🎵 Frontend Development Project Documentation

1. Introduction

- **Project Title**: Music Player App
- **Team Members**:

  Sriram S,Vasanth M,Manimaran S

2. Project Overview

- **Purpose**:
  A responsive and interactive music player built with React.js that allows users to browse songs, play audio, manage favorites, and create custom playlists.
- **Features**:
  - Audio playback with controls
  - Search functionality by song name, genre, or artist
  - Add to Favorites & Playlist
  - Navigation through Home, Library, Favorites, and Playlist sections
  - Responsive and modern UI with album artwork

3. Architecture

- **Component Structure**:
    - `App.js`: Root component handling routing and layout
    - `Sidebar.js`: Left navigation menu (Home, Favorites, Playlist)
    - `SearchBar.js`: Search input field
    - `SongCard.js`: Displays individual song details and audio player
    - `Playlist.js`, `Favorites.js`, `Library.js`: Pages for different views
- **State Management**:
    - **Context API** used to manage global state for:
        - Favorites
        - Playlist
        - Search Query
- **Routing**:
    - `react-router-dom` for page navigation:
        - `/` → Home
        - `/favorites` → Favorites Page
        - `/playlist` → Playlist Page

---

4. Setup Instructions

- **Prerequisites**:
    - Node.js >= 14.x

- npm or yarn

Installation:

```
 git clone https://github.com/your-username/music-player-app.git
cd music-player-app
npm install
```

-

---

5. Folder Structure

```
music-player-app/
├── public/
├── src/
│   ├── assets/         # Images, logos
│   ├── components/     # Reusable components (SongCard, Sidebar, etc.)
│   ├── context/        # Context API for global state
│   ├── pages/          # Home, Favorites, Playlist
│   ├── App.js
│   ├── index.js
│   └── styles/         # CSS or Styled-Components
```

- **Utilities**:
  - Custom Hooks for managing audio state
  - Utility functions for filtering/searching songs

---

6. Running the Application

Start Development Server:

```
 npm start
```

-

---

7. Component Documentation

- **`SongCard`**:
    - Props: `title`, `genre`, `singer`, `audioSrc`, `imageSrc`
    - Features: Play/Pause, Add to Playlist/Favorites
- **`Sidebar`**:
    - Navigation component for switching views
- **Reusable Components**:
    - `AudioPlayer`
    - `Button`
    - `SearchBar`

---

8. State Management

- **Global State** (via Context API):
    - `favorites`
    - `playlist`
    - `searchQuery`
- **Local State**:
    - Component-level toggles (e.g., play/pause button)
    - UI states (e.g., hover, active)

---

9. User Interface

Here’s a preview of the UI (as seen in the uploaded image):

📸 Screenshot:

- Left sidebar for navigation
- Search bar for filtering songs
- Responsive card layout showing:
    - Song cover
    - Title, genre, and artist
    - Audio controls
    - "Add to Playlist" button

---

10. Styling

- **CSS Frameworks**: Plain CSS with Flexbox and Grid
- **Theming**:
    - Gradient background (blue to purple)
    - Consistent card design
    - Responsive layout for all screen sizes

---

11. Testing

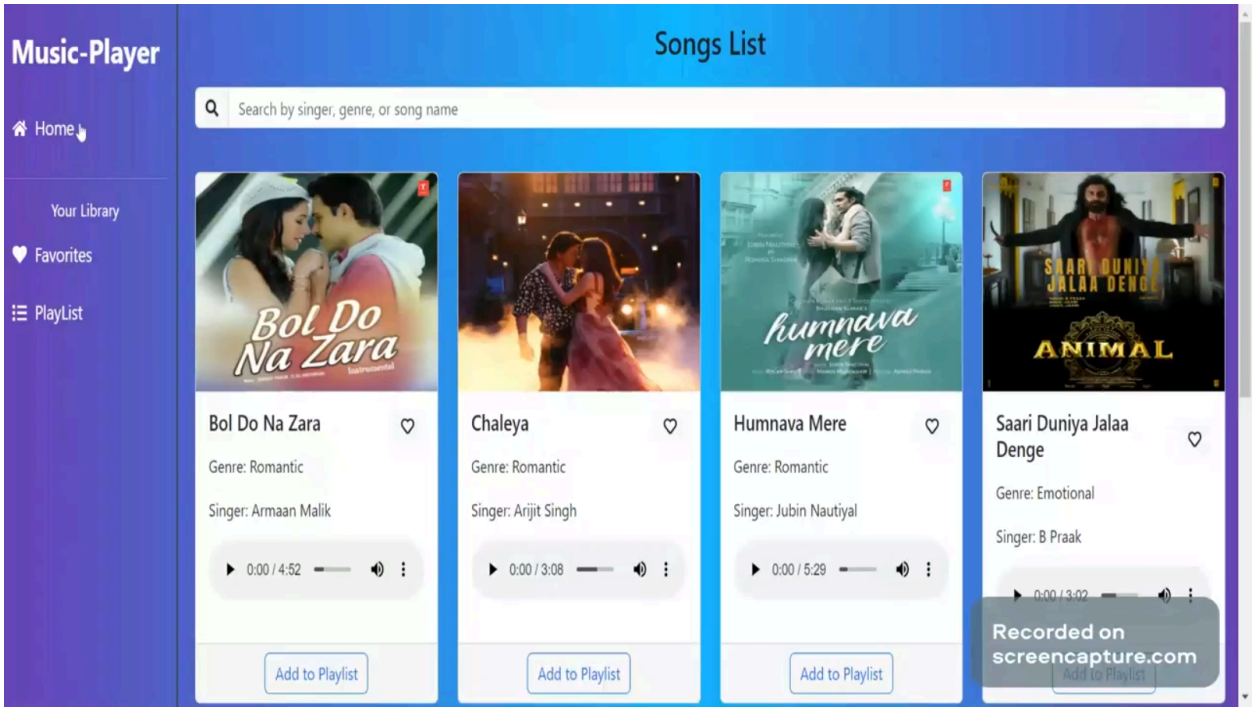
- **Testing Strategy**:

- Unit tests for components using `Jest`
- Integration tests for Playlist and Favorites logic using `React Testing Library`

- **Code Coverage**:
  - Minimum 80% coverage goal
  - `coverage/` reports auto-generated

---

12. Screenshots or Demo



---

13. Known Issues

- Song duration does not update dynamically when scrubbing audio
- No user authentication (favorites/playlist not saved after refresh)

---

14. Future Enhancements

- User authentication with Firebase/Auth0
- Persistent state with localStorage or backend
- Add song upload feature
- Theme toggle (Dark/Light mode)
- Song categories & filters
