Frontend Development withReact.js

Project Documentation for Rhythmic Tunes

---

1. Introduction

- **Project Title**: Rhythmic Tunes
- **Team Members**:
  - **VARSHA P(Team Leader)** [EmailId:varshavarshuu523@gmail.com]
  - **Naveena A** [EmailId: naveenaanbu32@gmail.com]
  - **Mageshwari R** [EmailId:mageshwari248@gmail.com]
  - **Malini S** [EmailId:smalinismalini647@gmail.com]

---

2. ProjectOverview

- **Purpose**:
  RhythmicTunesisawebapplicationdesignedtoprovideuserswithaseamlessmusic listening experience. The application allows users tobrowse, search, and play music tracks,createplaylists,anddiscovernewmusicbasedontheirpreferences.
- **Features**:
  - Musicplayerwithplay,pause,skip,andvolumecontrol.
  - Searchfunctionalitytofindsongs,albums,andartists.
  - Userauthentication(login/signup).
  - Playlistcreationandmanagement.
  - Responsivedesignformobileanddesktop.

---

3. Architecture

- **ComponentStructure**:
  TheapplicationisbuiltusingReact.jswithacomponent-basedarchitecture.Major components include:

- **Header**:Containsthenavigationbarandsearchbar.
- **Player**:Musicplayercontrols(play,pause,volume,etc.).
- **Sidebar**:Displaysuserplaylistsandnavigationlinks.
- **HomePage**:Displaysfeaturedtracks,recommendedplaylists,andnew releases.
- **SearchPage**:Allowsuserstosearchforsongs,albums,andartists.
- **PlaylistPage**:Displaysuser-createdplaylistsandallowsplaylistmanagement.

- **StateManagement**: Theapplicationuses**Redux**forglobalstatemanagement.TheReduxstoremanages user authentication, current playing track, playlist data, and search results.
- **Routing**: Theapplicationuses**ReactRouter**fornavigation.Routesinclude:
  - /:Home page
  - /search:Searchpage
  - /playlist/:id:Playlistdetailspage
  - /login:Userloginpage

---

4. Setup Instructions

- **Prerequisites**:
  - Node.js(v16orhigher)
  - npm(v8or higher)
  - Git
- **Installation**:
  1. Clonetherepository:gitclone https://github.com/unm115u22cs138/Rythmic-tunes
  2. Navigatetotheclientdirectory:cdrhythmic-tunes/client
  3. Installdependencies:npminstall
  4. Configureenvironmentvariables:Createa.envfileintheclientdirectoryand add the necessary variables (e.g., API keys).
  5. Startthedevelopmentserver:npmstart

---

5. FolderStructure

- **Client**:
    - **src/components:**#Reusablecomponents(Header,Player,etc.)
    - **src/pages:**#Pagecomponents(HomePage,SearchPage,etc.)
    - **src/assets:**#Images,icons,andotherstaticfiles
    - **src/redux:**#Reduxstore,actions,andreducers
    - **src/utils:**#Utilityfunctionsandhelpers
    - **App.js:**#Mainapplicationcomponent
    - **index.js:**#Entrypoint
- **Utilities**:
    - **api.js**:HandlesAPIrequeststothebackend.
    - **auth.js**:Managesuserauthenticationandtokenstorage.
    - **hooks/usePlayer.js**:Customhookformanagingthemusicplayerstate.

---

6. RunningtheApplication

Frontend:

- Tostartthefrontendserver,runthefollowingcommandintheclientdirectory: npm start
- npminstall
- npxjson-server./db/db.json
- npmrundev
- Theapplicationwillbeavailableathttp://localhost:3000

---

7. ComponentDocumentation

- **KeyComponents**:
    - **Header**:Displaysthenavigationbarandsearchbar.
        - Props:onSearch(functiontohandlesearchqueries).
    - **Player**:Controlsthemusicplayback.
        - Props:currentTrack(objectcontainingtrack details), onPlay, onPause, onSkip.

    - **PlaylistCard**:Displaysaplaylistwithitsnameandcoverimage.
        - Props:playlist(objectcontainingplaylistdetails),onClick(functionto handle playlist selection).

- **ReusableComponents**:
    - **Button**:Acustomizablebuttoncomponent.
        - Props:text,onClick,disabled.
    - **Input**:Areusableinputfieldforformsandsearch.
        - Props:type,placeholder,value,onChange.

---

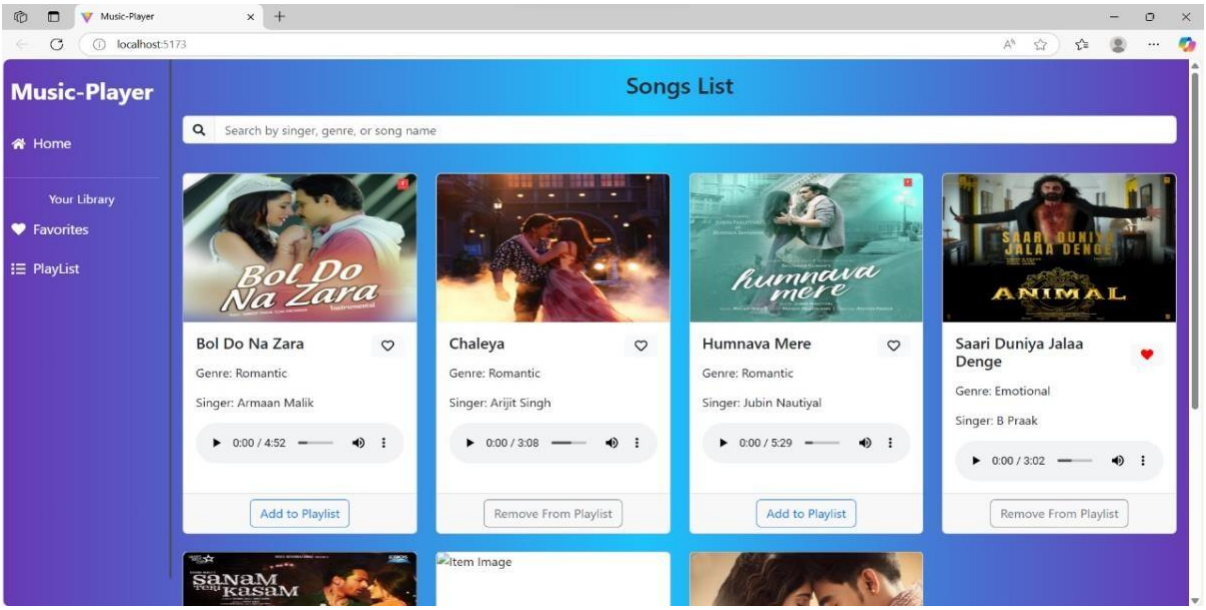
8. StateManagement

- **GlobalState**:
  TheReduxstoremanagesthefollowingglobalstates:
    - **user:**Currentauthenticateduser.
    - **player:**Currentplayingtrack,playbackstatus(playing/paused),andvolume.
    - **playlists:**User-createdplaylists.
    - **searchResults:**Resultsfromthesearchfunctionality.
- **Local State**:
  LocalstateismanagedusingReact'suseStatehookwithincomponents.Forexample, the SearchPage component manages the search query input locally.

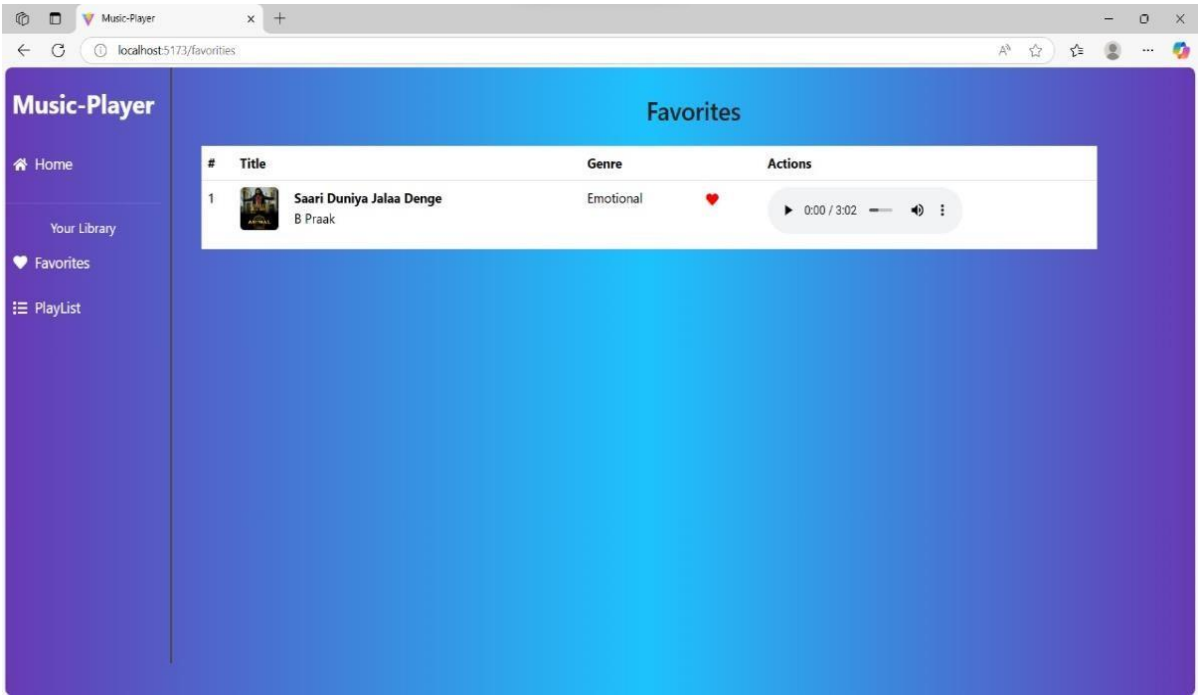
---

9. UserInterface

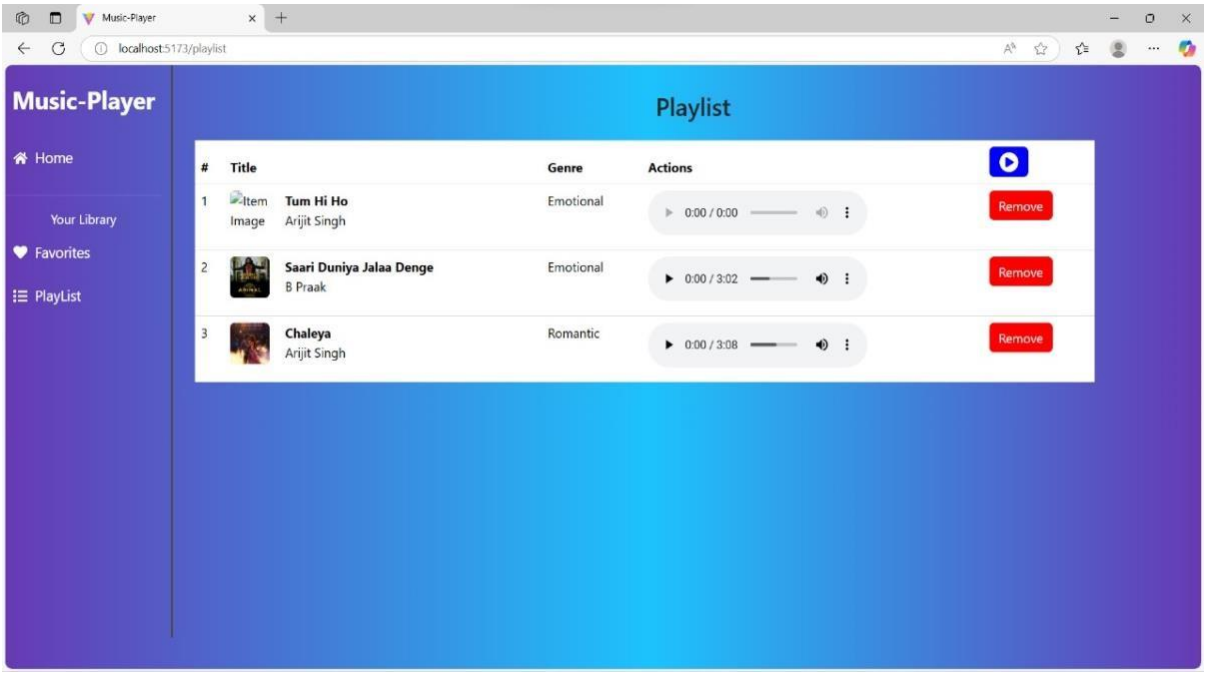
- **Screenshots**
    - **HomePage:**Displayfeaturedtracksandrecommendedplaylists.



- **SearchPage:**Allowsuserstosearchforsongs,albums,andartists.



- **PlaylistPage:**Displaysuser-createdplaylistsandallowsplaylistmanagement.



---

10. Styling

- **CSS Frameworks/Libraries**: Theapplicationuses**Styled-Components**forstyling.Thisallowsformodularand scoped CSS within components.
- **Theming**: AcustomthemeisimplementedusingStyled-Components,withsupportforlightand dark modes.

---

11. Testing

- **TestingStrategy**:
  - **UnitTesting:**Using**Jest**and**ReactTestingLibrary**.
  - **IntegrationTesting**:Isperformedtoensurethatcomponentsworktogether as expected.
  - **End-to-EndTesting:Cypress**isusedforend-to-endtestingofuserflows.
- **Code Coverage**:
  - CodecoverageismonitoredusingJest'sbuiltincoveragetool.Thecurrent coverage is 85%.

---

12. ScreenshotsorDemo

- **Demo Link: https://github.com/unm115u22cs138/Rythmic-tunes/commit/9ea9ad641eaca25607e3bc3ab5c0444a9ba9024b**

Screenshots:Seesection9forUI screenshots.

13. Known Issues

- **Issue1**:Themusicplayersometimesskipstracksunexpectedly.
- **Issue2**:Thesearchfunctionalityisslowwithlargedatasets.

---

14. Future Enhancements

- **Future Features**:
  - Addsupportforuserprofilesandsocialsharing.
  - Implementarecommendationengineforpersonalizedmusicsuggestions.
  - Addanimationsandtransitionsforasmootheruserexperience.

Thisdocumentationprovidesacomprehensiveoverviewofthe**RhythmicTunes**project, including its architecture, setup instructions, and future plans.