**RHYTHMIC TUNES: YOUR MELODIC**

**COMPANION – PROJECT**

**DOCUMENTATION**

1. **INTRODUCTION**

**Project Title:**

Rhythmic Tunes : Your Melodic Companion

**Team Leader:**

Aswinth JJ

**Team Members:**

1. Balaji M
2. Abdhul Ibrahim A
3. **PROJECT OVERVIEW**

**Purpose:**

* Rhythmic Tunes is a music-based web application designed to provide users with a seamless platform to discover, play, and manage music.
* The goal is to enhance the listening experience with modern UI, personalized playlists, and smooth performance.

**Features:**

* + - Music player with play/pause, next/previous, and volume control.
    - Playlist creation and management.
    - Search functionality for tracks and artists.
    - Responsive design for mobile and desktop.
    - User authentication (login/signup).

1. **Architecture**

**Component Structure:**

* + Navbar – Navigation across pages
  + MusicPlayer – Core player UI and controls
  + Playlist – Displays user’s playlist
  + SearchBar – Enables music search
  + AuthForm – Handles login/signup

**State Management:**

* + Global state handled using **Context API / Redux**.
  + Local state for UI interactions (like play/pause).

**Routing:**

* + / → Home
  + /search → Search Page
  + /playlist → User Playlists
  + /login → Login/Signup

1. **Setup Instructions**

**Prerequisites:**

Node.js, npm, Git

**Installation:**

* + Clone repo: git clone <repo-link>
  + Navigate: cd rhythmic-tunes
  + Install dependencies: npm install
  + Configure environment variables (API keys if using music APIs)

1. **Folder Structure**

**Client:**

* + src/components – Navbar, Player, Playlist, etc.
  + src/pages – Home, Search, Login
  + src/assets – Icons, images, audio files

**Utilities:**

Custom hooks for audio handling, helper functions for playlist logi

1. **Running the Application**

Start frontend locally:

npm start

1. **Component Documentation**

**Key Components:**

* + MusicPlayer: Controls playback
  + Playlist: Displays tracks in a list
  + SearchBar: Provides search input

**Reusable Components:**

* + Button, Card, Modal, Input

1. **State Management**

**Global State:**

* Stores user authentication, playlists, current song info

**Local State:**

* Controls UI (play/pause toggle, modal open/close)

**9. User Interface (UI)**

Screenshots to include:

* + - * Home page
      * Music player
      * Playlist view
      * Search results

**10. Styling**

**CSS Frameworks/Libraries:**

* Tailwind CSS + Styled Components

**Theming:**

* Dark mode and light mode options

**11. Testing**

**Testing Strategy:**

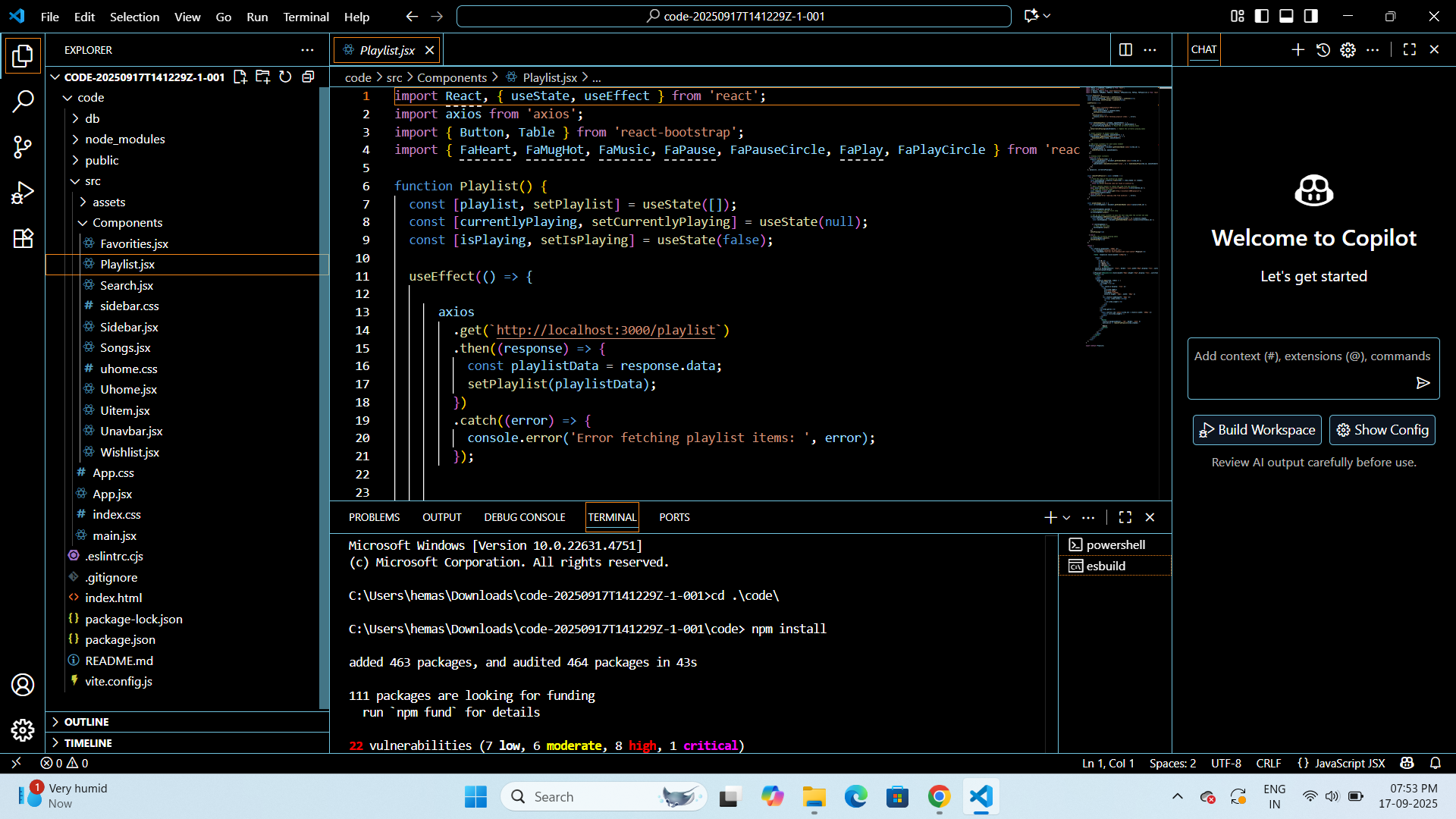
* Unit tests for components, integration for player + search

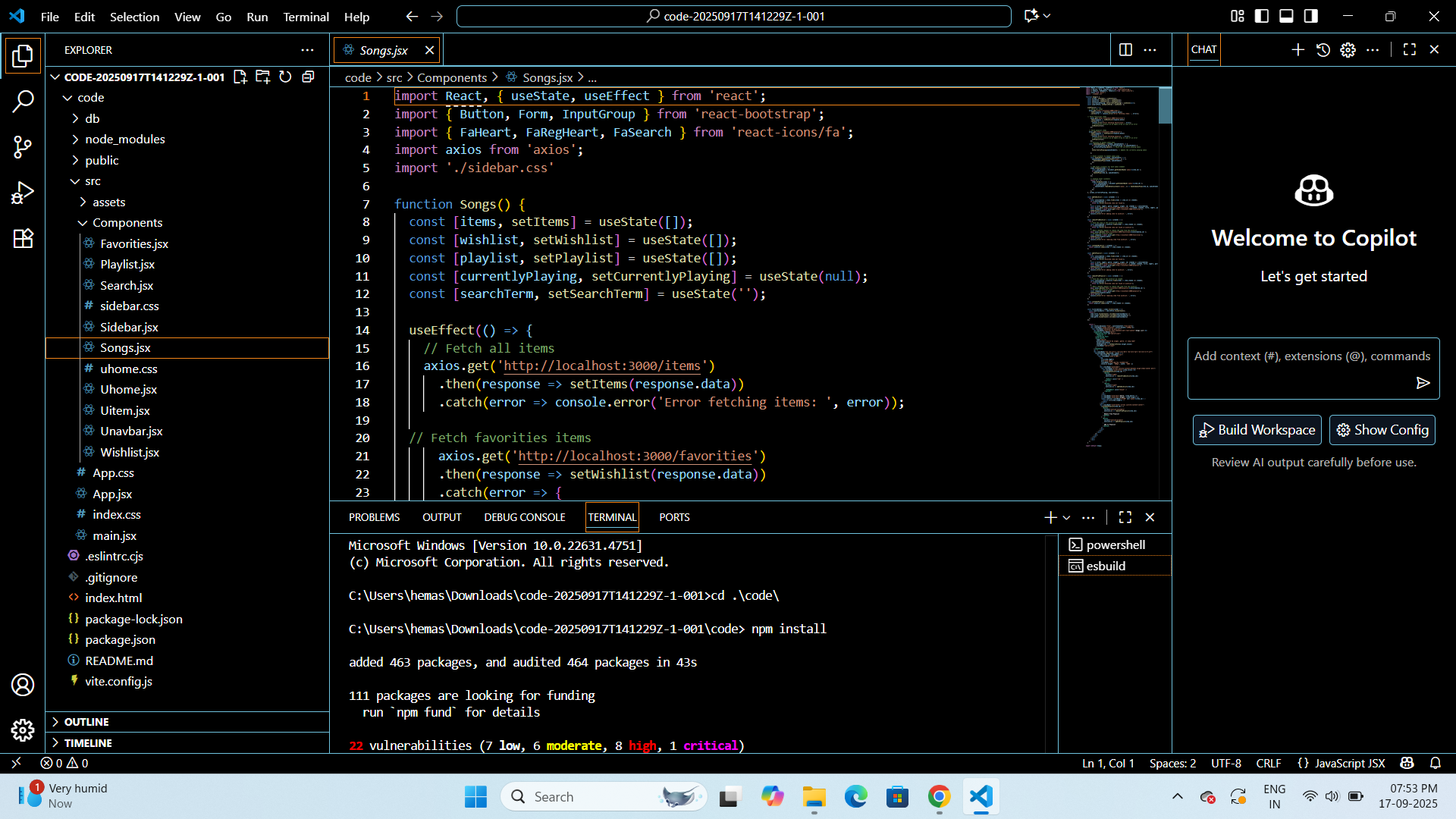
**Code Coverage:**

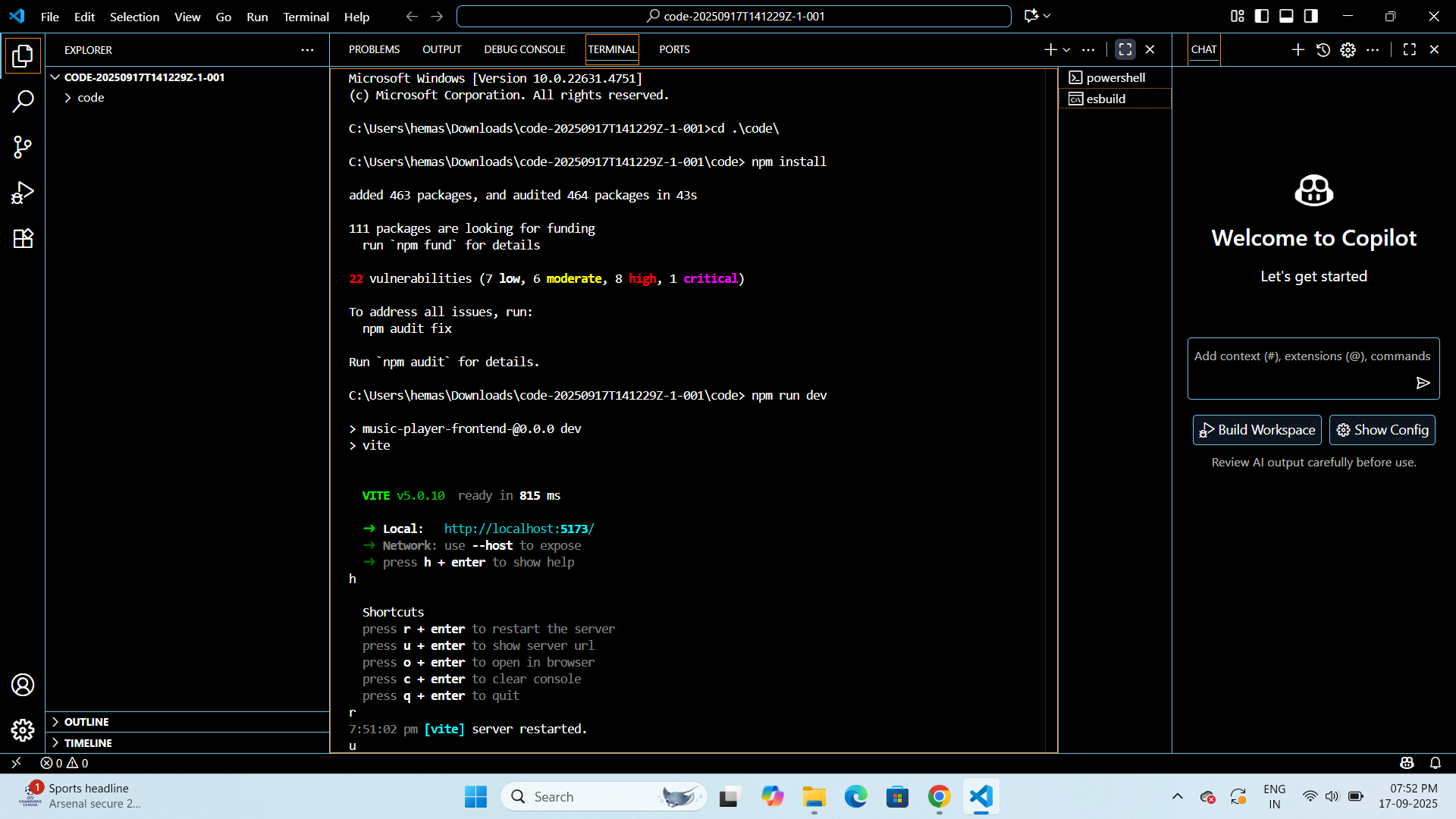
* Jest + React Testing Library

**12. Screenshots or Demo**

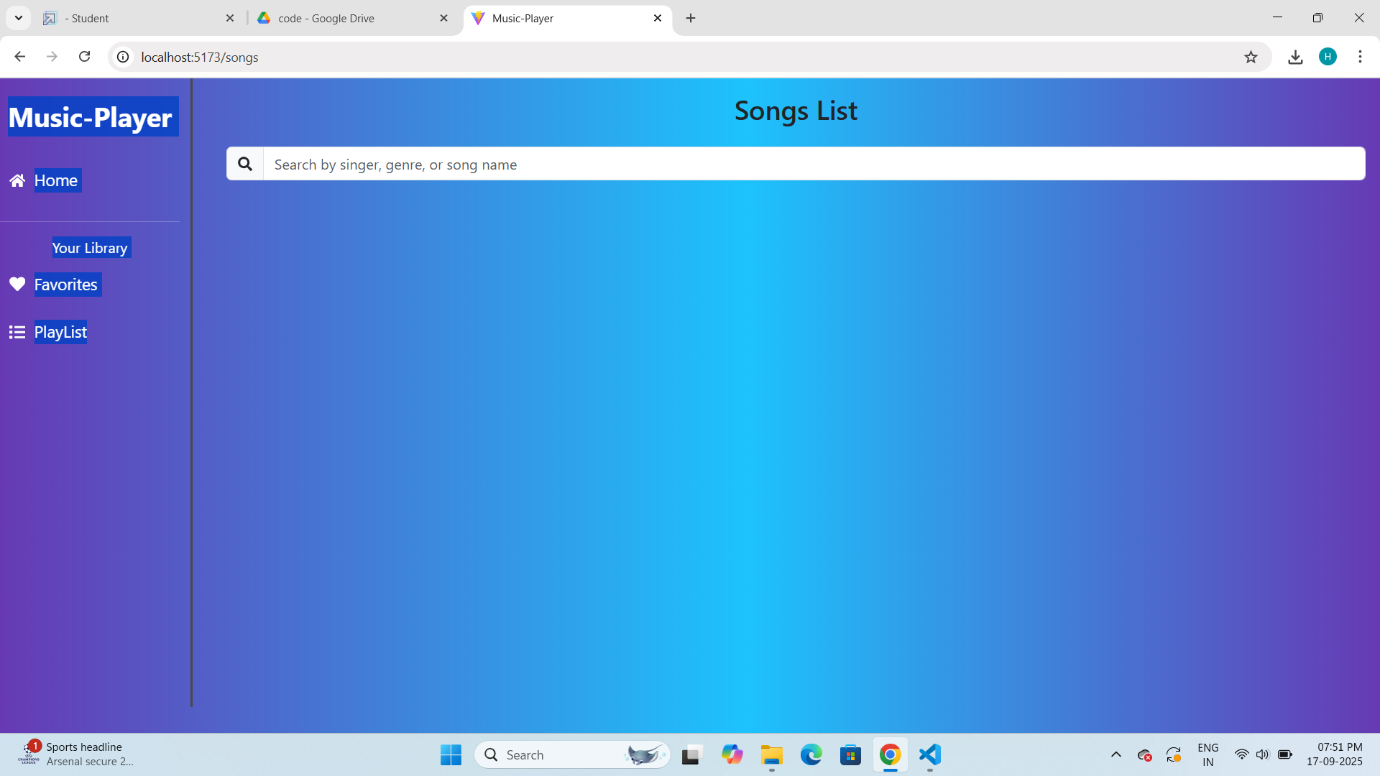
Provide screenshots of music player, playlists, and search page







**OUTPUT**



**13. Known Issues**

* + Playback might lag on slow networks
  + Some APIs may require premium access

**14. Future Enhancements**

* + - AI-powered song recommendations
    - Offline playback
    - Lyrics display with sync
    - Social sharing of playlists