FRONTED DEVELOPMENT WITH REACT.js

FITFLEX

1.INTRODUCTION

*Project title:fitflix

*Team members

Aafila parveen: code executor

Bhuvaneshwari: document creator

Asma khatoon :Demo video

Mary Jeniliya: GitHub creator

2.PROJECT OVERVIEW

*Purpose: fitflix is a web-based fitness platform that combines video streaming with fitness tracking, helping users track workouts, follow routines, and maintain a healthy lifestyle. Key features include:

- Video streaming for workouts
- Personalized fitness dashboards

- Workout and diet tracking
- User authentication and profiles
- Responsive design for mobile and desktop

. *features:

Video streaming for workouts

Personalized fitness dashboards

- Workout and diet tracking
- User authentication and profiles
- Responsive design for mobile and desktop

3.ARCHITECTURE:

**Component Structure:*

- App.js (main entry point)
- Navbar (navigation)
- WorkoutDashboard (workouts)
- VideoPlayer (video streaming)
- Profile (user details)
- Footer (copyright/info)

State Management:

- Global State: React Context API (user auth, workout progress)

- Local State: useState and useReducer (video controls, filters)

Routing:

- Implemented using React Router:

  - /dashboard

  - /videos

  - /profile

  - /login

4.SETUP INSTRUCTION

Prerequisites:

- Node.js (>= 16.x)

- npm or yarn

- Git

Installation:

1. Clone repo: `git clone https://github.com/username/fitflix.git`

2. `cd fitflix`

3. `npm install`

4. `npm start`

5.FOLDER STRUCTURE

Client:

- Organized into components, pages, context, and utils

* Utilities: API calls, authentication helpers, video streaming logic

6.RUNNING THE APPLICATION

.*Project Commands:*

1. Clone: `git clone https://github.com/username/fitflix.git`

2. Install dependencies: `npm install`

3. Start development server: `npm start

4. Build for production: `npm run build

Frontend:

- Built with React
- Uses React Router for navigation
- Utilizes React Context API for state management
- Responsive design for various screen sizes

7.COMPONENT DOCUMENTATION

Key Components:

1. Navbar
2. WorkoutDashboard
3. VideoPlayer
4. Profile

Reusable Components:

1. Button
2. Card
3. InputField
4. VideoCard

8.STATE MANAGEMENT

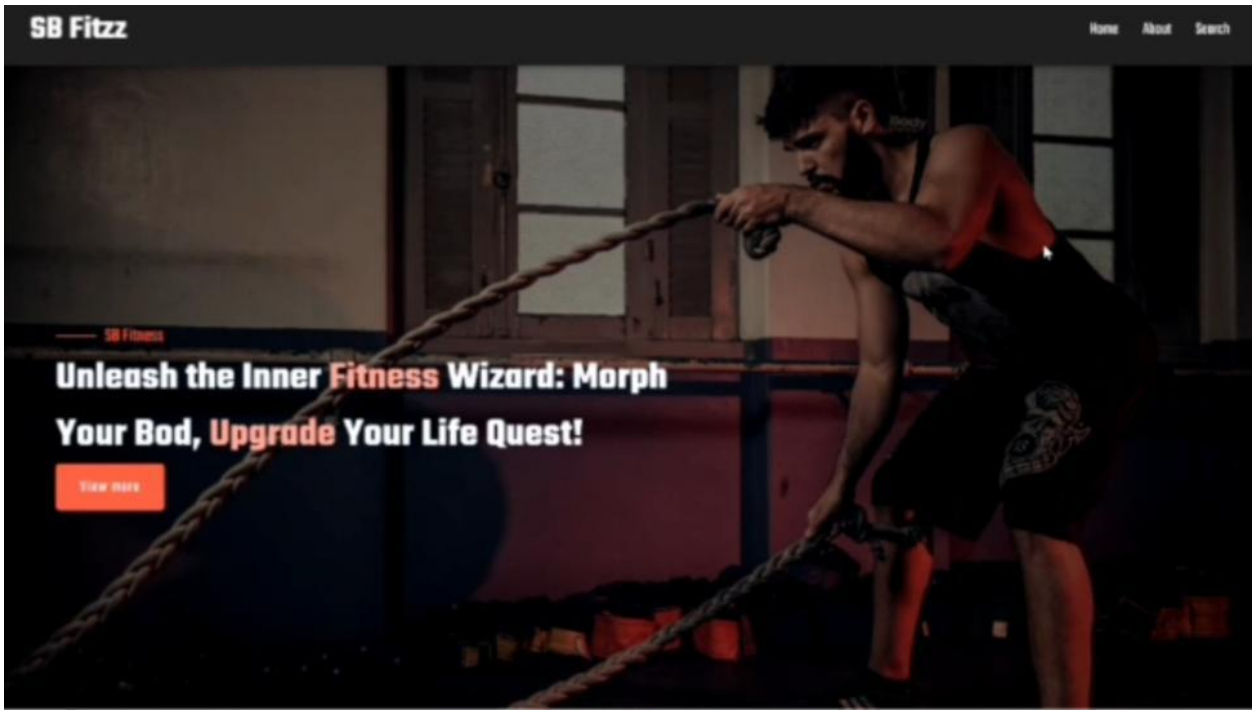
Global State:

- Managed with React Context API
- Stores user auth data and workout progress

Local State:

- Managed with `useState` and `useReducer`
- Handles component-specific data, such as:
  - Video controls
  - Filter settings
  - Form inputs

9.USER INTERFERENCE



10.STYLING

CSS Framework:

- Bootstrap
- or possibly Material-UI / Tailwind CSS

Theming:

- Custom color scheme (primary, secondary, accent)
- Typography (font family, sizes)
- Consistent spacing and layout

11.TESTING

Testing Strategy:

- Unit testing with Jest
- Component testing with React Testing Library
- Integration testing for API interactions

Code Coverage:

- Aim for 80-90% coverage
- Tracked with tools like Jest coverage reports
- Focus on critical components and logic

12.SCREENSHOT OR DEMO

https://drive.google.com/file/d/1C2Y9i7PnCWmvItM4oTHmlOALTSnPjw_J/view?usp=drivesdk

13.KNOWN ISSUE

1. Windows Autopilot: TPM attestation failures, kiosk device profile issues

2. Microsoft Edge: Sync functionality, PDF rendering, extension issues

3. Azure SQL Managed Instance: Backup retention, login, service principal issues

4. FSLogix: Service crashes, LocalCache/TempState issues

5. Business Central: Installation, upgrade issues

6. Customer Insights: Email editor limitations, form issues

14.FUTURE ENHANCEMENT

1. AI-powered workout recommendations

2. Social sharing for progress tracking
3. Personalized nutrition planning
4. Interactive video content
5. Expanded wearable integrations
6. Mobile app development
7. Gamification elements
8. Advanced analytics for user insights