Ai Use Case : concept of the sports Results Chatbot

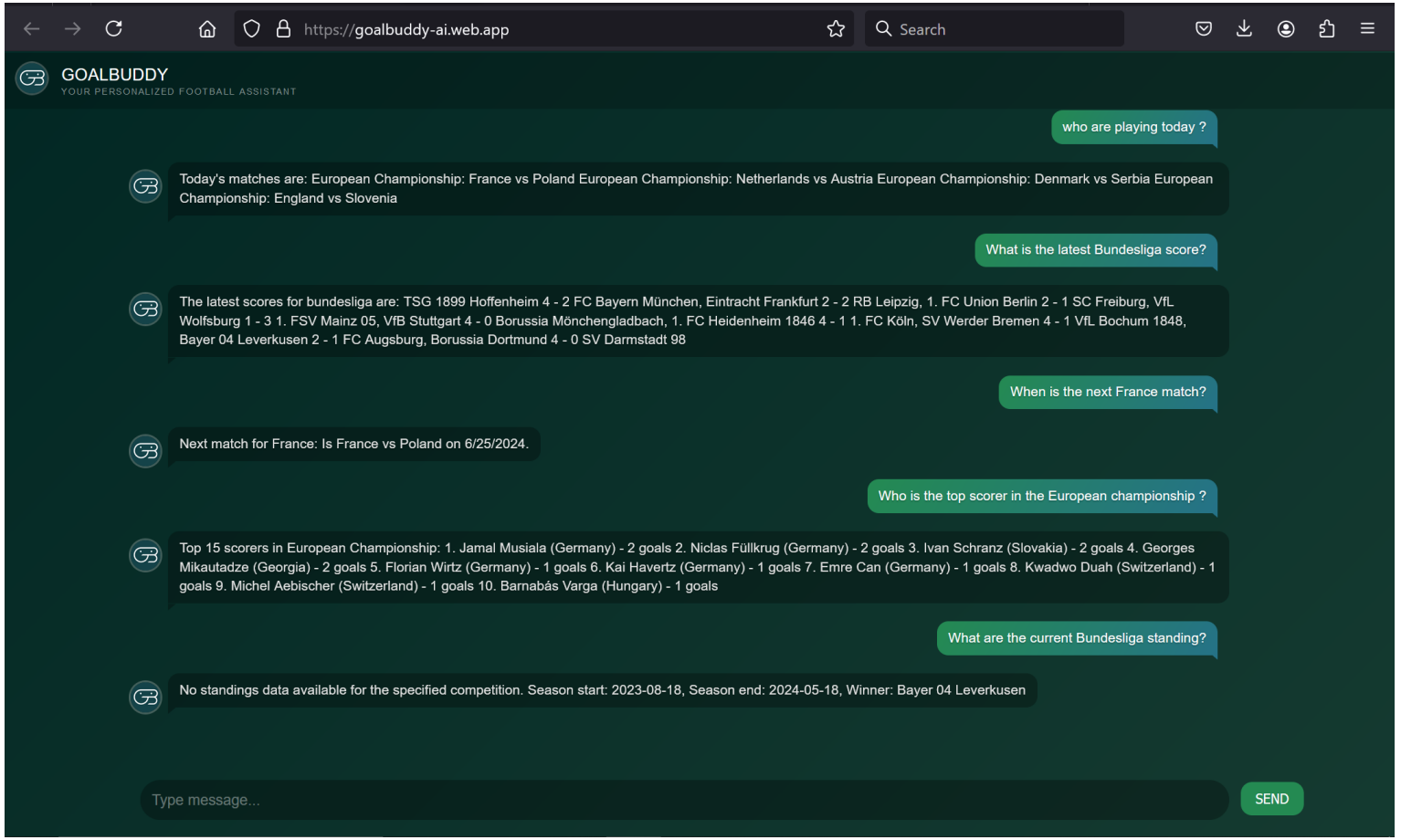
GoalBuddy – Your Personalized Football Assistant

Clement kanku

introduction

GoalBuddy is an application we've developed to make life easier for soccer fans, by enabling them to easily track competition scores and rankings. This project was carried out as part of our studies in artificial intelligence. The application is built using modern technologies such as React, Node.js and Wit.ai, creating a simple and pleasant user experience. You can discover and test **goalbuddy** at the following link: https://goalbuddy-ai.web.app/

GoalBuddy Features



Technologies Used

List of frameworks and tools:

React https://reactjs.org/ :

Role: React allowed us to build the app user interface. It allows us to create reusable components and efficiently manage application state.

Description: Thanks to React, the application offers a responsive and dynamic interface where users can easily view the scores, standings and upcoming matches of their favorite teams.

Node.js https://nodejs.org/en :

Role: Node.js is the server technology used for our backend. It allows JavaScript to be executed on the server side, making it easier to manage requests and responses between the frontend and the databases.

Description: Node.js handles user requests, interacts with the soccer API to retrieve the necessary data, and returns that data to React for display.

Wit.ai https://wit.ai/ :

Role: We used Wit.ai for intent recognition and natural language understanding (NLP). It allows the application to understand and interpret user queries in natural language.

Description: Thanks to Wit.ai, our web app can understand questions like "What is France's next match?" and provide appropriate responses by querying the Football API.

Football-data.org https://www.football-data.org/ :

Role: Football-data.org provides necessary football data, such as match scores, league standings and match schedules.

Description: The football-data.org API is essential for obtaining the up-to-date information that our app ddisplays to users. It allows you to retrieve precise and reliable data on football competitions.

Firebase https://firebase.google.com/ :

Role: We used Firebase to host the frontend application. It offers a reliable and scalable hosting solution for web applications.

Description: With Firebase, we deployed and hosted our application with ease, ensuring optimal availability and performance for all users.

Heroku https://dashboard.heroku.com/apps :

Role: We used Heroku to host the backend server of our application. It makes it easier to deploy and manage Node.js applications.

Description: Heroku allows us to deploy our backend server quickly and reliably. It also manages application updates and scalability.

Visual studio code https://code.visualstudio.com/ :

Role: Visual studio code is a simple, flexible code editor with an easy-to-use interface, it allowed us to implement and compile the code of our application

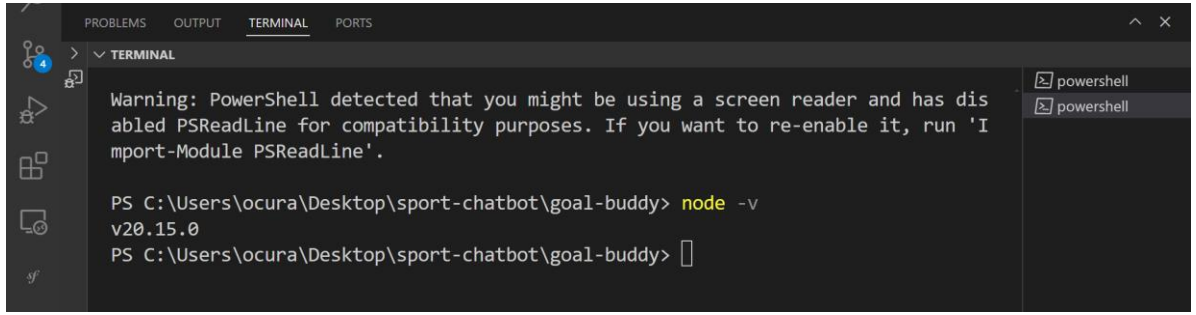
Description: it was with visual studio code that we were able, from our terminal, to launch the “create-react-app goalbuddy” command for the base of our application.

Environment Configuration

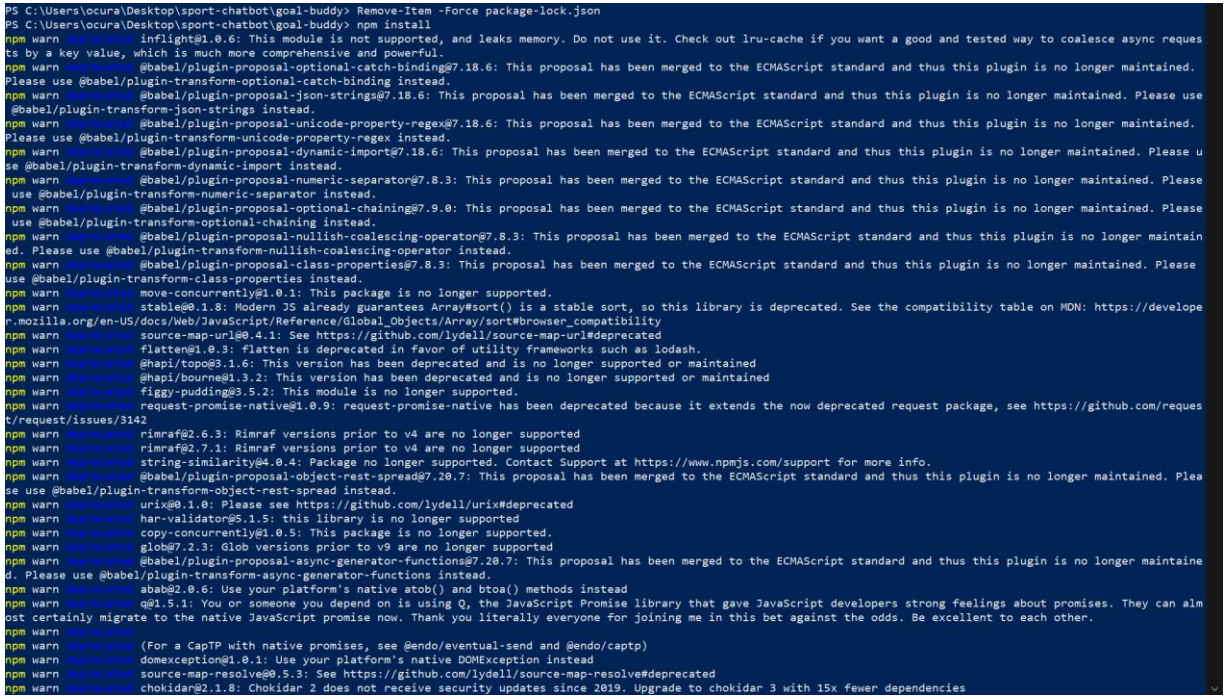
Preparing the development environment for building our web ap

Description: To start working on a web app, it is essential to configure our development environment correctly. Here are the detailed steps to prepare our environment and start the project on our local machine.

Installing Node.js



Installing dependencies :



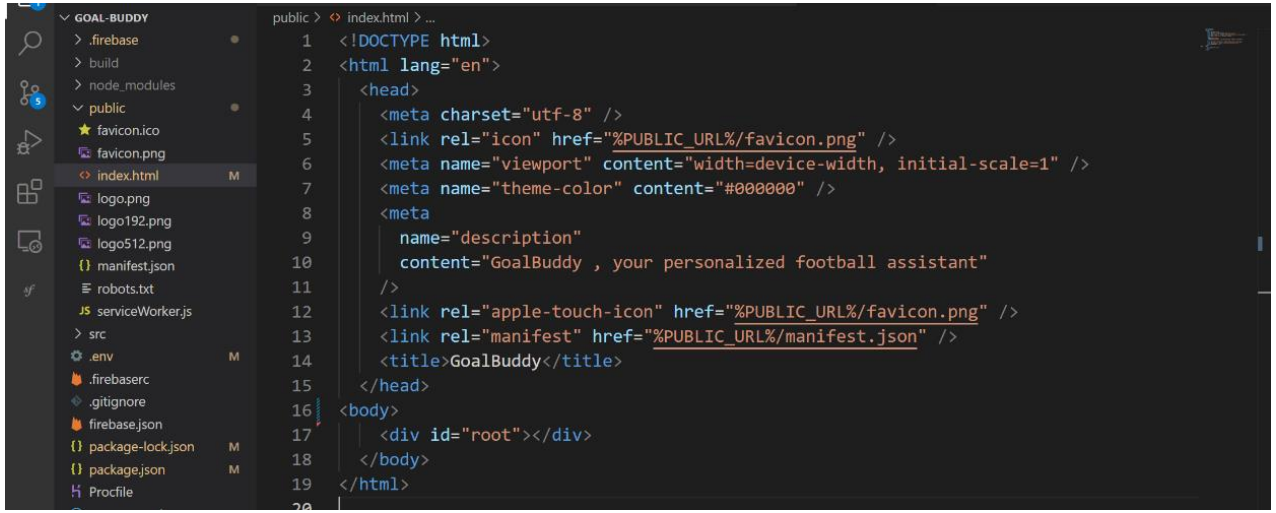
Description of the main directories and main files

Please note that we will not be able to illustrate the entire tree of our project... We will illustrate here the main directories and the main files of each directory :

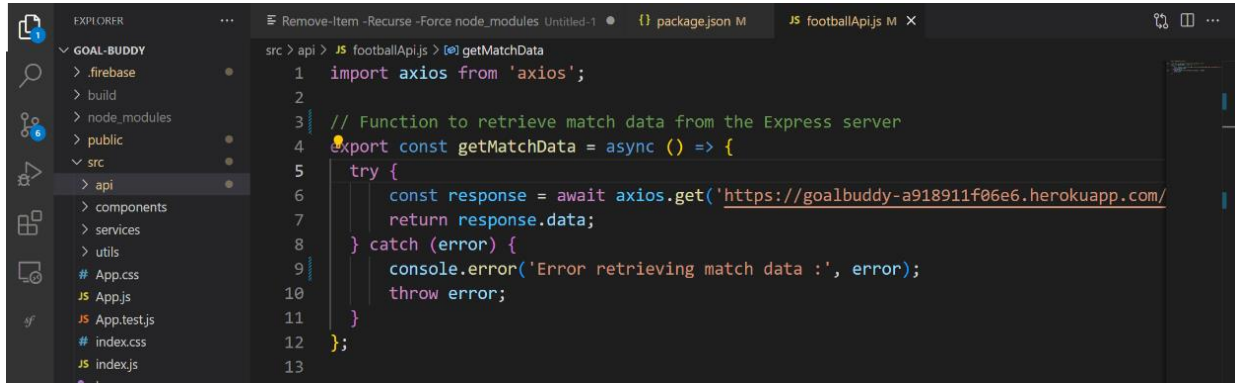
A. **public/**: Contains static public files that will be served directly by the server.
B. **src/**: Contains all the source code of the application.

- In the **public/** folder, the main file is **index.html, which** is the main **HTML** page of the application.:
- In the **/src** folder, we have the **api/** subfolder which contains the **footballApi.js** file, which contains code to retrieve match data from the Express server.

Index.html



footballApi.js

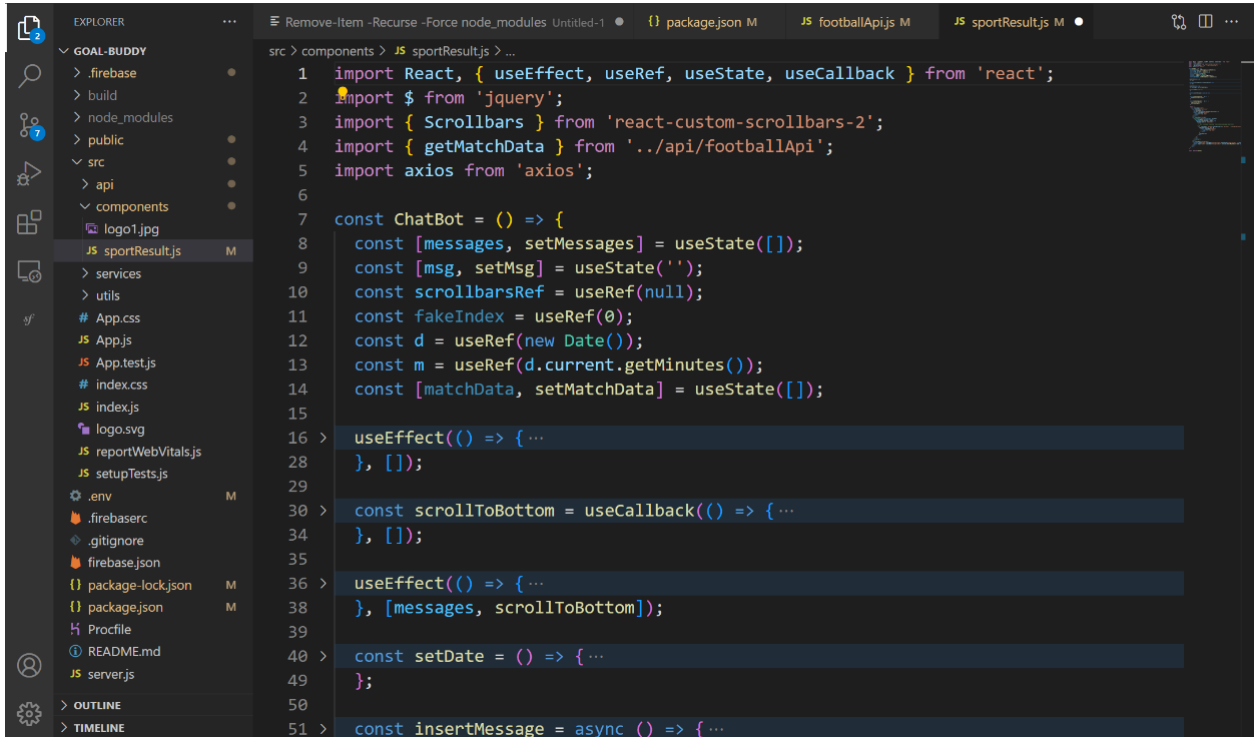


Description of the main directories and main files

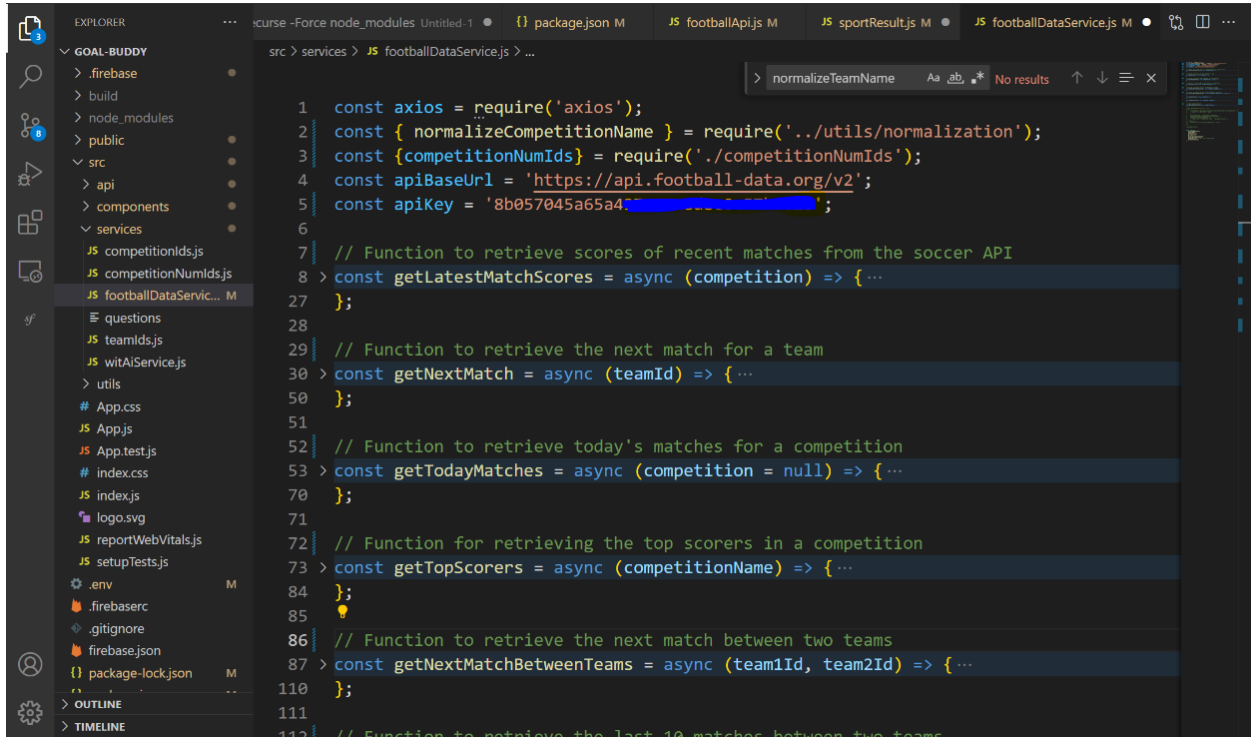
c) components/: Contained in the **src/** folder, contains the React components used in the application.

- In the **component/** folder we have the **sportResult.js** file, which plays a central role in the application by providing an interactive user interface to interact with the chatbot, It integrates API calls to retrieve match data and manages the state and rendering of messages reactively to user interface.
- In the **service/** folder we have the file **footballDataService.js,** which centralizes the interaction logic with the **football-data.org** API. It provides functions to retrieve various football information, such as match scores, upcoming matches, top scorers, and league standings.

sportResult.js



footballDataService.js

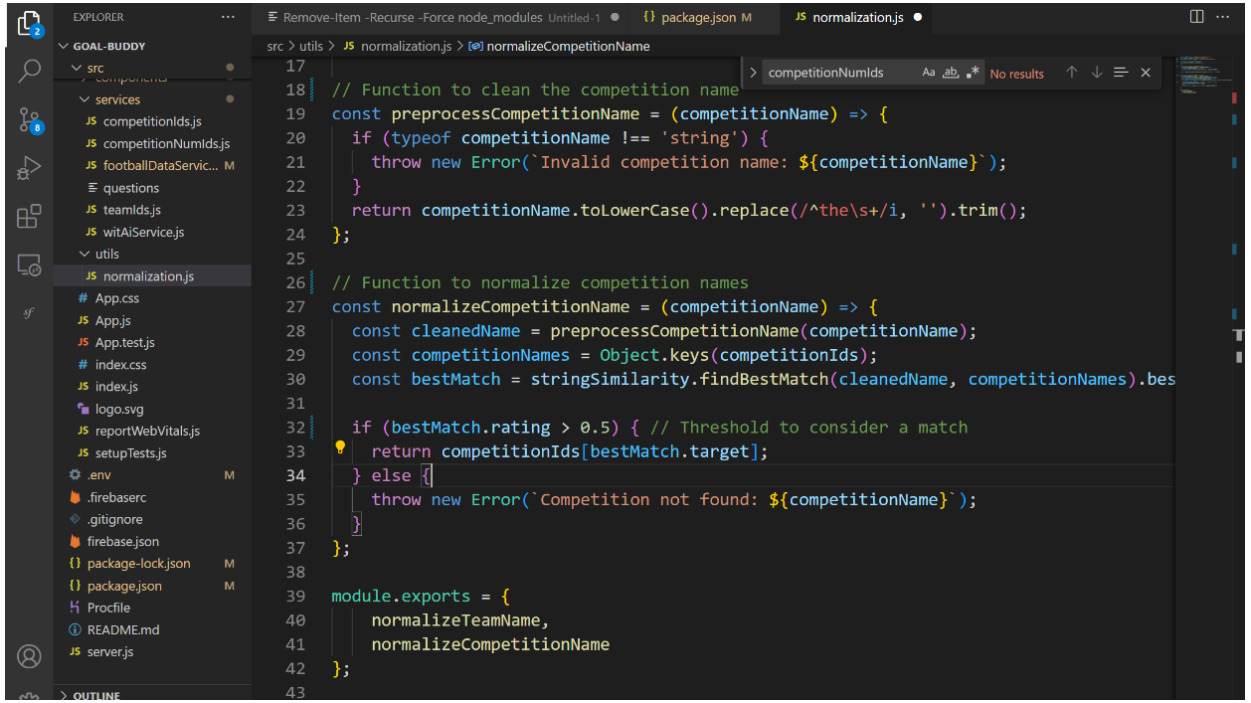


Description of the main directories and main files

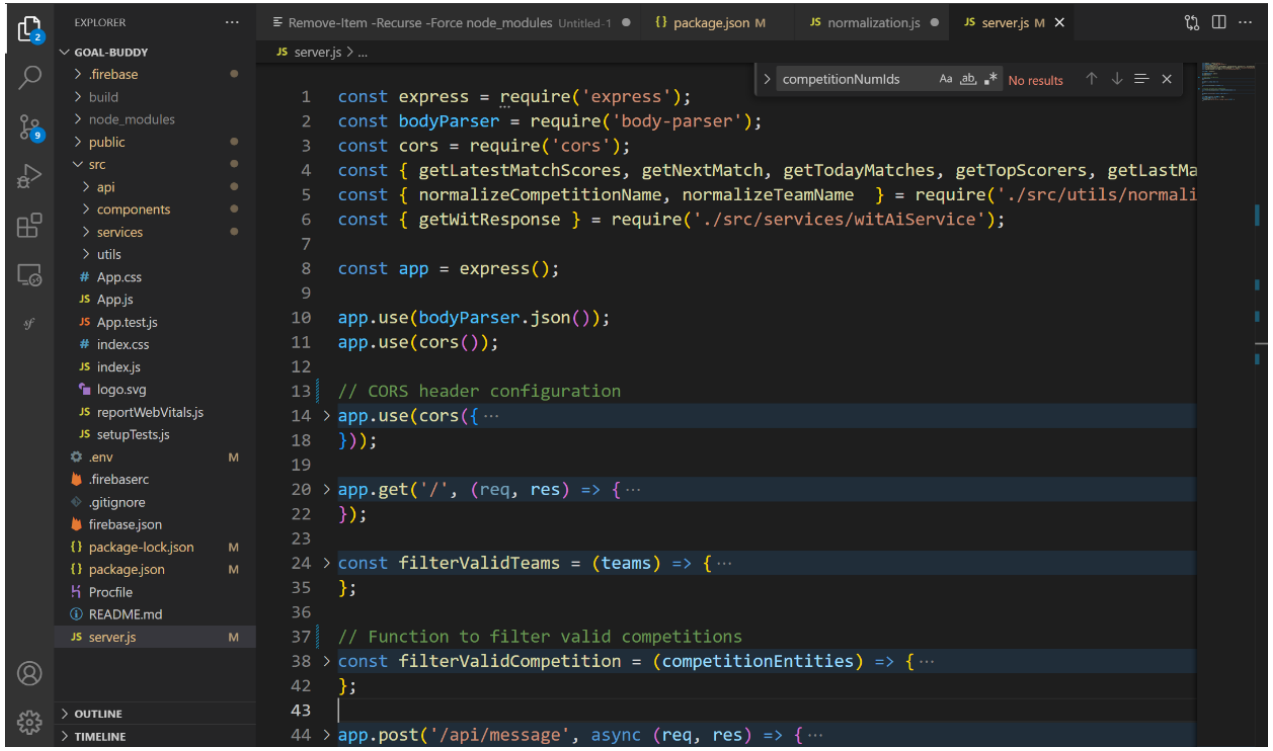
utils/: Contained in the **src/** folder, Contains utility files for various auxiliary functions.
Regarding the **App.js** and **index.js** files, there were no major changes in the basic configuration generated by the compilation of the "**create-react-app**" code, which allowed us to initialize our react project . However since they play an essential role in the execution of our program we have mentioned them.

- In **utilils/** folder we have the **normalization.js** file, which ensures that team and competition names are standardized before being used to interact with the **football-data.org** API. This helps minimize errors due to variations in names and ensures API requests are accurate.
- The file **server.js** is essential for bridging the web app frontend and backend services, particularly for processing user requests, interacting with the Wit.ai API to understand user intent, and retrieving relevant data from the **football-data.org** API.

normalization.js

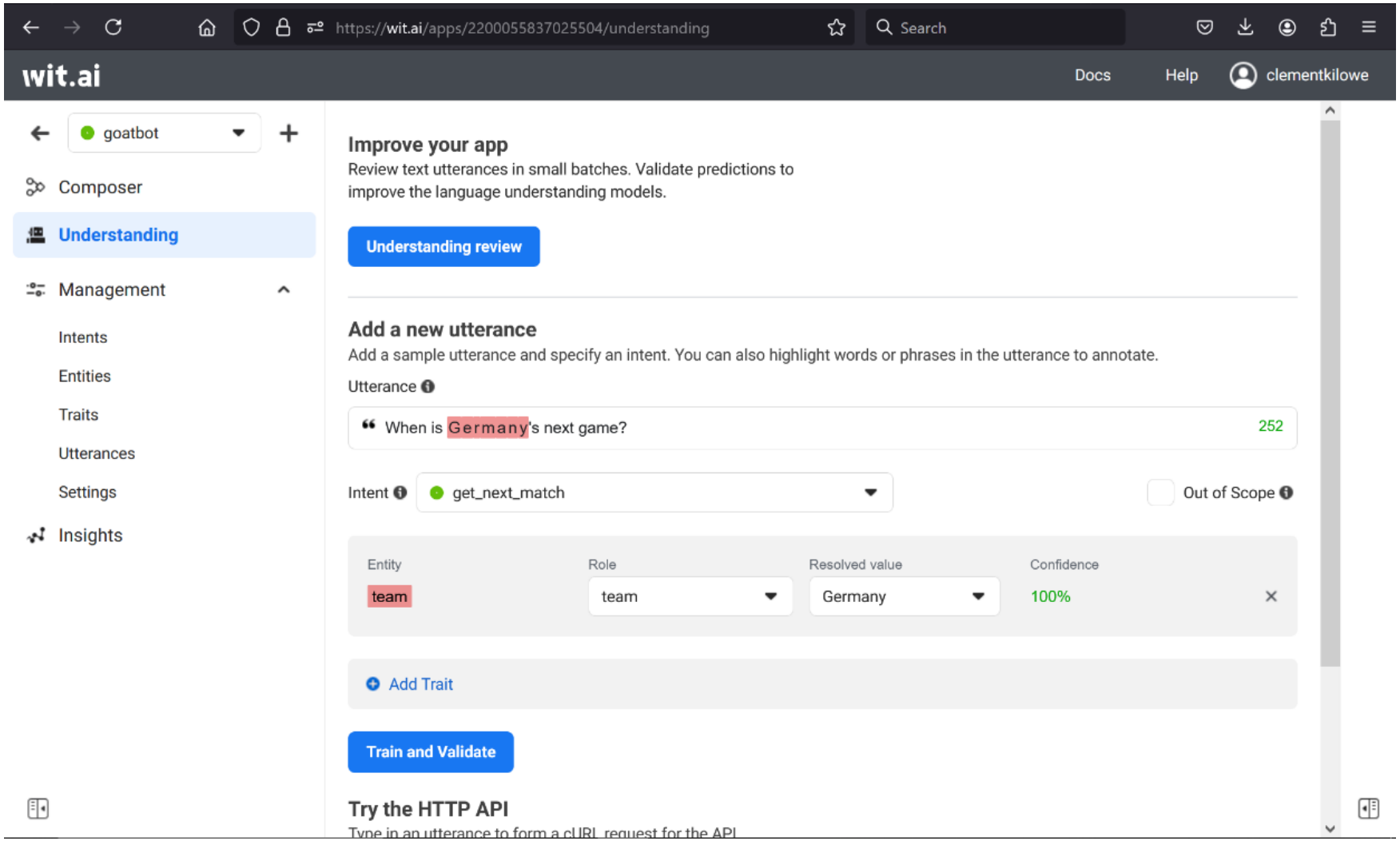


server.js



Wit.ai API

Wit.ai has been a valuable asset in the development of our app, adding a layer of artificial intelligence that significantly improves the user experience. Thus transforming our football score tracking application into a real intelligent assistant. Using **Wit.ai**, users can ask questions in natural language, and our app understands these questions and answers them in a meaningful way.



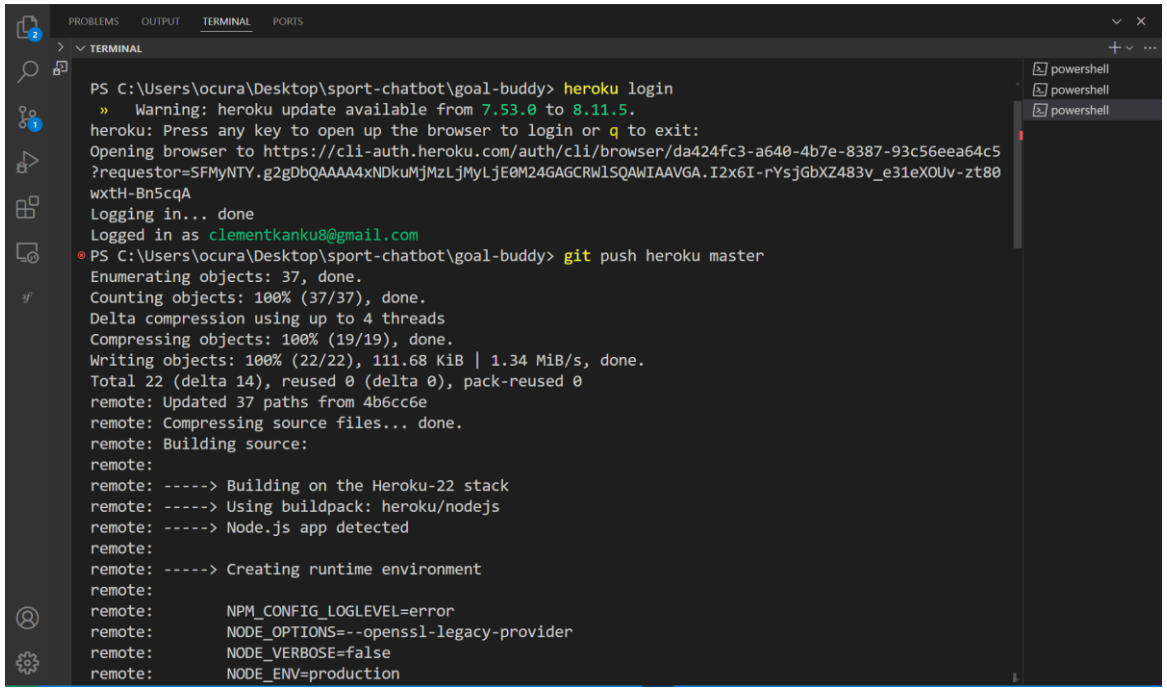
Deployment on Heroku and Google firestore

To make our application accessible to a wide audience, it is essential to host it on reliable and efficient platforms. We chose to use **Heroku** to host our backend server and **Firebase** to deploy our frontend application.

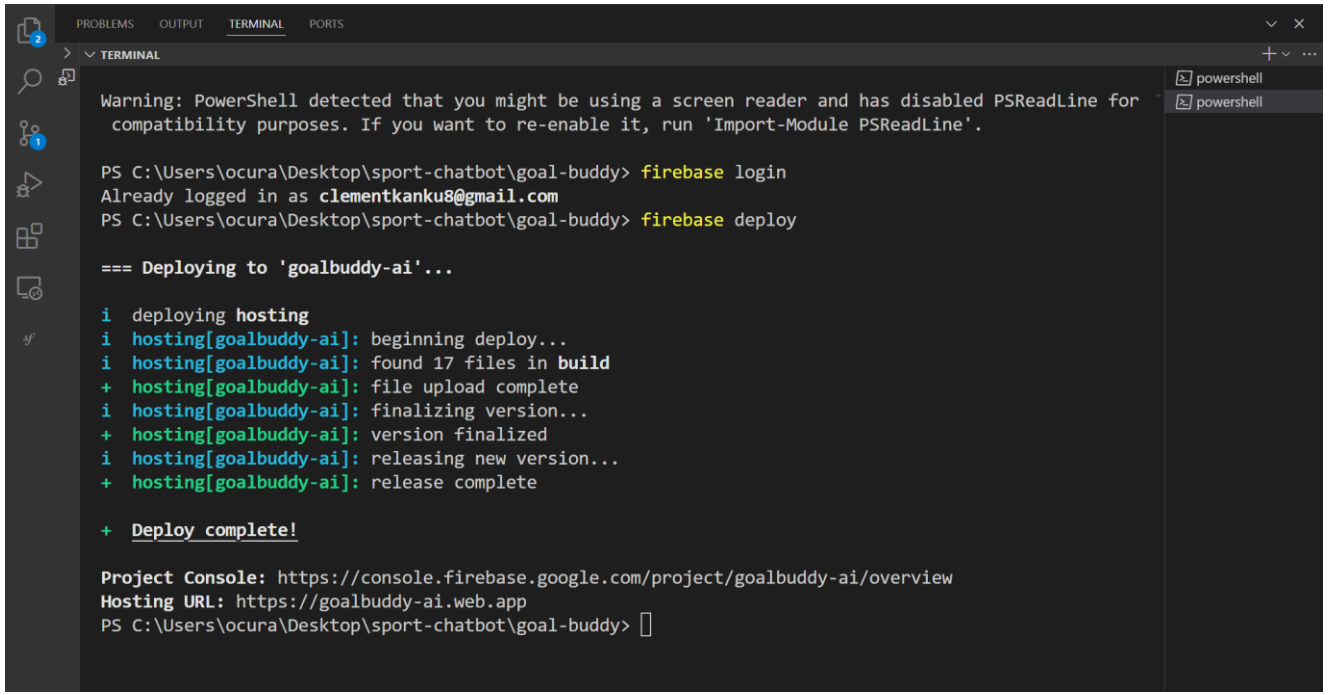
Heroku makes it quick and easy to deploy applications with a simple Git command. This greatly simplifies the process of bringing our **Nodejs** server online. To do this we went to the root of our project via command prompt, as illustrated:
https://goalbuddy-a918911f06e6.herokuapp.com/

Firebase offers fast and secure hosting for static web applications, with automatic **SSL** configuration. To host the font-end part of our application on **firestore**, we once again went to the root of our project as the illustration shows:
Link to our chatbot:https://goalbuddy-ai.web.app/

Heroku deployment



firebase deployment



conclusion

This practical work was an enriching adventure full of learning. We started with the choice of a simple idea (among the ones proposed to us): create an application capable of providing real-time information on football matches. Through this project, we explored and mastered different technologies, each bringing its own benefit.

Indeed, this project not only allowed us to apply the theoretical concepts of artificial intelligence and web development, but also to discover the practical challenges and real solutions in the development of a modern application. We gained a deep understanding of the technologies used and learned to overcome technical obstacles, thereby strengthening our software development skills.

Finally, this project reminds us of the importance of collaboration and innovation. By combining powerful tools and creative ideas, we were able to turn a simple idea into a useful and powerful application. **GoalBuddy** is the result of this synergy between technology and imagination, and we are proud of what we have accomplished.

Thank you for your attention and support throughout this journey. We hope you find **GoalBuddy** as exciting and useful as we enjoyed developing it.