Construire un taleau de Bord analytique avec React

Développeur chez sportsee, une startup de coaching en pleine
croissance.
L'entreprise lance une nouvelle version de la page "profil" d'un utilisateur. Elle permettra à l'utilisateur de suivre le nombre de séances effectuées ainsi que le nombre de calories brûlées

Résumé des conditions

•Langage : React, bibliothèque JavaScript pour la construction d'interfaces utilisateur interactives. React permet de développer des applications web réactives à l'aide de composants réutilisables.

• Dépendances :

•Axios : Bibliothèque JavaScript utilisée pour effectuer des requêtes HTTP à partir d'un navigateur ou d'un serveur. . Axios est utilisé dans le projet pour effectuer des appels API et récupérer des données.

•Recharts : Recharts est une bibliothèque de visualisation de données basée sur React. Recharts est utilisé dans le projet pour afficher des données sous forme de graphiques.

• npm : (Node Package Manager) Il est utilisé pour installer les dépendances du projet, gérer les versions des paquets et s'occuper des scripts de construction et de démarrage du projet.

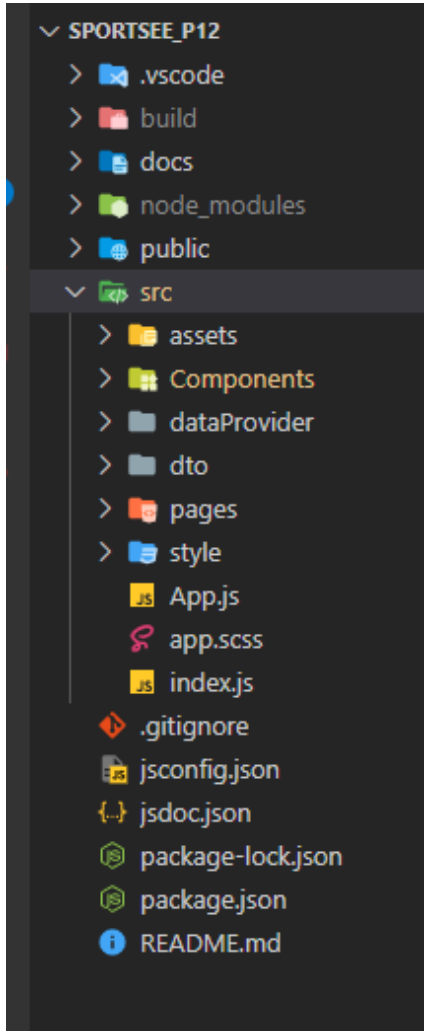
•Node js

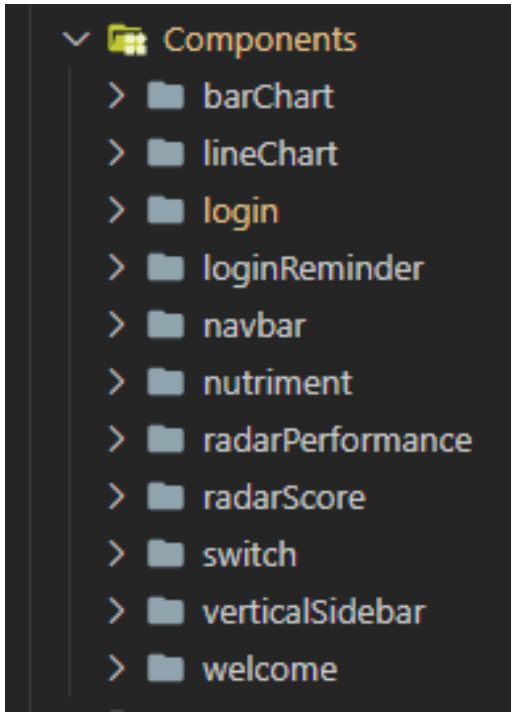
Il doit être installé à l'aide de la commande npm install dans le répertoire racine du projet. Cette commande téléchargera et installera les paquets nécessaires dans le dossier node_modules du projet.

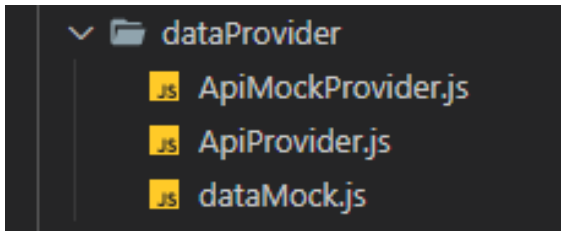
OBJECTIF CAHIER DES CHARGES

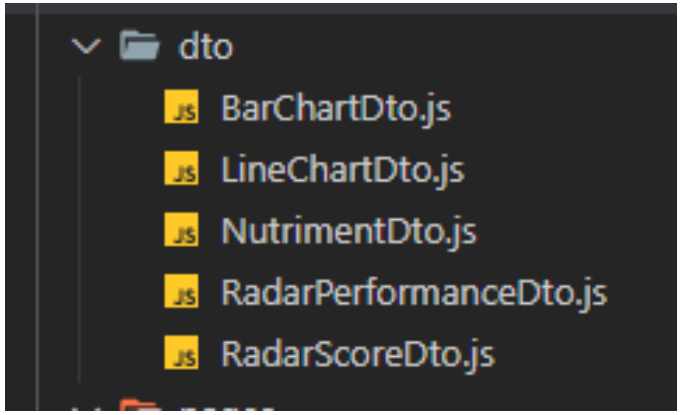
- Refaire la page Profil avec React
- Intégration des graphiques
- Responsive 1024 x 780 pixels
- Récuperer les donnees du backend
- Réaliser le mock des données
- Utiliser node Js
- Intégrer l'api
- Réaliser le mock des données
- Documentation en anglais
- Intégrer des PropTypes pour chacun des composants

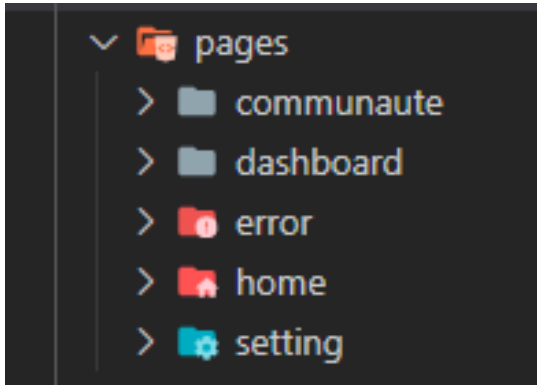
ARCHITECTURE du Projet







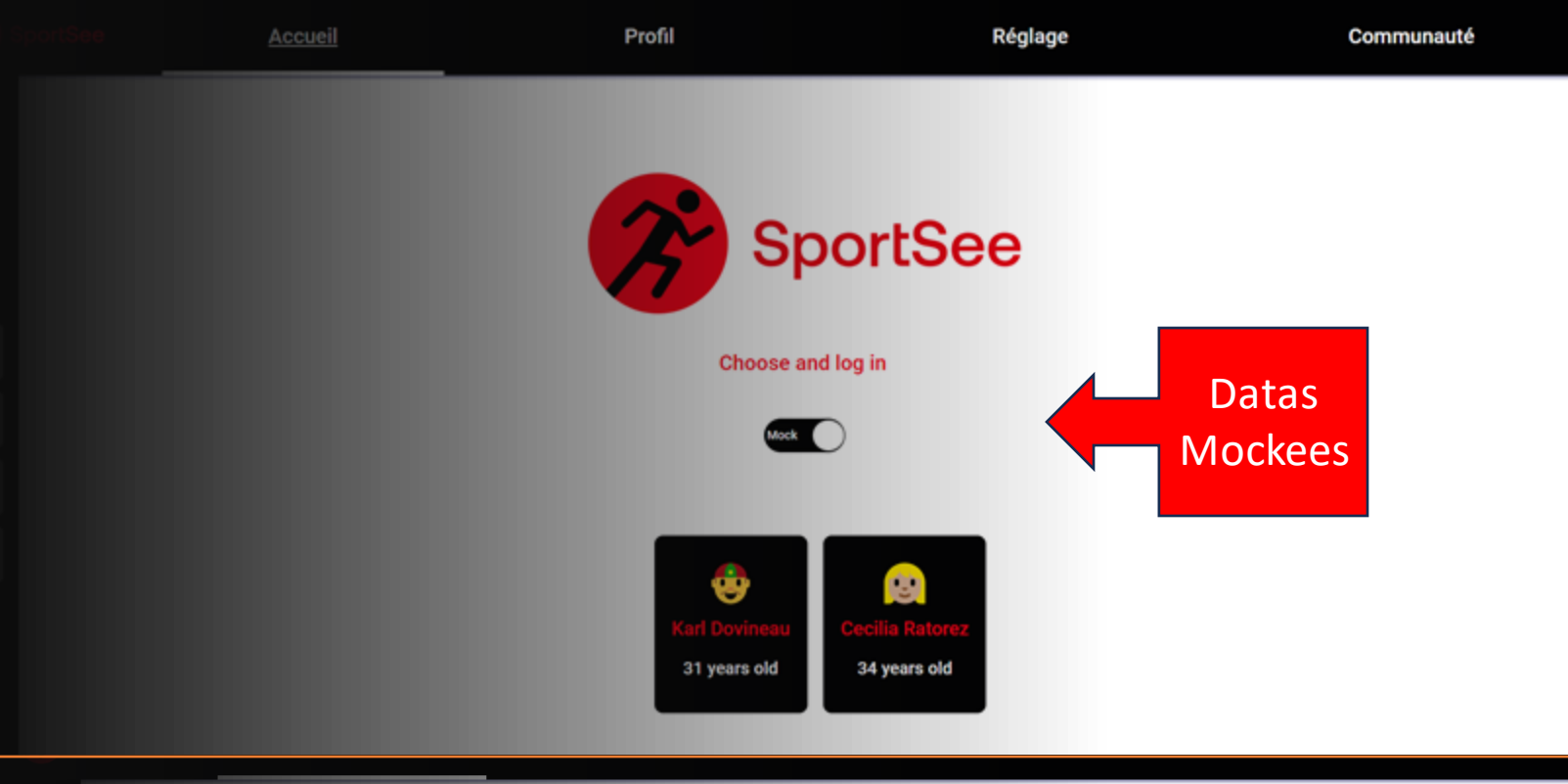


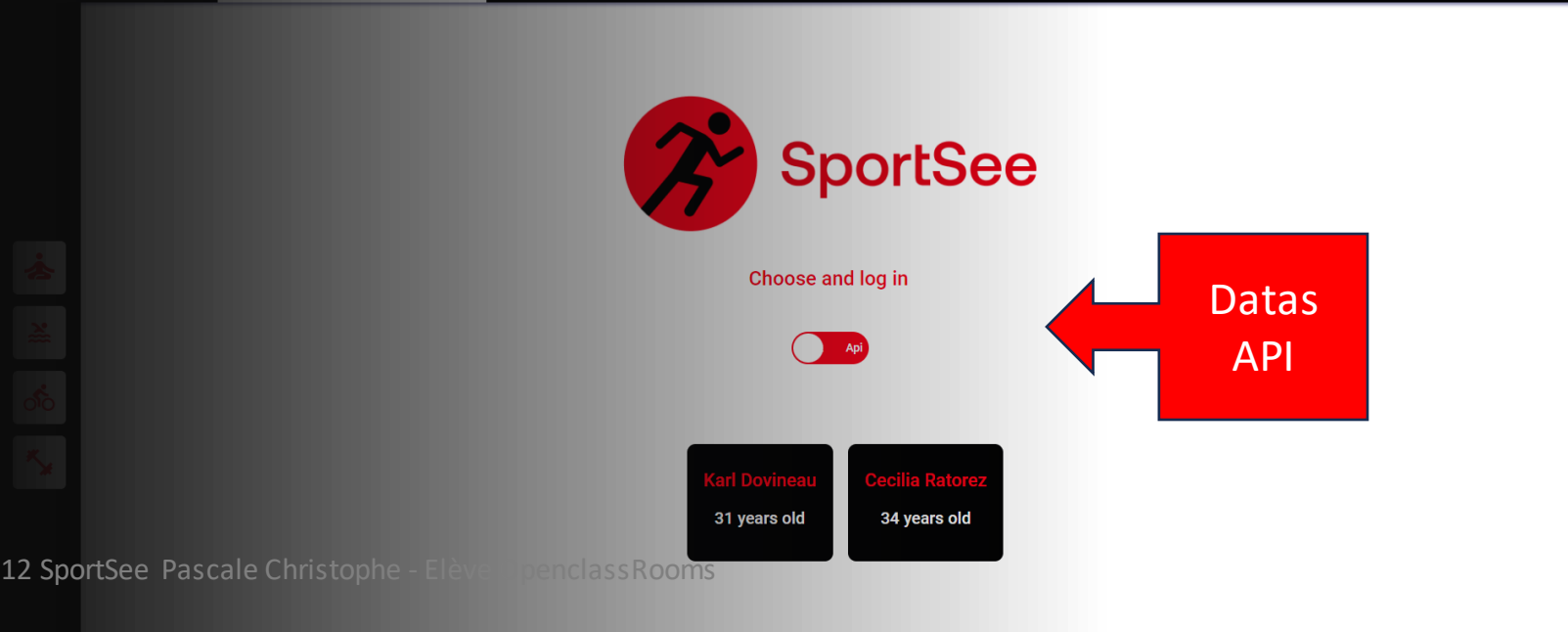


```jsx
const App = () => {
  return (
    <header className="routePage">
      <Navbar />
      <VerticalSidebar />
      <section className="content">
        <Routes>
          <Route path="/" element={<Home  />} />
          <Route path="/dashboard/:userId/:isDemo" element={<Dashboard  />}/>
          <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

ACCUEIL


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

LOADER Page Dashboard

Hook

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

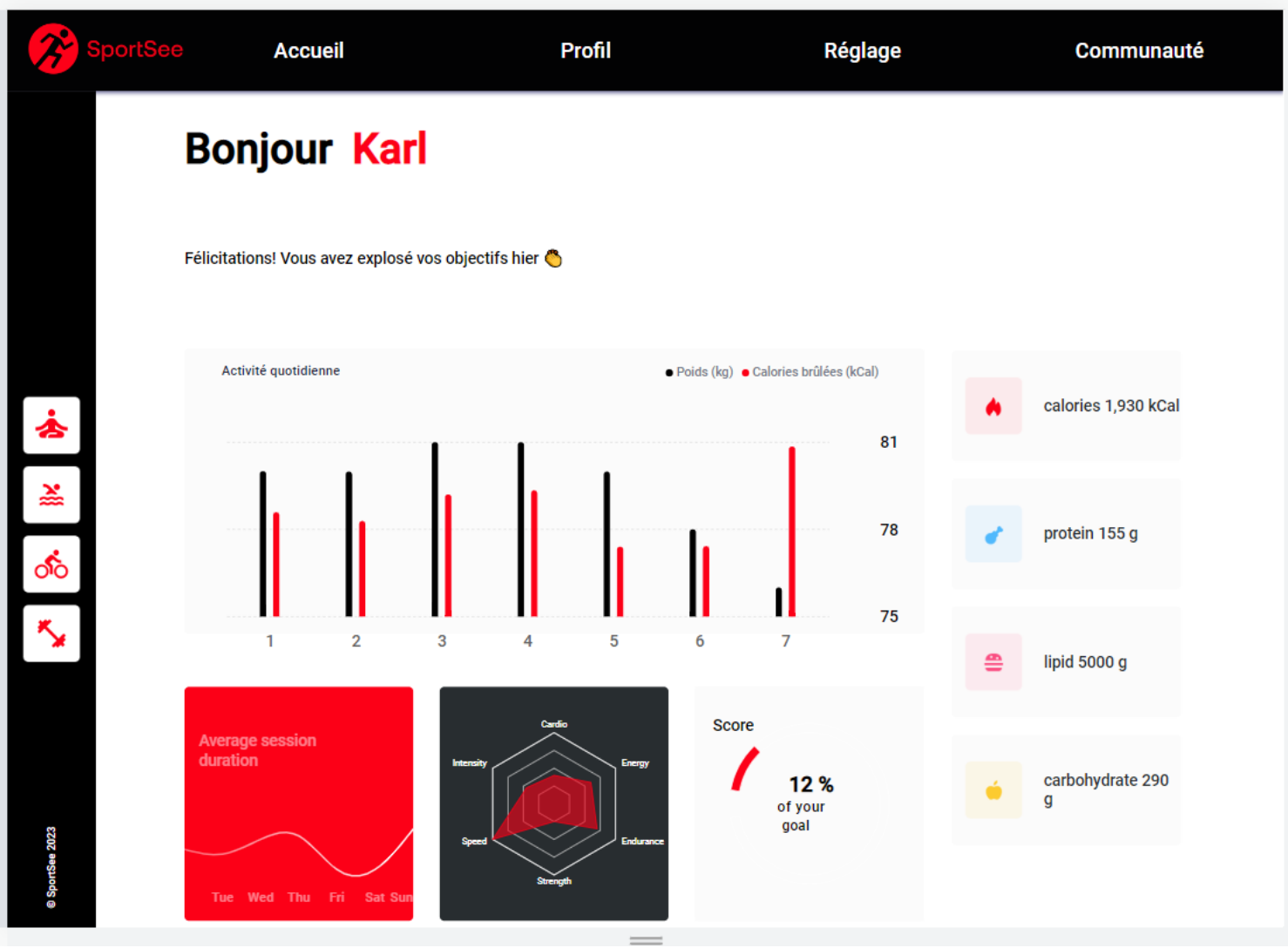
```
61        // Simulate loading for 5 seconds
62        await new Promise((resolve) => setTimeout(resolve, 1000));
```

Crée une promesse asynchrone qui fait attendre pendant une seconde (1000 millisecondes) avant de continuer.

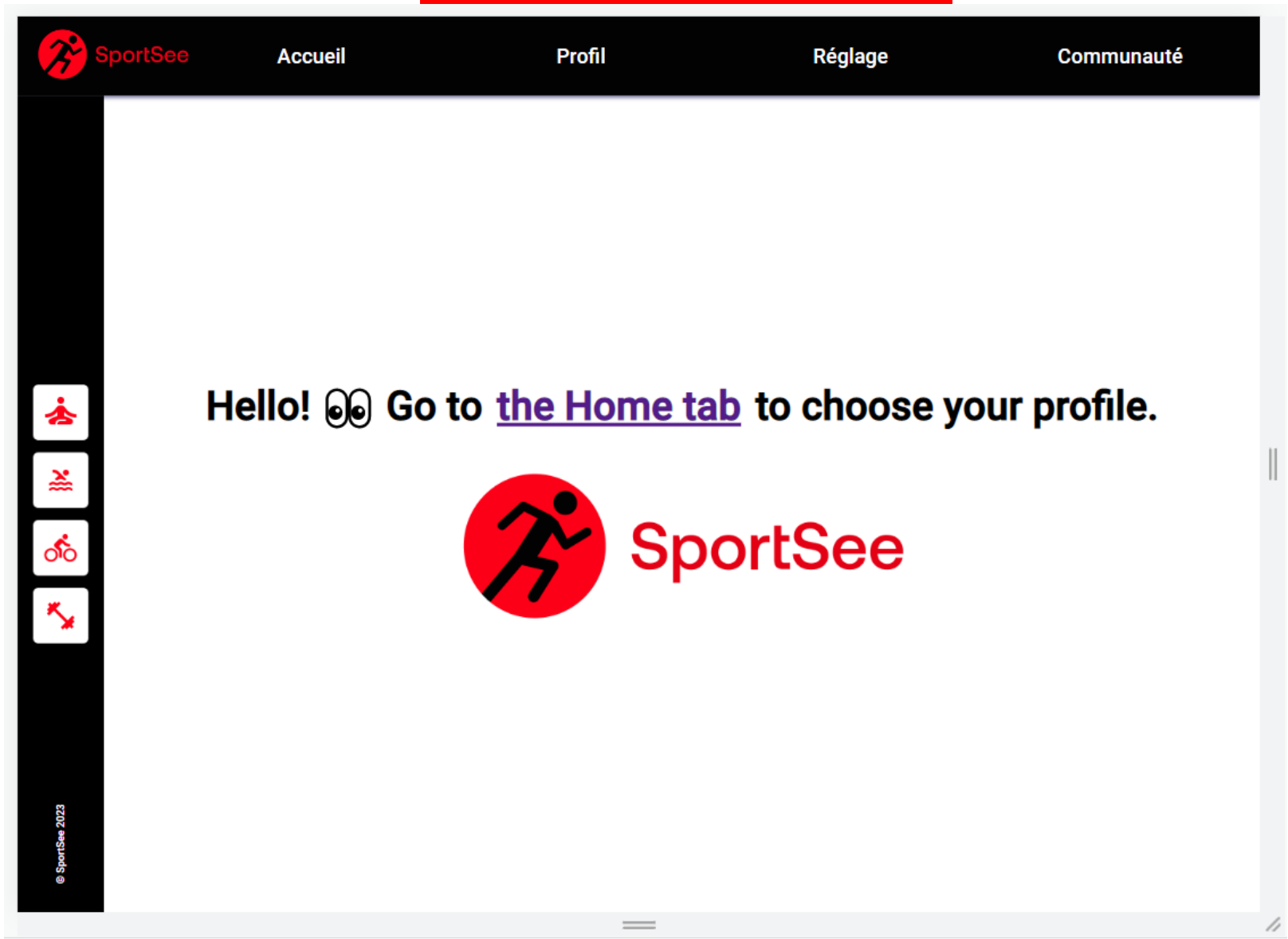
```
70          setIsLoading(false);
71        } catch (error) {
72          setIsLoading(false);
73        }
74      }
```

setIsLoading(false) attribue la valeur false à la variable isLoading
= indique le chargement des données terminé.

Tableau de Bord



Profil



Connectez-vous pour être redirigé vers votre profil

```
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

- La classe **`ApiMockProvider`** est une classe qui fournit des fonctions pour récupérer des données fictives (mockées) liées à un utilisateur. Elle offre des méthodes pour récupérer différentes informations.
- Aussi conçue pour faciliter la récupération de données fictives pour des tests ou des simulations.

Class DataProvider

La classe **`ApiProvider`** est une classe qui fournit des fonctions pour récupérer des données utilisateur à partir d'une API. Elle utilise la bibliothèque Axios pour effectuer des requêtes HTTP.
La classe comporte plusieurs méthodes pour afficher des graphiques

Elle est conçue pour faciliter l'accès aux données utilisateur à partir d'une API en utilisant des méthodes simples et en retournant les résultats sous forme d'objets DTO spécifiques.

```
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;
      })
```