FitFlex: Your Personal Fitness Compassion

Welcome to the forefront of fitness exploration with SB Fitzz! Our innovative fitness app is meticulously designed to revolutionize the way you engage with exercise routines, catering to the diverse interests of both fitness enthusiasts and seasoned workout professionals. With a focus on an intuitive user interface and a comprehensive feature set, SB Fitzz is set to redefine the entire fitness discovery and exercise experience.

Crafted with a commitment to user-friendly aesthetics, SB Fitzz immerses users in an unparalleled fitness journey. Effortlessly navigate through a wide array of exercise categories with features like dynamic search, bringing you the latest and most effective workouts from the fitness world.

Elevate your fitness exploration with SB Fitzz, where every exercise becomes a gateway to a world of wellness waiting to be discovered and embraced. Trust SB Fitzz to be your reliable companion on the journey to staying connected with a fit and active lifestyle.

Project Goals and Objectives:

The overarching aim of SB Fitzz is to offer an accessible platform tailored for individuals passionate about fitness, exercise, and holistic well-being.

Our key objectives are as follows:

1. **User-Friendly Experience:** Develop an intuitive interface that facilitates easy navigation, enabling users to effortlessly discover, save, and share their preferred workout routines.

2. **Comprehensive Exercise Management:** Provide robust features for organizing and managing exercise routines, incorporating advanced search options for a personalized f itness experience.

3. **Technology Stack:** Harness contemporary web development technologies, with a focus on React.js, to ensure an efficient and enjoyable user experience.

Features of SB Recipess:

1. **Exercises from Fitness API:** Access a diverse array of exercises from reputable fitness APIs, covering a broad spectrum of workout categories and catering to various fitness goals.

2. **Visual Exercise Exploration:** Engage with workout routines through curated image galleries, allowing users to explore different exercise categories and discover new f itness challenges visually.

3. **Intuitive and User-Friendly Design:** Navigate the app seamlessly with a clean, modern interface designed for optimal user experience and clear exercise selection.

4. **Advanced Search Feature:** Easily find specific exercises or workout plans through a powerful search feature, enhancing the app's usability for users with

varied fitness preferences.

PRE-REQUISITES:

Here are the key prerequisites for developing a frontend application using React.js:

1. **Node.js and npm:**
   Node.js is a powerful JavaScript runtime environment that allows you to run JavaScript code on the local environment. It provides a scalable and efficient platform for building network applications.
   Install Node.js and npm on your development machine, as they are required to run JavaScript on the server-side.

- Download: https://nodejs.org/en/download/

- Installation instructions: https://nodejs.org/en/download/package-manager/

2. **React.js:**
   React.js is a popular JavaScript library for building user interfaces. It enables developers to create interactive and reusable UI components, making it easier to build dynamic and responsive web applications.
   Install React.js, a JavaScript library for building user interfaces.

- Create a new React app:
  npx create-react-app my-react-app
  Replace my-react-app with your preferred project name.

- Navigate to the project directory:
  cd my-react-app

- Running the React App:
  With the React app created, you can now start the development server and see your React application in action.

- Start the development server:
  npm start
  This command launches the development server, and you can access your React app at http://localhost:3000 in your web browser.

3. **HTML, CSS, and JavaScript:** Basic knowledge of HTML for creating the structure of your app, CSS for styling, and JavaScript for client-side interactivity is essential.

4. **Version Control:** Use Git for version control, enabling collaboration and tracking changes throughout the development process. Platforms like GitHub or Bitbucket can host your repository.
   - Git: Download and installation instructions can be found at: https://git scm.com/downloads

5. **Development Environment:** Choose a code editor or Integrated Development Environment (IDE) that suits your preferences, such as Visual Studio Code, Sublime Text, or WebStorm.
    - Visual Studio Code:
      Download from https://code.visualstudio.com/download
    - Sublime Text: Download from https://www.sublimetext.com/download
    - WebStorm:
      Download from https://www.jetbrains.com/webstorm/download

To get the Application project from drive:
Follow below steps:

6. **Get the code:**
    - Download the code from the drive link given below:

https://drive.google.com/drive/folders/14f9eBQ5W7VrLdPhP2W6PzOU_HCy8UMex?usp=sharing

Install Dependencies:

- Navigate into the cloned repository directory and install libraries:

```
cd fitness-app-react
npm install
```

7. **Start the Development Server:**
    - To start the development server, execute the following command:

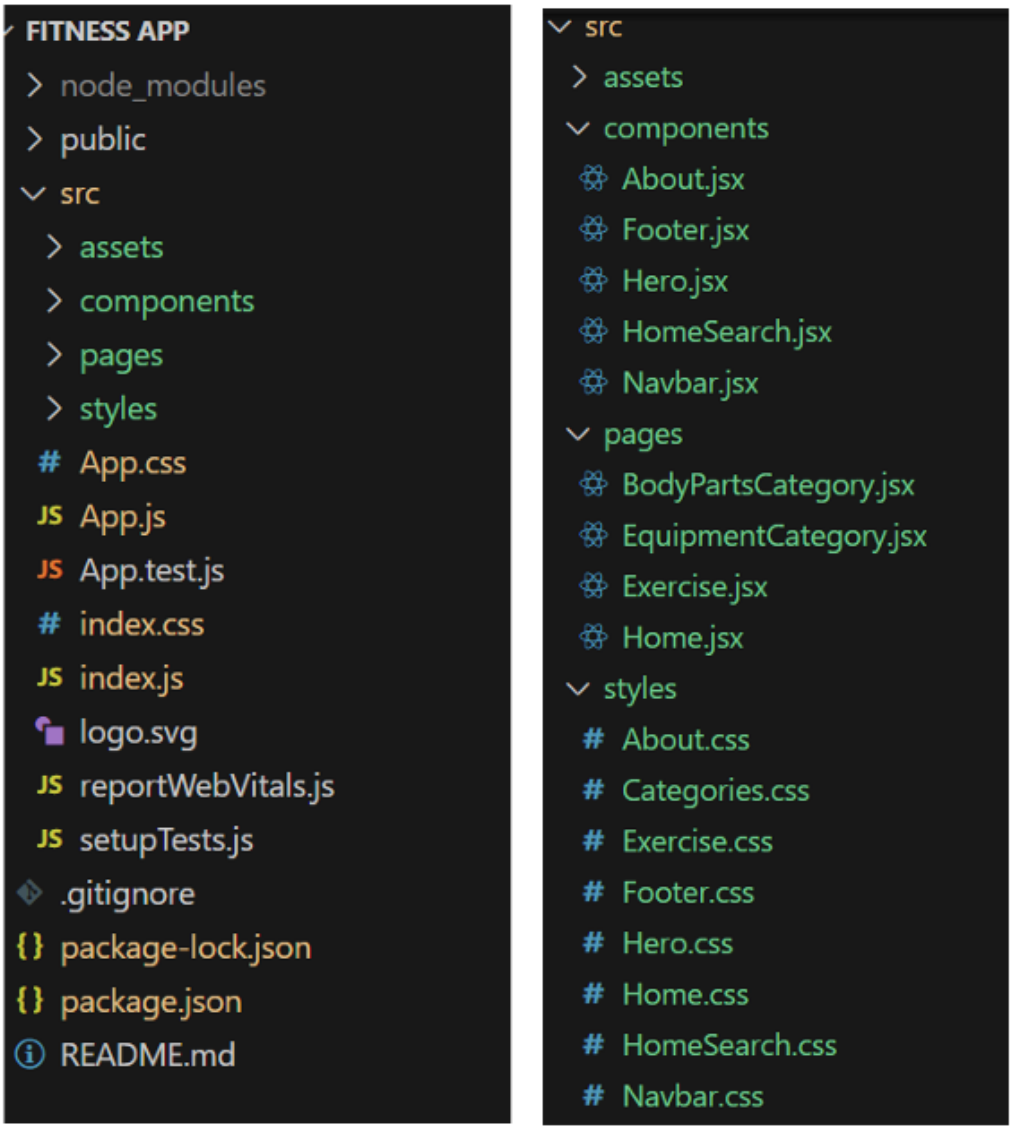
```
npm start
```

Access the App:

- Open your web browser and navigate to http://localhost:3000.
- You should see the application's homepage, indicating that the installation and setup were successful.

You have successfully installed and set up the application on your local machine. You can now proceed with further customization, development, and testing as needed.

Project structure:



In this project, we've split the files into 3 major folders, Components, Pages and Styles. In the pages folder, we store the files that acts as pages at different URLs in the application. The components folder stores all the files, that returns the small components in the application. All the styling css files will be stored in the styles folder.

Project Flow:

Project demo:

Before starting to work on this project, let's see the demo.

Demo link:

https://drive.google.com/file/d/1dVVEwbZgAltQyv8yXszbQkw98dhnOb9V/view?usp=sharing

Use the code in:

https://drive.google.com/drive/folders/14f9eBQ5W7VrLdPhP2W6PzOU_HCy8UMex?usp=sharing

Milestone 1: Project setup and configuration.

- Installation of required tools:

1. Open the project folder to install necessary tools

In this project, we use:

- React Js
- React Router Dom
- React Icons
- Bootstrap/tailwind css
- Axios

- For further reference, use the following resources
- https://react.dev/learn/installation
- https://react-bootstrap-v4.netlify.app/getting-started/introduction/
- https://axios-http.com/docs/intro
- https://reactrouter.com/en/main/start/tutorial

Milestone 2: Project Development

- Setup the Routing paths

  Setup the clear routing paths to access various files in the application.

  Ex:

```
<div className="App">

  <Navbar />

  <Routes>

    <Route path="/" element={<Home />} />
    <Route path="/bodyPart/:id" element={<BodyPartsCategory />} />
    <Route path="/equipment/:id" element={<EquipmentCategory />} />
    <Route path="/exercise/:id" element={<Exercise />} />
  </Routes>

  <Footer />

</div>
```

a. Develop the Navbar and Hero components

b. Code the popular search/categories components and fetch the categories from rapid Api.

c. Additionally, we can add the component to subscribe for the newsletter and the footer.

d. Now, develop the category page to display various exercises under the category.

e. Finally, code the exercise page, where the instructions, other details along with related videos from the YouTube will be displayed.

Important Code snips:

- **Fetching available Equipment list & Body parts list**

```
const bodyPartsOptions = {
  method: 'GET',
  url: 'https://exercisedb.p.rapidapi.com/exercises/bodyPartList',
  headers: {
    'X-RapidAPI-Key': 'place your api key',
    'X-RapidAPI-Host': 'exercisedb.p.rapidapi.com'
  }
};

const equipmentOptions = {
  method: 'GET',
  url: 'https://exercisedb.p.rapidapi.com/exercises/equipmentList',
  headers: {
    'X-RapidAPI-Key': 'place your api key',
    'X-RapidAPI-Host': 'exercisedb.p.rapidapi.com'
  }
};

useEffect(() => {
  fetchData();
}, [])

const fetchData = async () =>{
  try {
    const bodyPartsData = await axios.request(bodyPartsOptions);
    setBodyParts(bodyPartsData.data);

    const equipmentData = await axios.request(equipmentOptions);
    setEquipment(equipmentData.data);
  } catch (error) {
    console.error(error);
  }
}
```

- **Fetching exercises under particular category**

```
const fetchData = async (id) => {
    const options = {
        method: 'GET',
        url: `https://exercisedb.p.rapidapi.com/exercises/equipment/${id}`,
        params: {limit: '50'},
        headers: {
          'X-RapidAPI-Key': 'your api key',
          'X-RapidAPI-Host': 'exercisedb.p.rapidapi.com'
        }
    };

    try {
        const response = await axios.request(options);
        console.log(response.data);
        setExercises(response.data);
    } catch (error) {
        console.error(error);
    }

}
```

- **Fetching Exercise details**

```
useEffect(()=>{
    if (id){
        fetchData(id)
    }
},[])

const fetchData = async (id) => {
    const options = {
      method: 'GET',
      url: `https://exercisedb.p.rapidapi.com/exercises/exercise/${id}`,
      headers: {
        'X-RapidAPI-Key': 'ae40549393msh0c35372c617b281p103ddcjsn0f4a9ee43ff0',
        'X-RapidAPI-Host': 'exercisedb.p.rapidapi.com'
      }
    };

    try {
        const response = await axios.request(options);
        console.log(response.data);
        setExercise(response.data);

        fetchRelatedVideos(response.data.name)
    } catch (error) {
        console.error(error);
    }

}
```

- **Fetching related videos from YouTube**

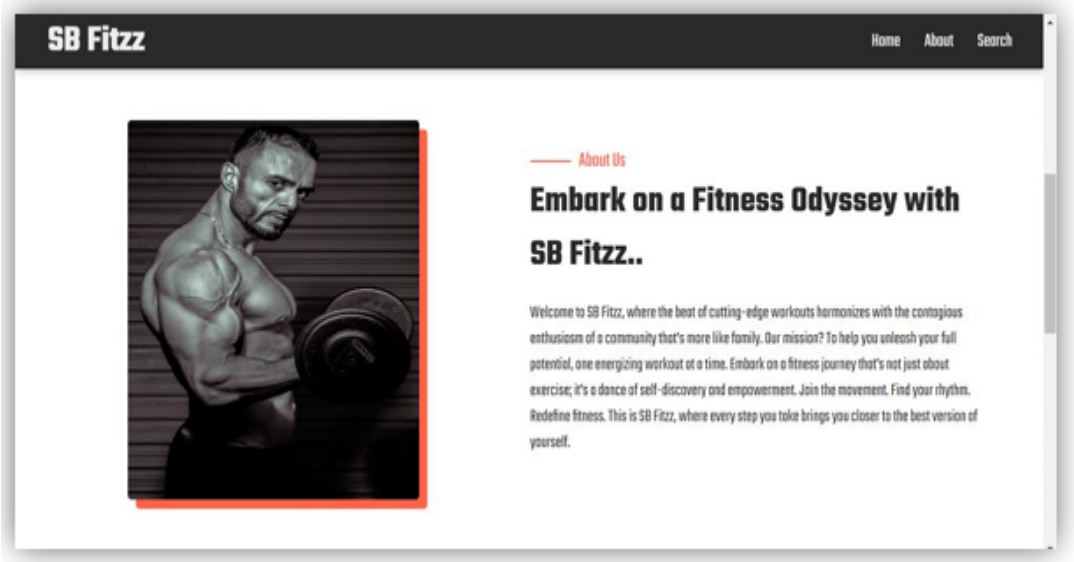
```
const fetchRelatedVideos = async (name)=>{
  console.log(name)
  const options = {
    method: 'GET',
    url: 'https://youtube-search-and-download.p.rapidapi.com/search',
    params: {
      query: `${name}`,
      hl: 'en',
      upload_date: 't',
      duration: 'l',
      type: 'v',
      sort: 'r'
    },
    headers: {
      'X-RapidAPI-Key': 'ae40549393msh0c35372c617b281p103ddcjsn0f4a9ee43ff0',
      'X-RapidAPI-Host': 'youtube-search-and-download.p.rapidapi.com'
    }
  };

  try {
    const response = await axios.request(options);
    console.log(response.data.contents);
    setRelatedVideos(response.data.contents);
  } catch (error) {
    console.error(error);
  }
}
```

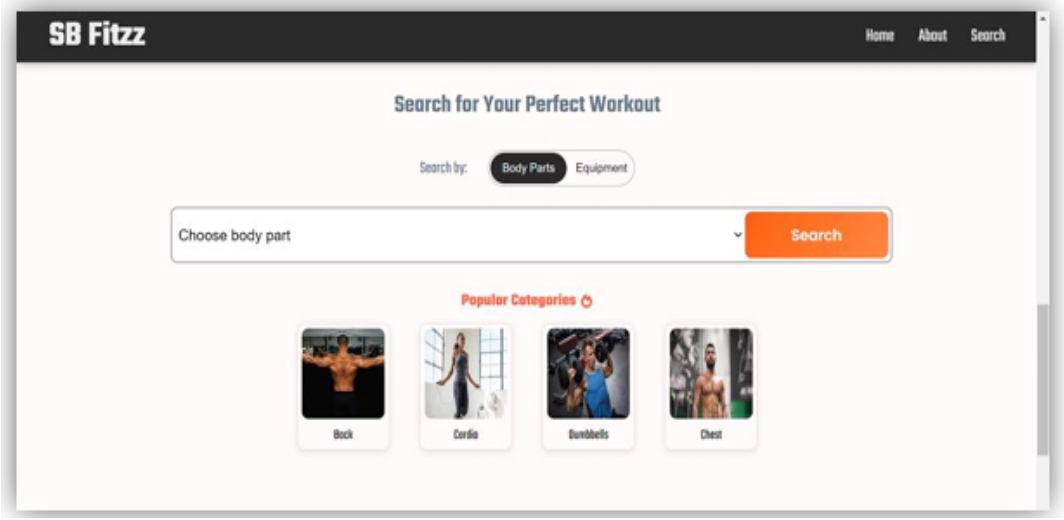
User Interface snips:

- **Hero component**

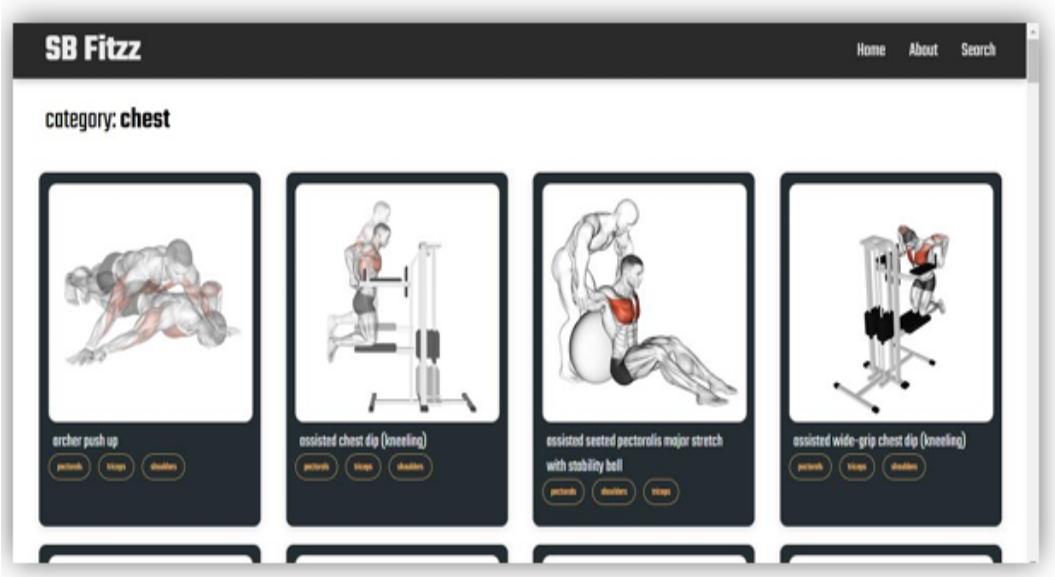
- **About**



- **Search**



- **Category page**



- **Exercise page**