Frontend Development with React.js Project Documentation

# Introduction

**Project Title :** FitFlex your personal fitness companion

**TEAM ID : NM2025TMID42081**

**TEAM LEADER : POOJA M**

**ROLE :** CODING AND DEVELOPMENT

**TEAM MEMBER : PRIYADHARSHINI A**

**ROLE :** UPLOADING

**TEAM MEMBER : RAGAVI S**

**ROLE :** DEMO VIDEO

**TEAM MEMBER : RAJESRI S**

**ROLE :** DOCUMENT CREATER

# Project Overview

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

**Features: Exercises from Fitness API, Visual Exercise Exploration, Intuitive and User-Friendly Design, Advanced Search Feature.**

# Architecture

**Component Structure**: UI components like the onboarding screens, login screen, Dashboard, and workout details.

**State Management**: Local component state, context API, External state management libraries, Backend state.

**Routing**: "Routing" can refer to different concepts, but for the FitFlex health and fitness app, it likely relates to internal navigation within the app or external routing of user data.

library like React Router DOM to manage different views or pages of your web application.

# Setup Instructions

**Prerequisites**: FitFlex isn't just another fitness app. We're meticulously designed to transform your workout experience, no matter your fitness background or goals.

**Installation**: Provide a step-by-step guide to Clone the repository from GitHub. Navigate to the client directory.

Run npm install to install dependencies.

Configure environment variables for any API keys

# Folder Structure

**Client**: The main React application is organized into folders such as components, pages, and assets for images and audio files.

**Utilities**: Helper functions for music playback and custom hooks for managing audio state are located in the utils folder

# Code:

project-name/

├── public/

│ ├── index.html

│ └── favicon.ico

├── src/

│ ├── assets/

│ │ ├── images/

│ │ └── styles/

│ ├── components/

│ │ ├── Header.js

│ │ ├── Footer.js

│ │ └── ...

│ ├── containers/

│ │ ├── App.js

│ │ └── ...

│ ├── actions/

│ │ ├── index.js

│ │ └── ...

│ ├── reducers/

│ │ ├── index.js

│ │ └── ...

│ ├── utils/

│ │ ├── api.js

│ │ └── ...

│ ├── index.js

│ └── setupTests.js

├── tests/

│ ├── components/

│ │ ├── Header.test.js

│ │ └── ...

│ └── ...

├── .git.ignore

├── package. json

└── README.md

# Running the Application

Provide commands to start the frontend server locally.

**Frontend**: npm start in the client directory.

# Component Documentation

**Key Components**: Manages music playback controls. It receives a song prop.

Playlist Component: Displays the list of songs and manages playlist creation.

Search Component: Handles searching for songs

**Reusable Components**: Button Component: A configurable button component used throughout the application.

Song Card Component: A component for displaying individual songs in lists.

# State Management

**Global State**: The Context API is used to manage the global state, ensuring that the current song and playback status are accessible to all components..

**Local State**: se use State hooks to manage their own local state, such as form inputs or UI toggles

# User Interface

Provide screenshots or GIFs showcasing different UI features, such as pages, forms, or interactions.

# Styling

**CSS Frameworks/Libraries**: Describe any CSS frameworks, libraries, or pre- processors (e.g., Sass, Styled-Components) used.

**Theming**: A custom design system with light and dark themes is implemented

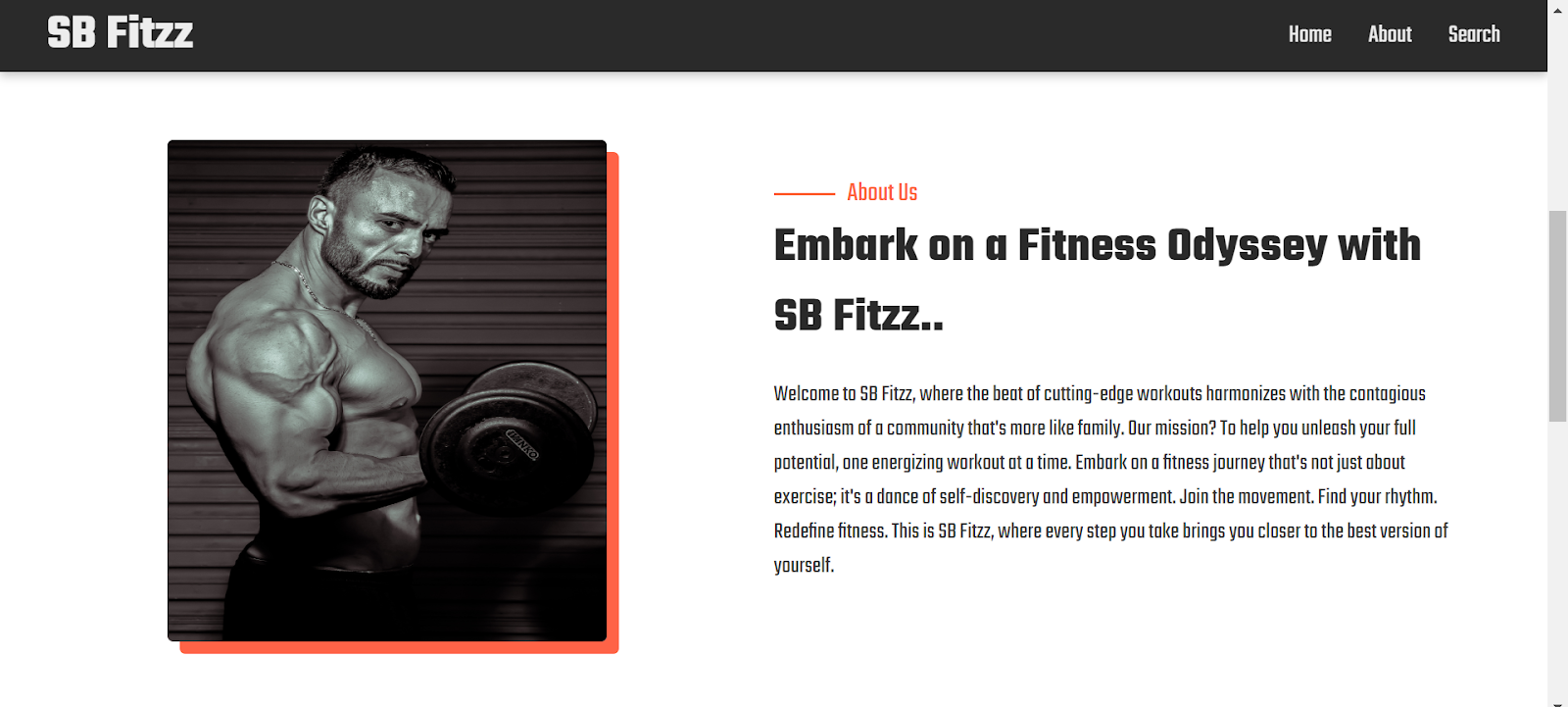
# Testing

**Testing Strategy**: The application uses Jest and React Testing Library for unit and integration testing of components.

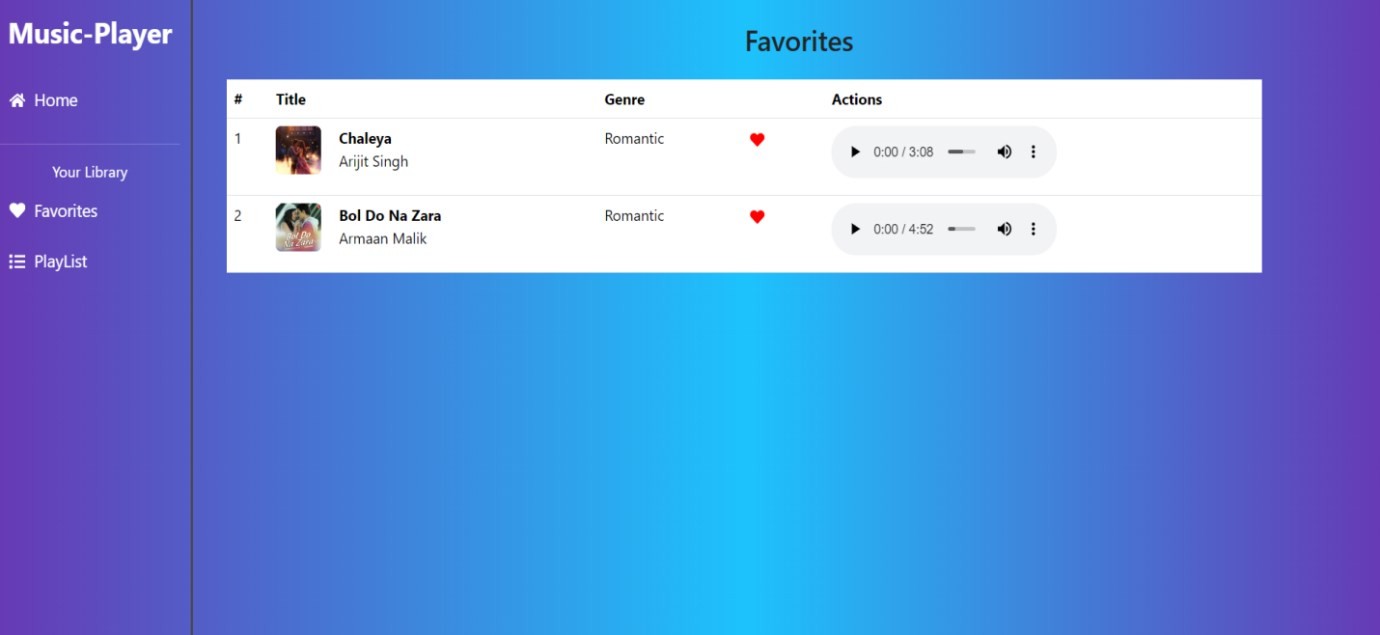
**Code Coverage**: Code coverage is tracked using a tool to ensure all key components are adequately tested.

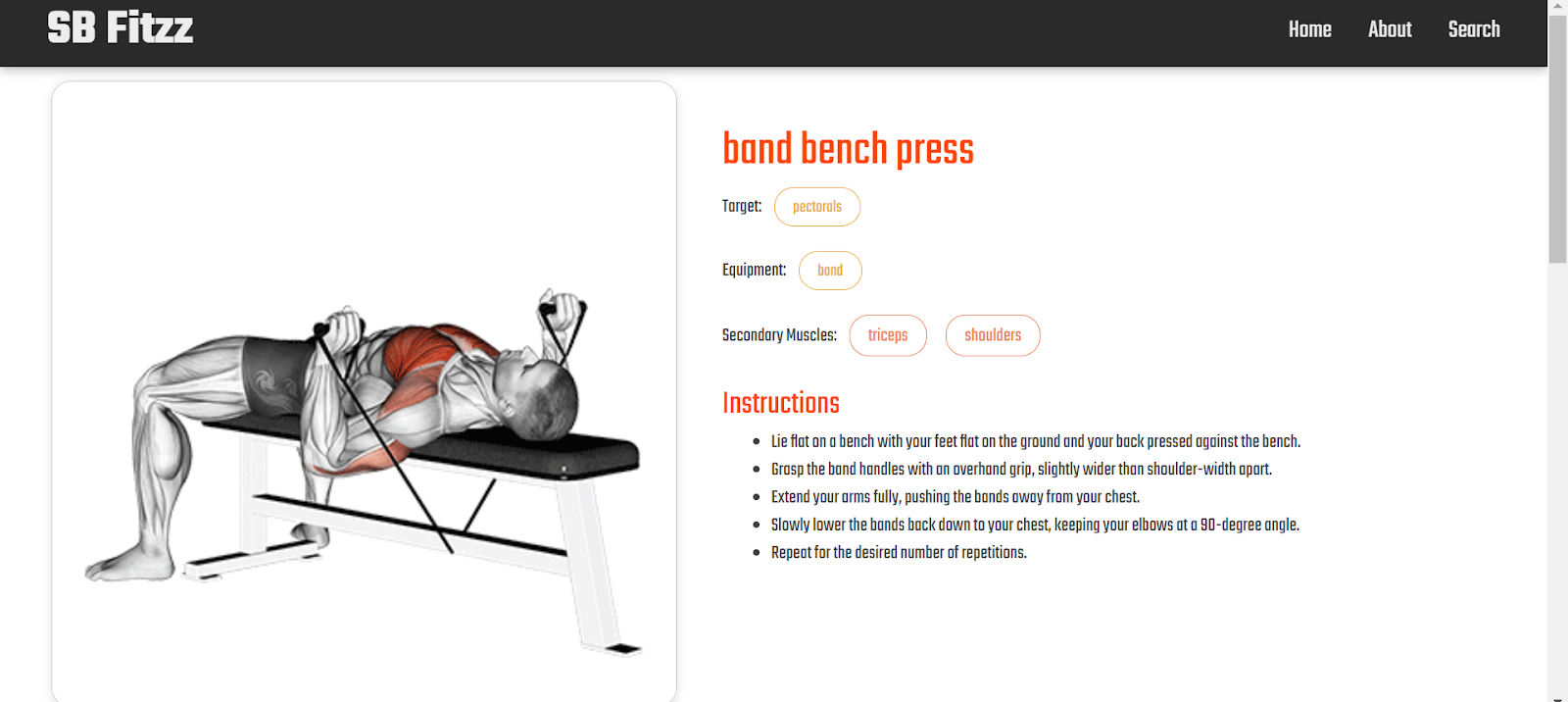
**Screenshots or Demo:**





**Favorites:**

****



**Known Issues**

Any known bugs or issues, such as occasional playback glitches on certain browsers, should be documented here

# Future Enhancements

Potential future features could include a user authentication system, enhanced search functionality, and adding more animations to the user interface.

# DEMOLINK

# https://drive.google.com/file/d/1xbzVoqxeJRnvsh0FWv5D16EOANfwZ8U8/view?usp=sharing