Build an analytics dashboard with React

Developer at sportsee, a fast growing coaching startup. The company is launching a new version of the "profile" page for a user. It will allow the user to track the number of sessions performed as well as the number of calories burned

The project uses React, a JavaScript library for building interactive user interfaces.

Project dependencies include Axios for making HTTP requests, Recharts for data visualization and npm for managing dependencies and build and startup scripts.

Node.js must also be installed. Dependencies can be installed using the npm install command in the project root directory.

Node.js version v18.14.0
npm version 8.4.1
Visual studio code version 1.75.1
React version 18.2.0
Recharts version 2.4.2
Sass version 1.58.1
Axios version 1.3.3

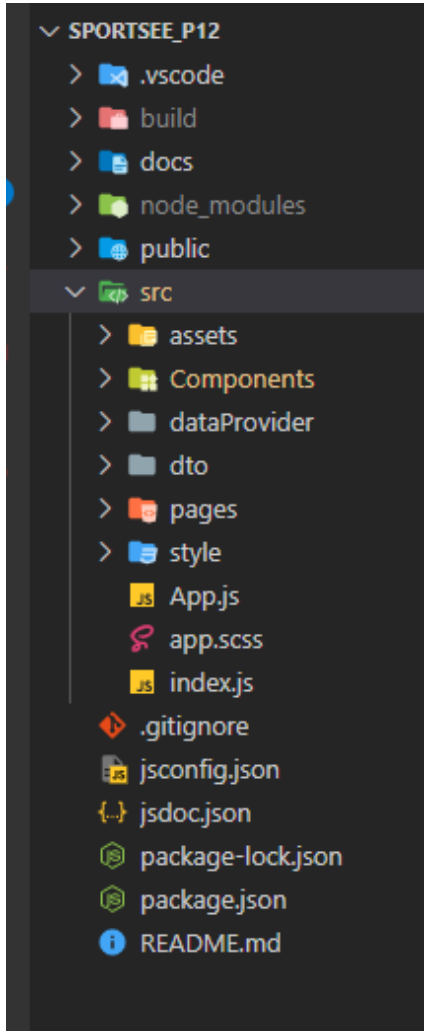
SportSee

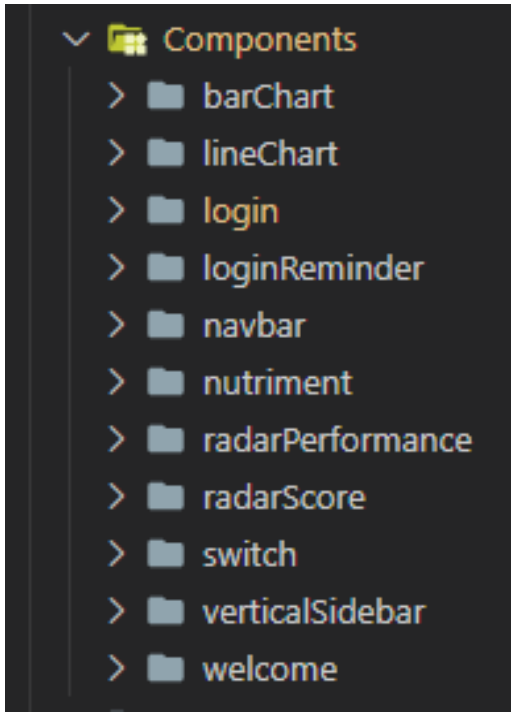
OBJECTIVES SPECIFICATIONS

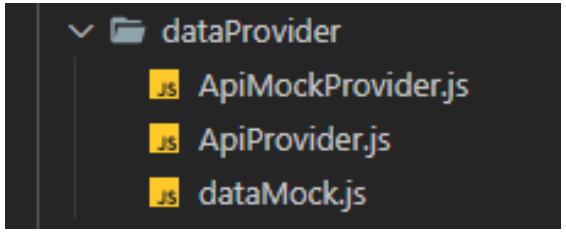
- make the profile page with React
- Graphics integration
- Responsive Desktop 1024 x 780 pixels
- Retrieve data from the backend
- Use node js
- Realize the data mock
- Integrate the api
- Documentation in english
- Integrate PropTypes for each component

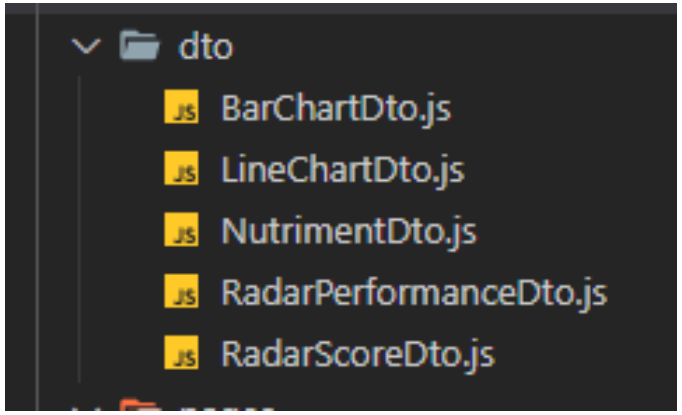
SportSee

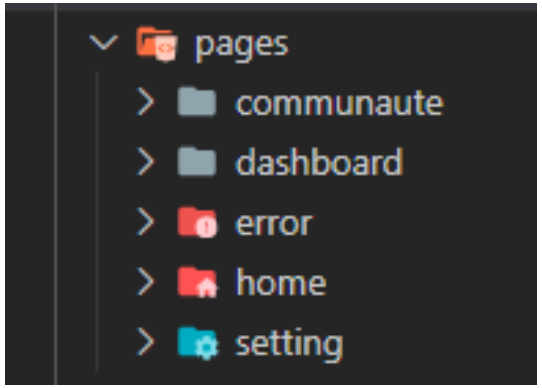
PROJECT ARCHITECTURES









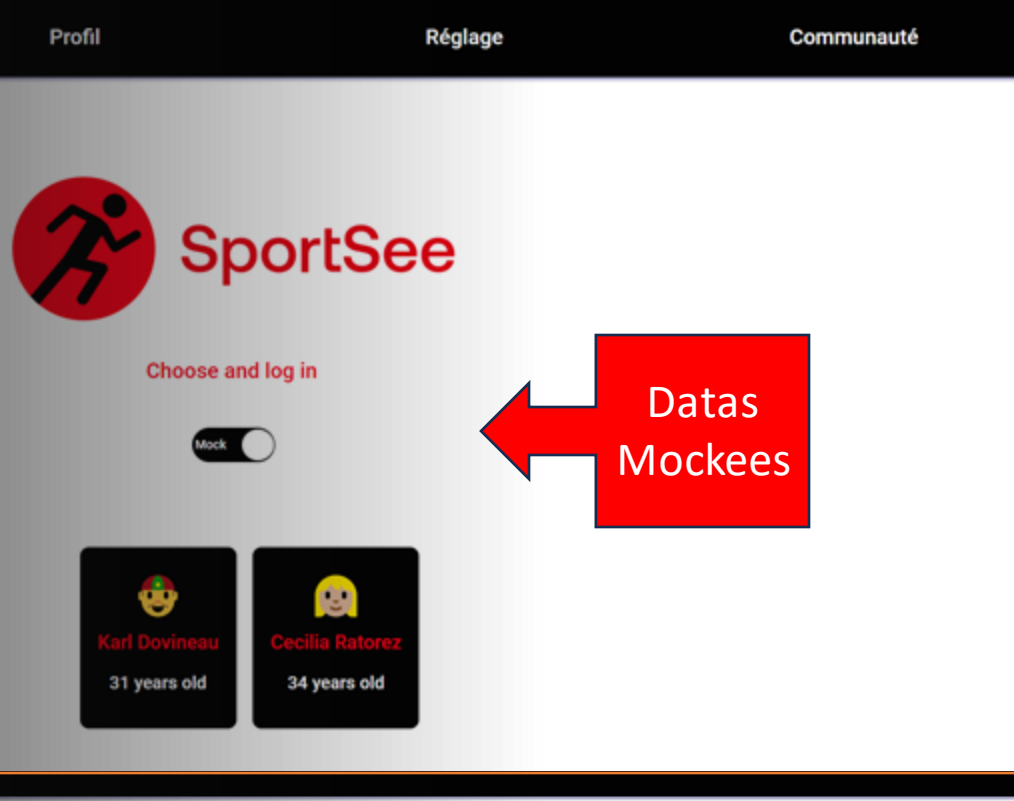


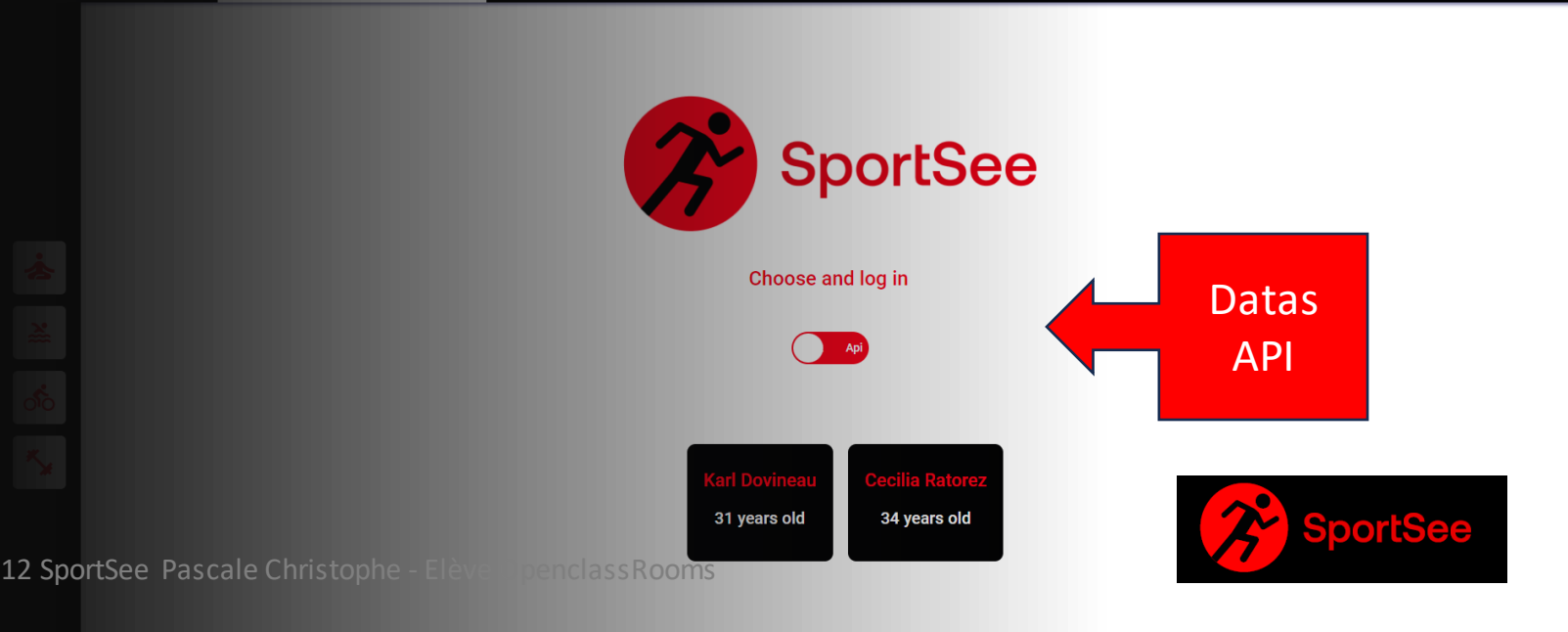
```jsx
const App = () => {
  return (
    <header className="routePage">
      <Navbar />
      <VerticalSidebar />
      <section className="content">
        <Routes>
          <Route path="/" element={<Home  />} />
          <Route path="/dashboard/:userId/:isDemo" element={<Dashboard  />}/>
          <Route path="/setting/" element={<Setting />} />
          <Route path="/communaute/" element={<Communaute />} />
          <Route path="*" element={<Error />} />
          <Route path="/login-reminder" element={<LoginReminder />} />
        </Routes>
      </section>
    </header>
  );
};
```

Routes - App.js

SportSee

Home


Accueil
Profil
Réglage
Communauté
SportSee
Choose and log in
Mock
Karl Dovineau
31 years old
Cecilia Ratorez
34 years old
Datas Mockees


SportSee
Choose and log in
Api
Karl Dovineau
31 years old
Cecilia Ratorez
34 years old
Datas API
SportSee

LOADER Dashboard page

HOOK - CREATE STATE

```
21    const [isLoading, setIsLoading] = useState(true);
```

```
61        // Simulate loading for 5 seconds
62        await new Promise((resolve) => setTimeout(resolve, 1000));
```

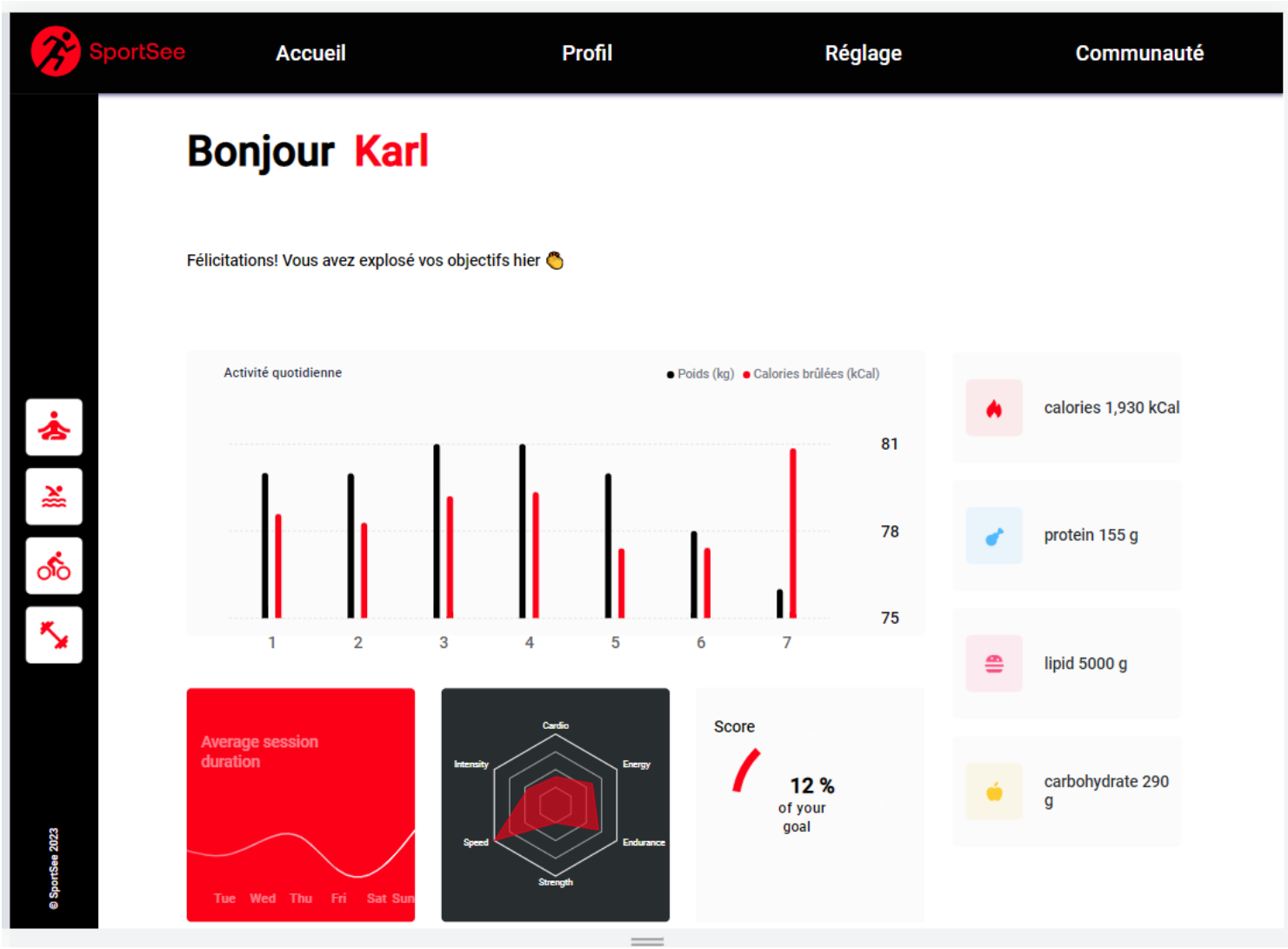
Creates an asynchronous promise that makes the code wait for one second (1000 milliseconds) before continuing

```
70          setIsLoading(false);
71        } catch (error) {
72          setIsLoading(false);
73        }
74      }
```

Finally, the line setIsLoading(false) sets the variable isLoading to false, indicating that the data loading is finished.

SportSee

Dashboard



SportSee

```
class ApiMockProvider {
  /**
   * Retrieves user activities by user ID.
   * @param {number} userId - The user ID.
   * @returns {BarChartDto} - User activities in the form of BarChartDto
   * @memberof ApiMockProvider
   */
  getActivitiesByUserId(userId) {
    const userSessions = [];
    const currentUser = USER_ACTIVITY.find(
      (user) => user.userId === parseInt(userId)
    );
    if (currentUser) {
      currentUser.sessions.forEach((session) => {
        const day = session.day.split("-");
        const formattedDate = `${day}`;
        userSessions.push({
          day: formattedDate,
          kilogram: session.kilogram,
          calories: session.calories,
        });
      });
    }
    return new BarChartDto(
      userSessions,
      "Day",
      "Kilograms",
      "Calories"
    );
  }
```

Class ApiMockProvider

- The ApiMockProvider class is a mock data provider that contains several methods for retrieving user data. These methods include
- Each method takes a user ID as a parameter and returns the corresponding data in the form of a specific data transfer object (DTO).
- The class imports several DTOs, including RadarPerformanceDto, NutrimentDto, LineChartDto, BarChartDto, and RadarScoreDto.
- In summary, the ApiProvider class provides convenient methods to retrieve various user information from an API using Axios and DTO objects to structure the retrieved data.

SportSee

Class ApiProvider

The ApiProvider class is a class that provides functions for retrieving user data from an API. It contains several methods for retrieving user data.

Each method takes a user ID as a parameter and returns the corresponding data in the form of a specific data transfer object (DTO). The class imports several DTOs, including

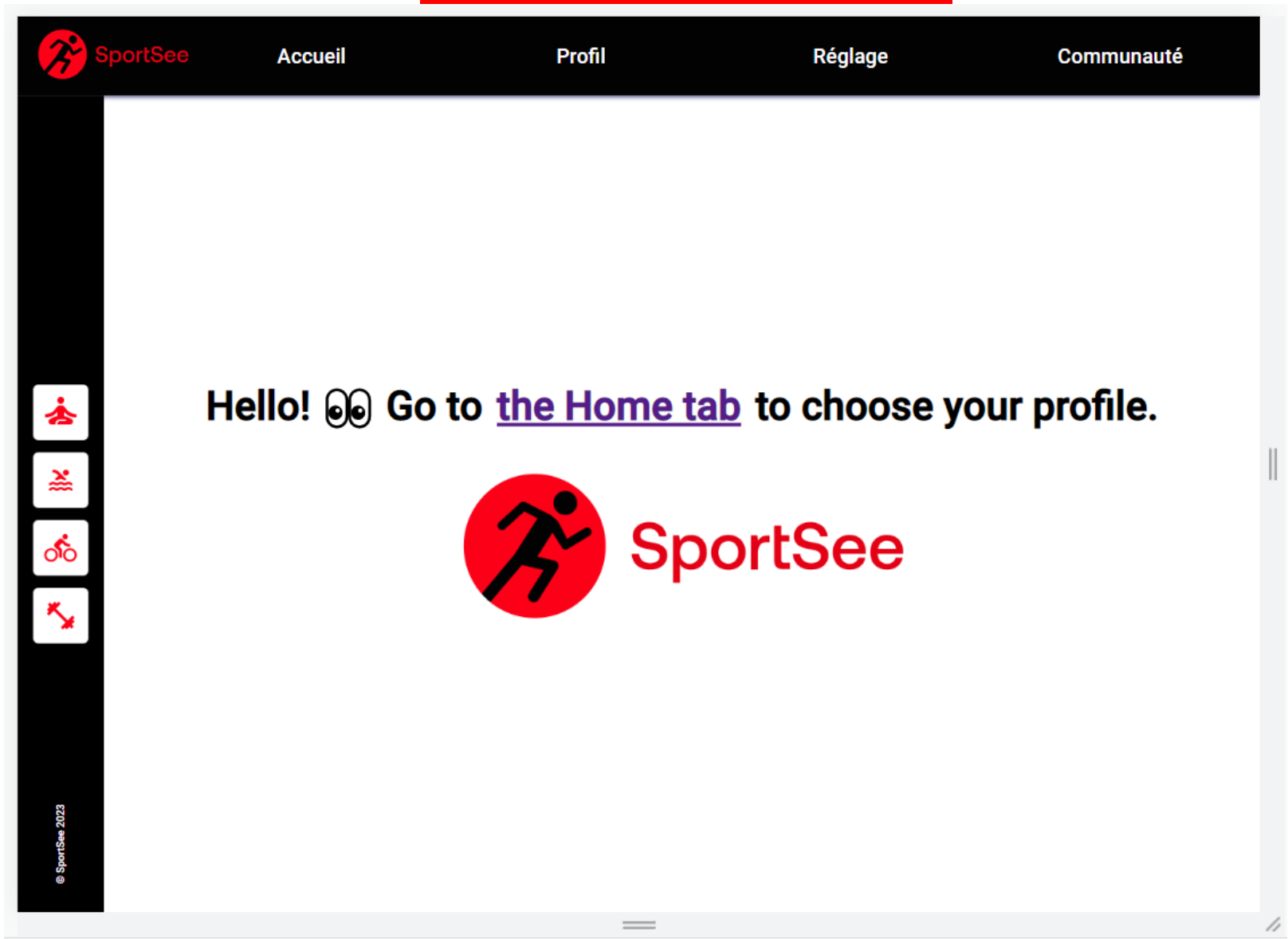
The class uses Axios to make HTTP requests to retrieve user data from the API. It also contains a handleError method for handling errors when retrieving user data.

In summary, the ApiMockProvider class provides functions to retrieve simulated user data. These functions are used to test and simulate calls to a real API, providing mock data for development and debugging purposes.

```
class ApiProvider {
  constructor() {
    this.baseURL = BASE_URL;
  }

  /**
   * Handles errors when retrieving user data.
   * @param {Error} error - The error generated when retrieving the data.
   * @throws {Error} - An error indicating that user data cannot be fetched.
   */
  handleError(error) {
    console.log("Error fetching user data: ", error);
    console.error("Error fetching user data: ", error);
    throw new Error("Unable to fetch user data");
  }

  /**
   * Retrieves the user's first name by user ID.
   * @param {string} userId - The user ID.
   * @returns {Promise<string|null>} - A promise that resolves with the user's first name
   */
  async getUserNameByUserId(userId) {
    // Check if the request URL is correct
    console.log("Request URL: ", this.baseURL + userId);

    // Make a GET request using axios
    return axios
      .get(this.baseURL + userId)
      .then((response) => {
        console.log("response: ", response);

        return response.data &&
          response.data.data &&
          response.data.data.userInfos &&
          response.data.data.userInfos.firstName
          ? response.data.data.userInfos.firstName
          : null;
      })
```

SportSee

Profile



Login to be directed to your profile

CONCLUDE

A React analytics dashboard has been developed for sports coaching startup SPORTSEE.

The project uses a backend for HTTP calls and integrates an API to retrieve data.

A data modeling class was created to ensure that API data is always formatted before being used.

The Récharts graphics library was used for the graphics part.

The project respects several technical constraints, such as the integration of Proptypes for each React compound.

The end result is an attractive, interactive analysis dashboard dedicated to sports coaching.

Translated with www.DeepL.com/Translator (free version)

SportSee