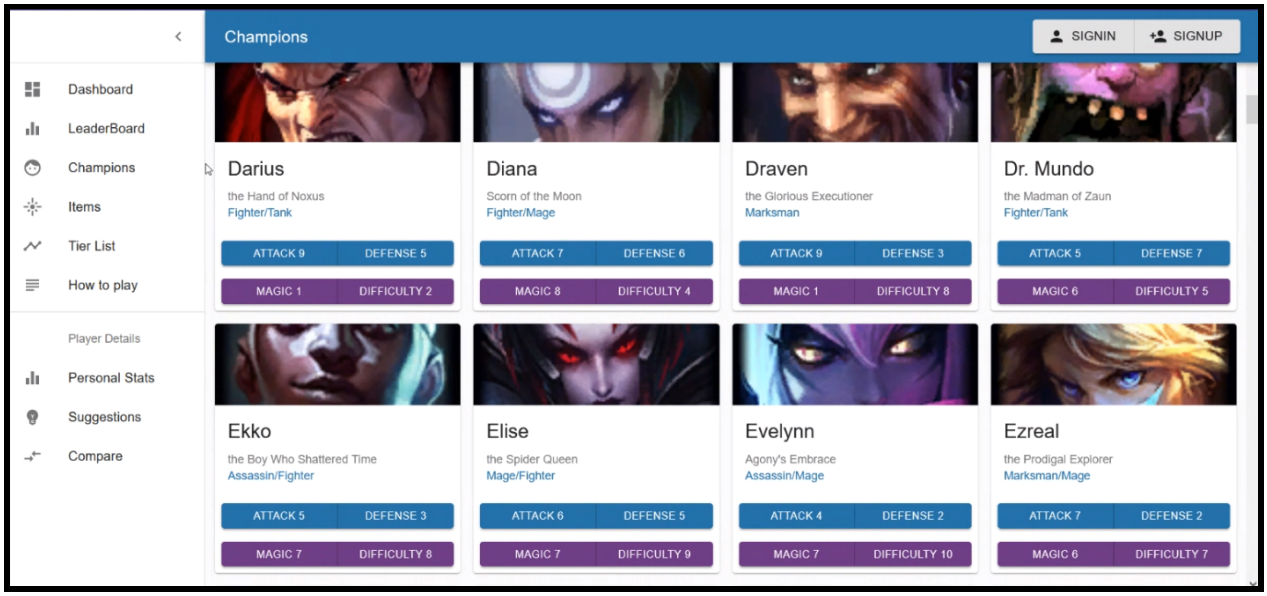
DashBoard:



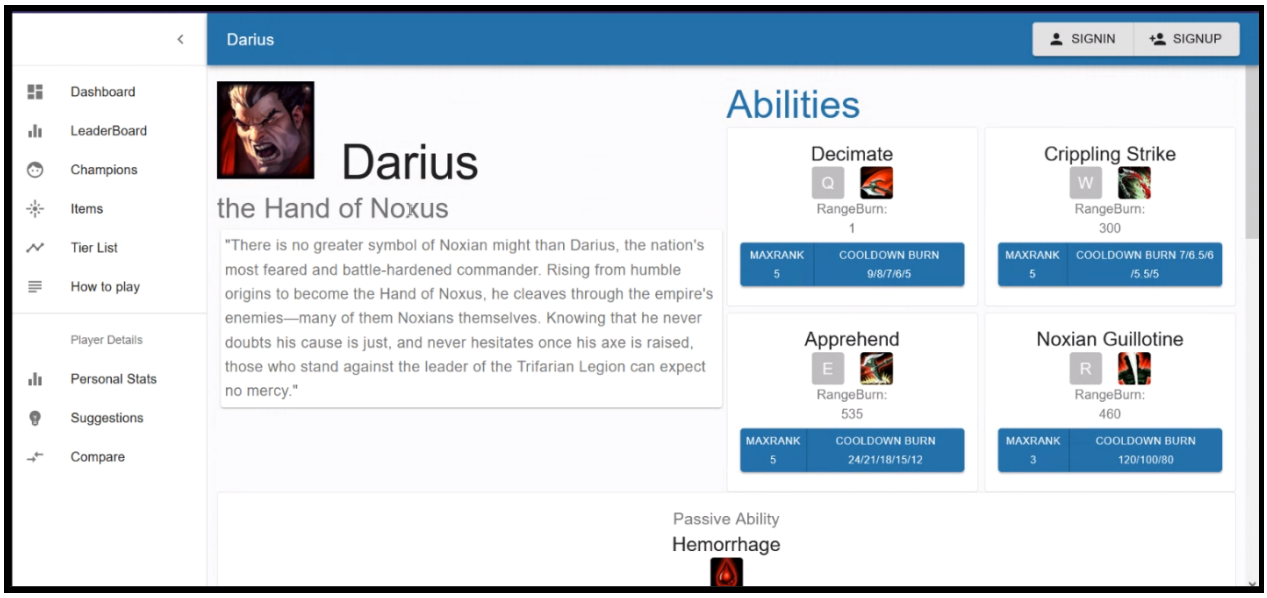
Leaderboard:



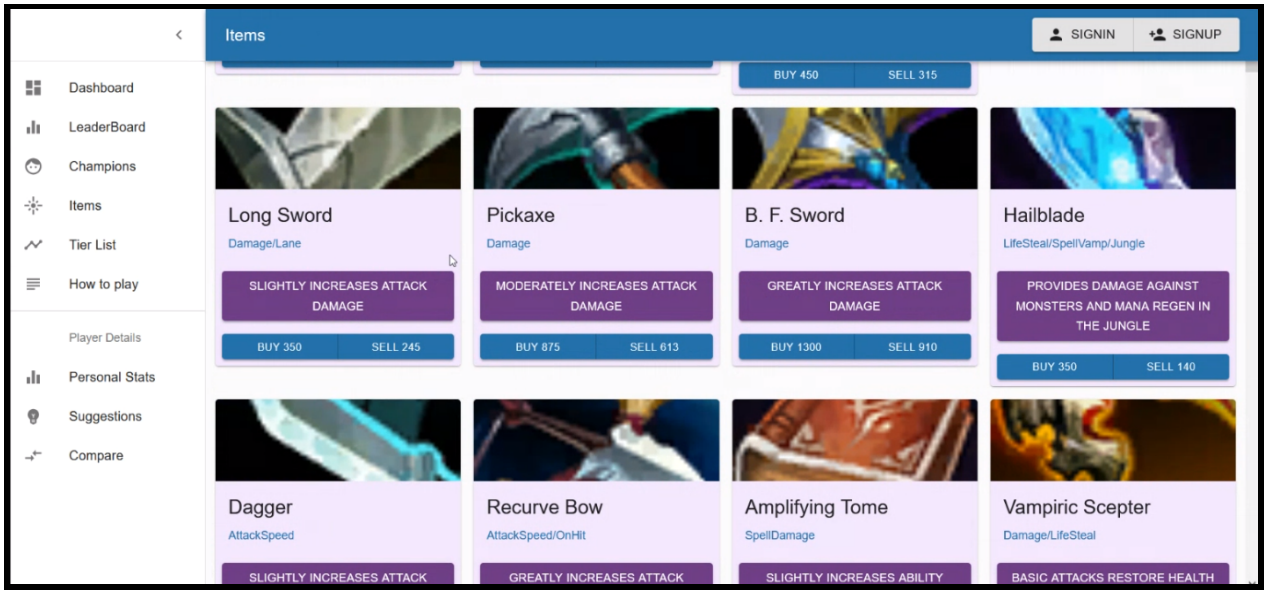
Champion Information:



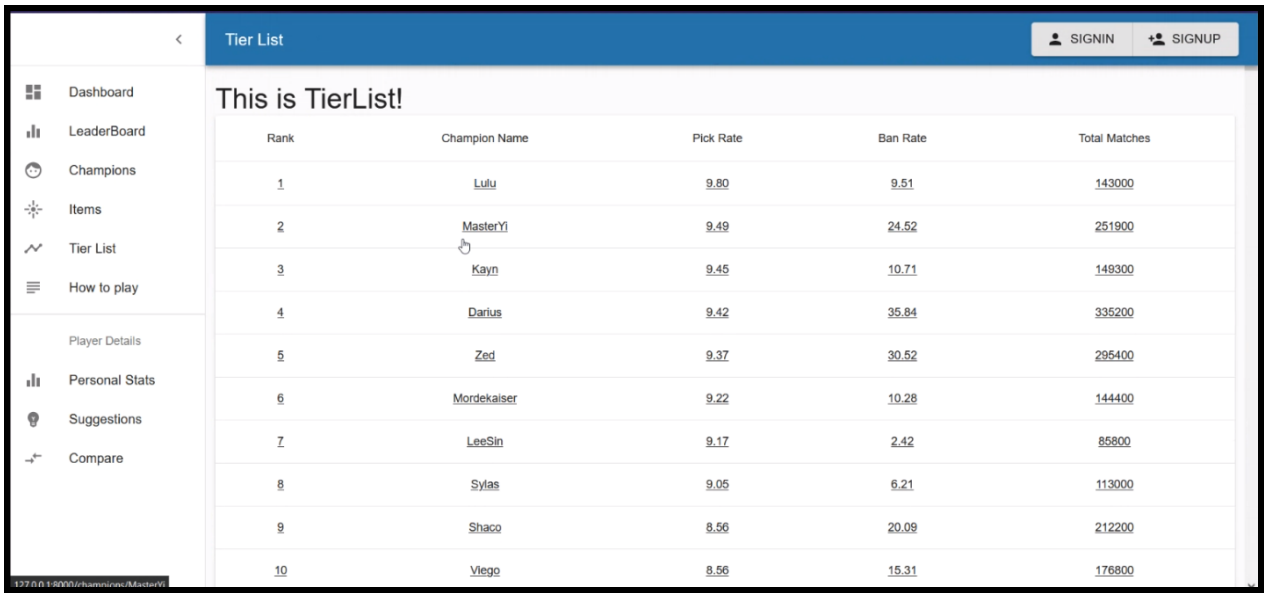
Particular champion's information:



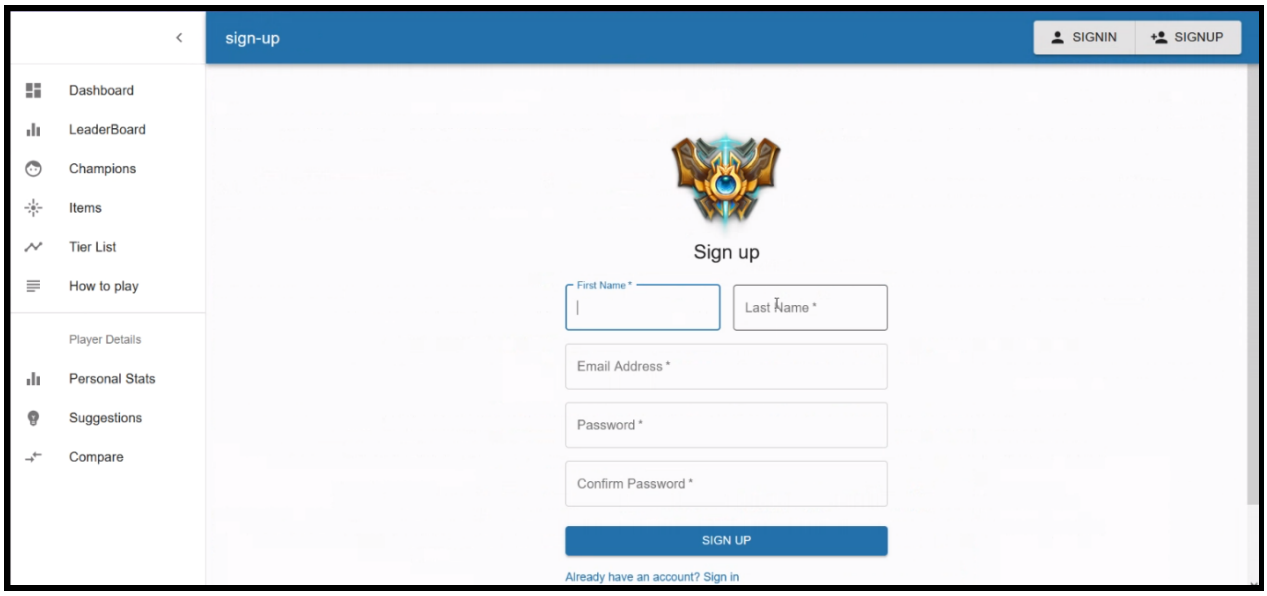
Item Information:



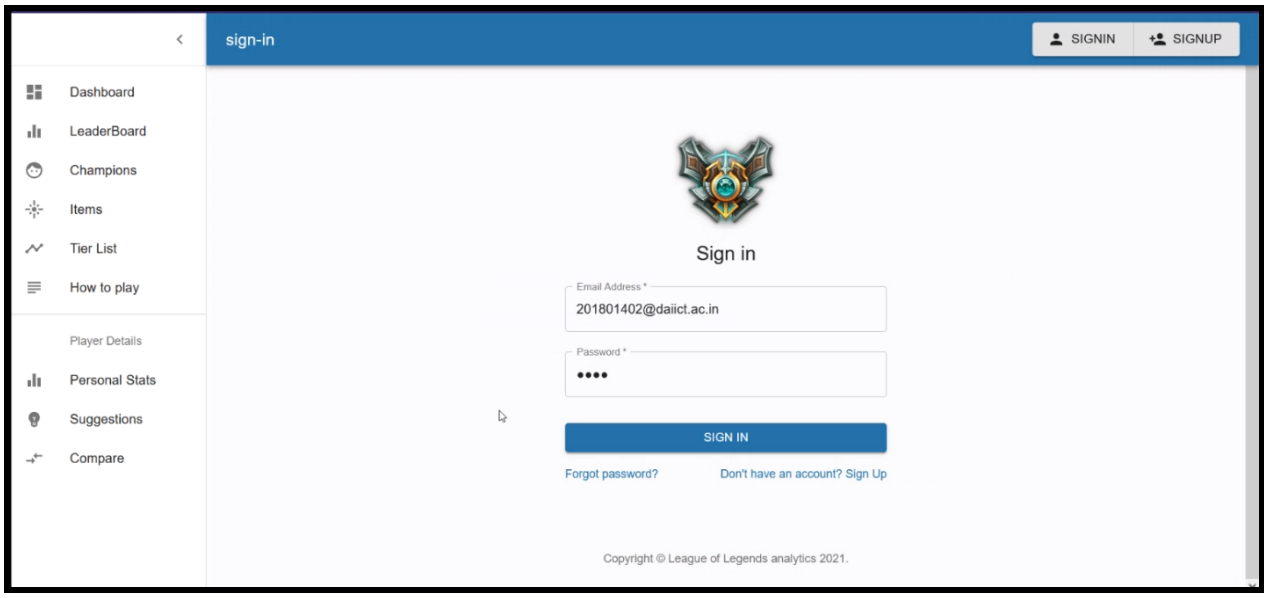
TierList:



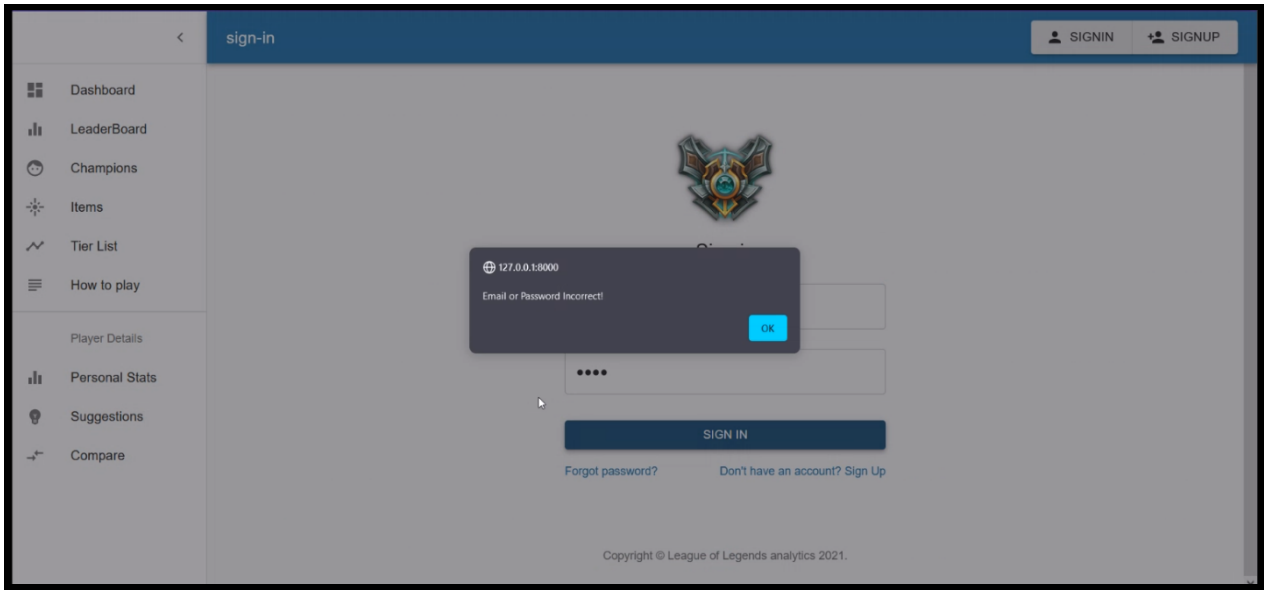
Sign-up:



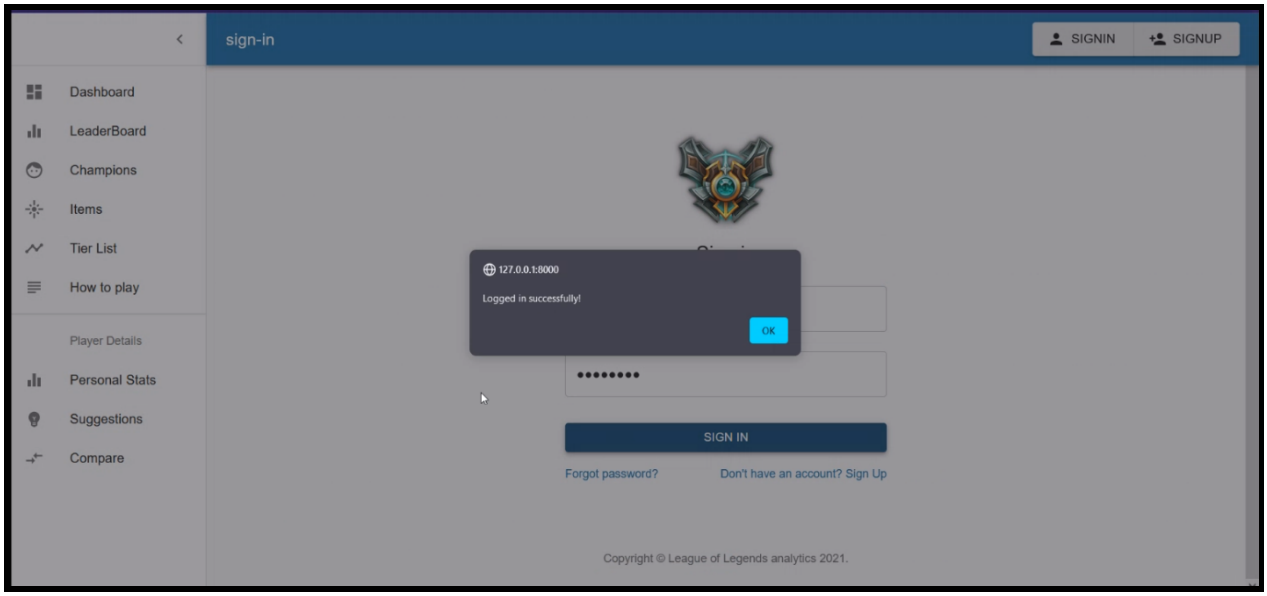
Sign -in :



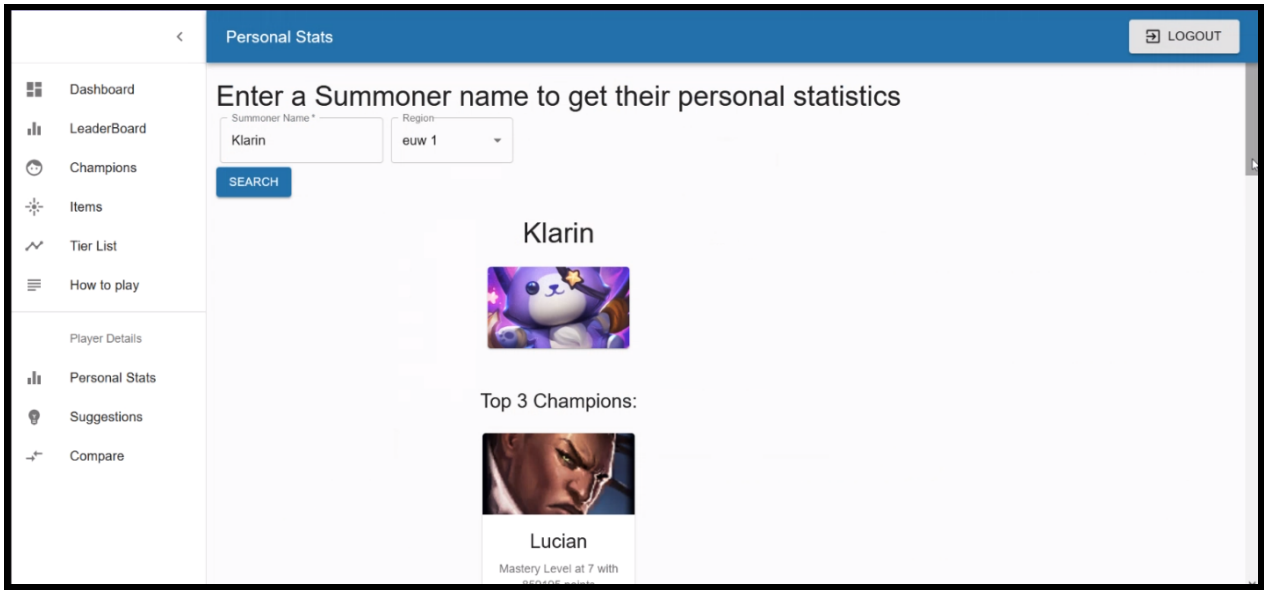
Invalid Password:

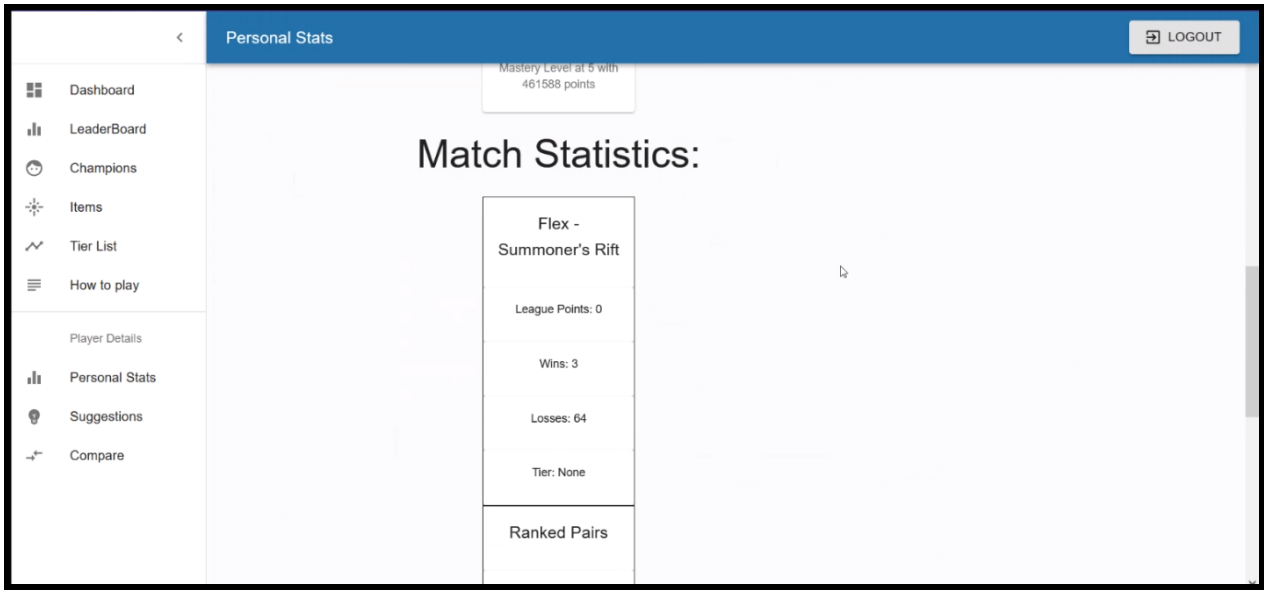


Correct Login:

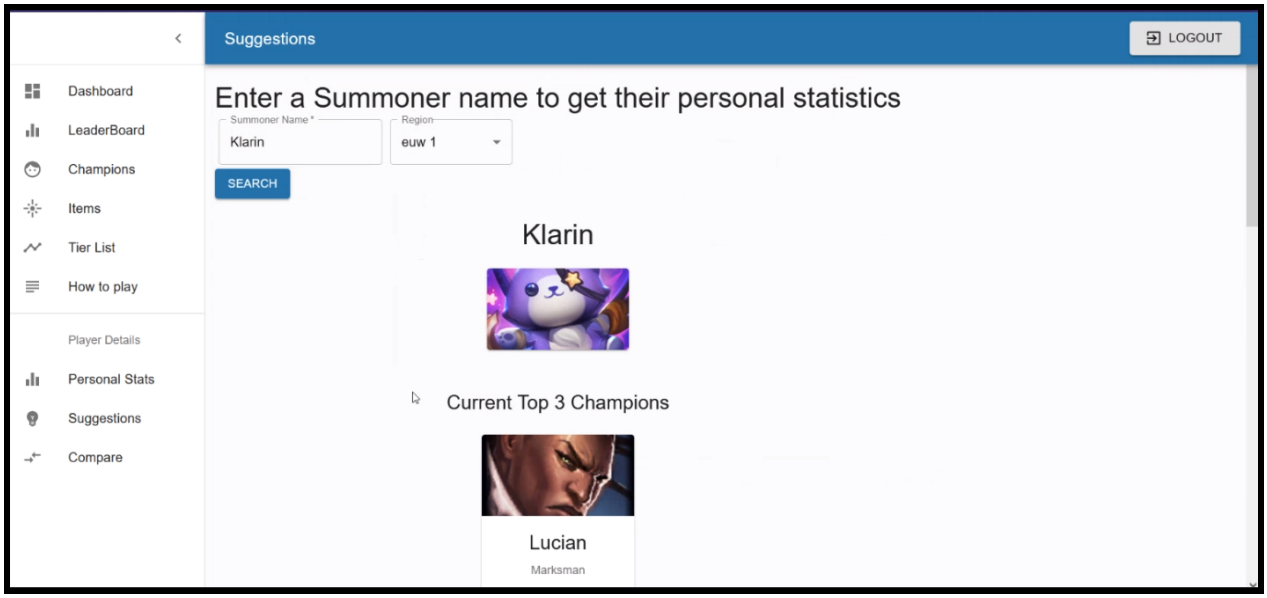


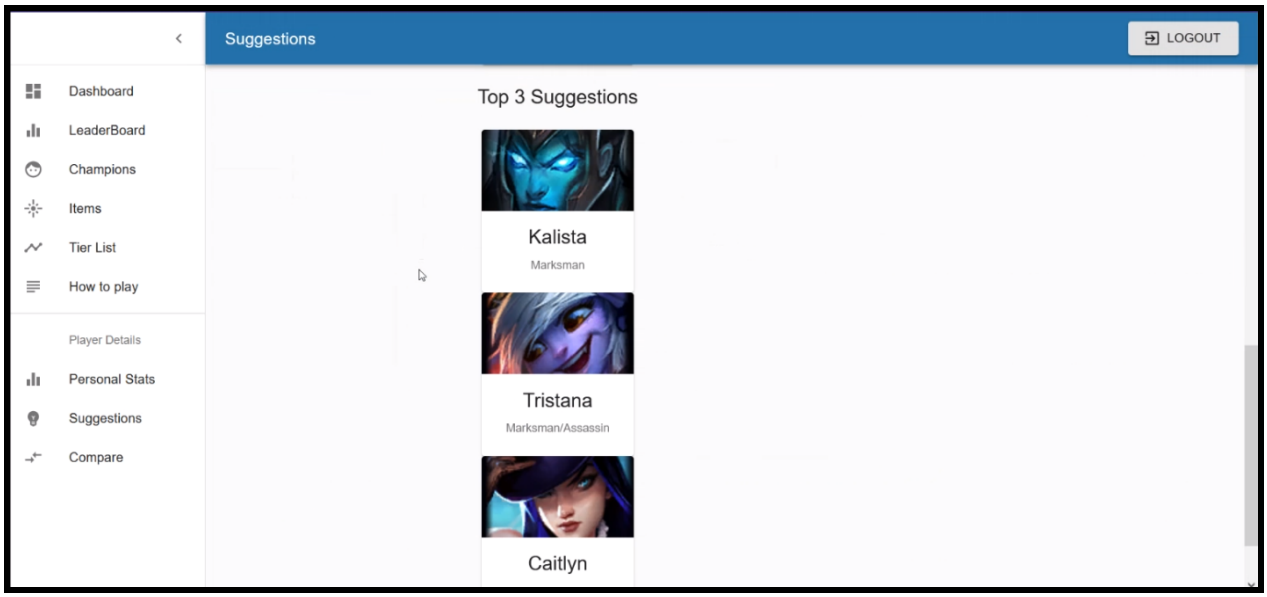
Personal Stats:



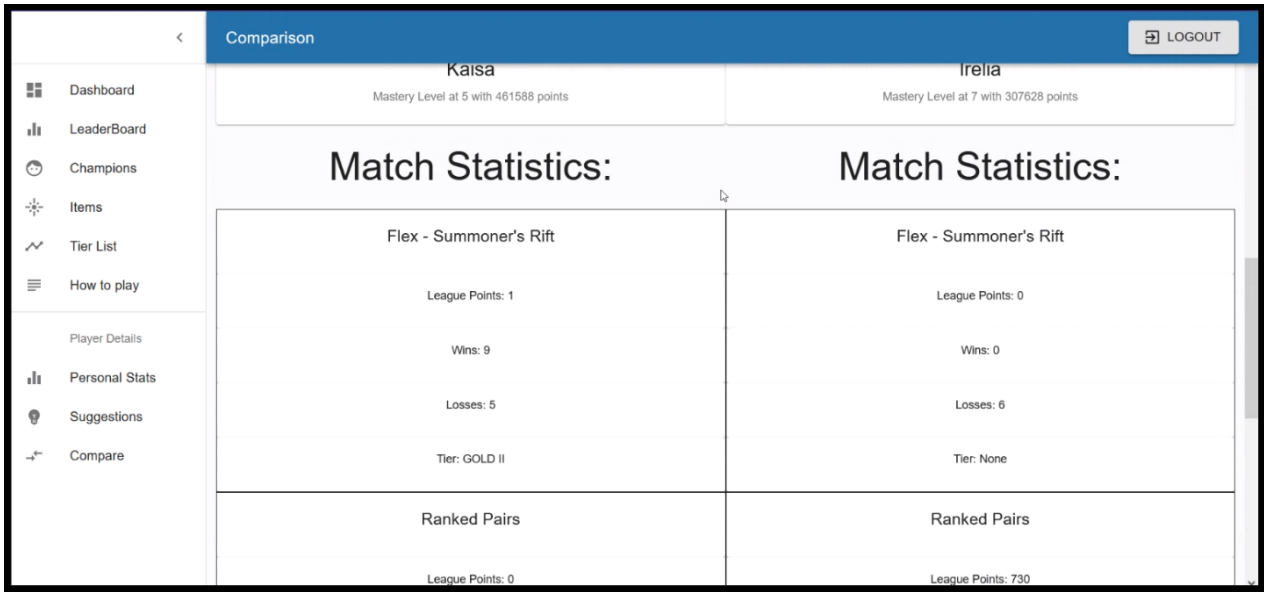


Suggestions:

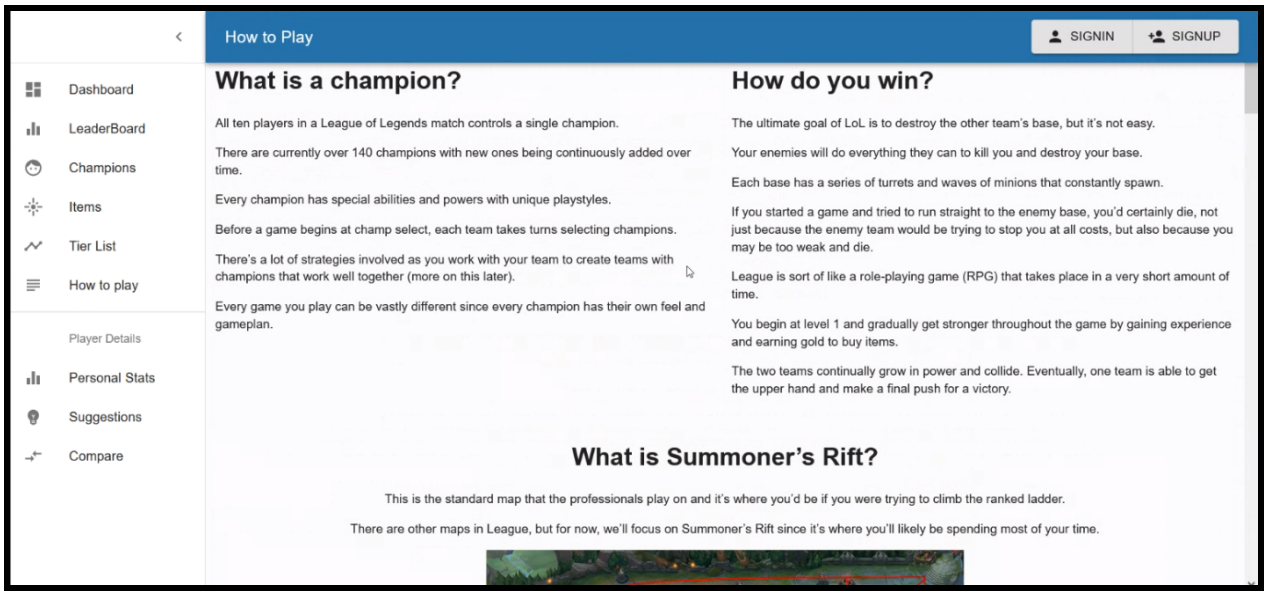




Comparison:

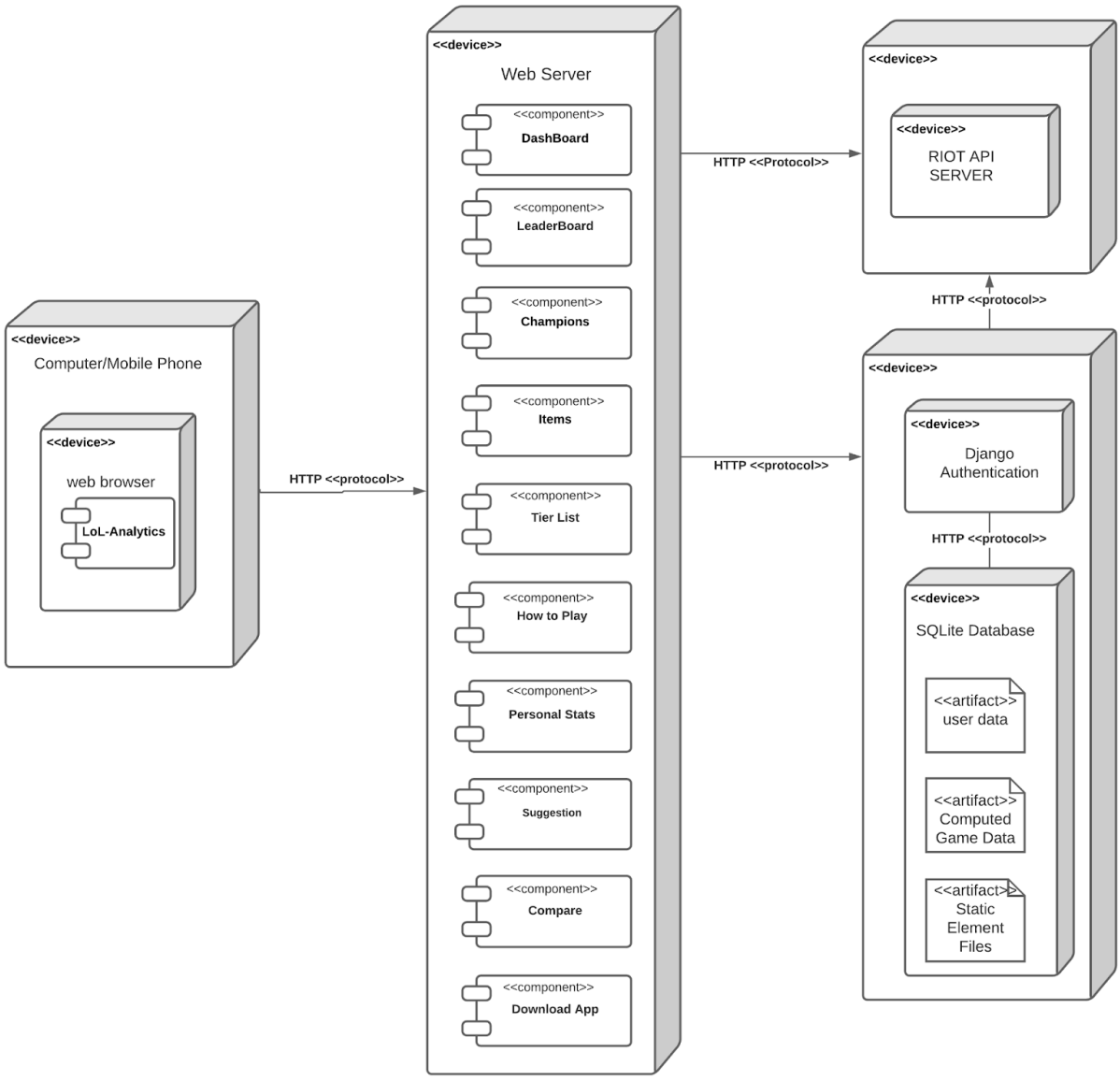


How to play:



7. System architecture model (or Implementation model)

DEPLOYMENT MODEL:



8. Implementation Technologies including front-end and back-end decisions

Frontend:

- react.js:
  React is a free and open-source front-end JavaScript library for building user interfaces based on UI components.

- node.js:
  Node.js is an open-source, cross-platform, back-end JavaScript runtime environment that runs on the V8 engine and executes JavaScript code outside a web browser. Node.js lets developers use JavaScript to write command line tools and for server-side scripting—running scripts server-side to produce dynamic web page content before the page is sent to the user's web browser.

- Material ui:
  Material-UI is simply a library that allows us to import and use different components to create a user interface in our React applications. This saves a significant amount of time since the developers do not need to write everything from scratch.

Backend:

- Python3: It is a newer version of python programming language which is an interpreted high-level general-purpose programming language.

- django (for server): Django is a Python-based free and open-source web framework that follows the model–template–views architectural pattern.Django's primary goal is to ease the creation of complex, database-driven websites.

- Sql: SQL is a domain-specific language used in programming and designed for managing data held in a relational database management system.

9. Details justifying how various team members contributed and their extent of contribution.

Name	Contribution
Smit Kumbhani	User stories Requirement gathering Planning workflow Making concept map Making state diagram Implementing home screen on react Backend code on python Linking of backend and frontend Perform testing
Bhargav Dave	Functional and non-functional requirements User stories Project planning Finding API as data source Activity Diagram Sequence diagram Implementing home screen on react Backend code on python Linking of backend and frontend GUI testing
Shreeya Godbole	Gathering requirements Writing user stories Project planning Creating Timelines Use case diagram Implementing home screen on react Backend code on python Linking of backend and frontend

10. Future enhancements to the system.

- We can have a program which renews the API key every 24 hours for a smooth functioning or we could use a paid version of API.
- In future this software can be deployed on a server which better tackles the case of user traffic on the website.
- We can use better security measures for resolving security issues of fake login and use data encryption for data privacy.

11. Open, unresolved Issues (if any)

- We could not implement Forgot Password Functionality.
- The API key expires every 24 hours so the API key changes frequently.
- A few users' personal statistics are not available publicly so their data is not fetched by the API.

Source Code link:
https://github.com/Bhargav-Dave/MOBA-Analytics.git