RHYTHMIC TUNES

DOCUMENTATION

INTRODUCTION:-

Project Title:-

Rhythmic Tunes.

* Team ID:-
* NM2025TMID38950
* Team Leader:-

|  |  |
| --- | --- |
| Anok K – | [Anokanok51@gmail.com](mailto:Anokanok51@gmail.com) |

* Team Members:-

|  |  |  |
| --- | --- | --- |
|  | Kamaresh.A | [Kamaresharumugam@gmail.com](mailto:Kamaresharumugam@gmail.com) |
|  | Dilli babu.R | [Dilli0052@gmail.com](mailto:Dilli0052@gmail.com) |
|  | Jaya prakash.N | [S90256965@gmail.com](mailto:S90256965@gmail.com) |
|  | Mohammed Ibrahim.A | [Mohammedmohammedz870@gmail.com](mailto:Mohammedmohammedz870@gmail.com) |

PROJECT OVERVIEW:-

**1.Purpose:-**  
The purpose of *Rhythmic Tunes* is to create a comprehensive and user-friendly platform that enhances the way users interact with music. It is designed to not only provide access to songs and playlists but also integrate advanced features like secure communication, feedback systems, and admin-level management. The project bridges entertainment, interaction, and security into a single music-based application.

**2.Features:-**  
 The major features of *Rhythmic Tunes* include:

* + **Project Posting and Binding** – Enables users to share music-related content, projects, or playlists with the community and bind them with relevant categories for better accessibility.
  + **Secure Chat System** – A built-in chat module that ensures private, encrypted communication between users for collaboration and interaction.
  + **Feedback and Review System** – Allows users to provide feedback on songs, playlists, or shared content, helping improve recommendations and community engagement.
  + **Admin Control Panel** – A dedicated dashboard for administrators to manage users, content, and system activities effectively, ensuring smooth operation and security.

**3.Architecture:-**

1. **Frontend – React.js with Bootstrap and Material UI**  
   The frontend of *Rhythmic Tunes* is built using **React.js**, ensuring a highly dynamic and responsive user interface. **Bootstrap** provides a robust grid system and layout framework, while **Material UI** adds modern, pre-designed components for consistency and visual appeal. Together, they deliver a smooth, intuitive, and interactive user experience across devices.
2. **Backend – Node.js and Express.js**  
   The backend is powered by **Node.js** for fast, scalable, and event-driven server operations. **Express.js** is used to handle routing, server logic, and API endpoints, ensuring efficient communication between the frontend and database. This combination supports secure data processing and reliable application performance.
3. **Database – MongoDB**  
   **MongoDB** serves as the primary database, chosen for its flexibility in handling unstructured data. It stores user information, project postings, application details, and chat messages in a secure and scalable manner. Its document-oriented structure makes it well-suited for real-time interactions and large-scale data management.

**4.Setup Instructions:-**

1. **Prerequisites**  
   Before setting up *Rhythmic Tunes*, ensure that the following tools and technologies are installed on your system:
   * **Node.js** – JavaScript runtime environment.
   * **MongoDB** – Database for storing user data, projects, and chat messages.
   * **Git** – For cloning and version control.
   * **React.js** – Frontend framework for building the user interface.
   * **Express.js** – Backend framework for handling APIs and server logic.
   * **Mongoose** – ODM (Object Data Modeling) library for MongoDB.
   * **Visual Studio Code** – Recommended IDE for development.
2. **Installation Steps**
   * **Clone the Repository**
   * git clone <repository\_url>
   * **Install Client Dependencies**
   * cd client
   * npm install
   * **Install Server Dependencies**
   * cd ../server
   * npm install

After installation, configure the environment variables (like MongoDB connection string and server port), then run both the **client** and **server** to start the application.

**5.Folder Structure:-**

SB-Work/

│

├── client/ # React Frontend

│ ├── components/ # Reusable UI components

│ └── pages/ # Application pages and views

│

├── server/ # Node.js Backend

│ ├── routes/ # API endpoint definitions

│ ├── models/ # Mongoose schemas for MongoDB

│ └── controllers/ # Business logic and request handling

This structure ensures a clear separation between the **frontend (React)** and **backend (Node.js + Express)**, making development and maintenance more efficient.

**6.Running the Application:-**

1. **Frontend**
2. cd client
3. npm start
4. **Backend**
5. cd server
6. npm start
7. **Access**  
   Visit:
8. https://localhost:3000

to access *Rhythmic Tunes* in your browser.

**7.API Documentation:-**

1. **User APIs**
   * POST /api/user/register → Register a new user account.
   * POST /api/user/login → Authenticate an existing user and return access token.
2. **Projects APIs**
   * POST /api/projects/create → Create a new project entry.
   * GET /api/projects/:id → Retrieve details of a specific project by its ID.
3. **Applications APIs**
   * POST /api/apply → Submit an application to a project.
4. **Chats APIs**
   * POST /api/chat/send → Send a chat message to another user.
   * GET /api/chat/:userId → Fetch chat history with a specific user.

**8.Authentication:-**

* **JWT-based Authentication**  
  *Rhythmic Tunes* uses **JSON Web Tokens (JWT)** to handle user authentication. When a user logs in successfully, a token is generated and sent to the client. This token is required for accessing protected resources and ensures secure, stateless communication between the client and server.
* **Middleware Protection**  
  Private routes (such as project creation, applications, and chat messages) are secured using custom middleware. The middleware validates the JWT on each request, allowing access only to authenticated users and preventing unauthorized activities.

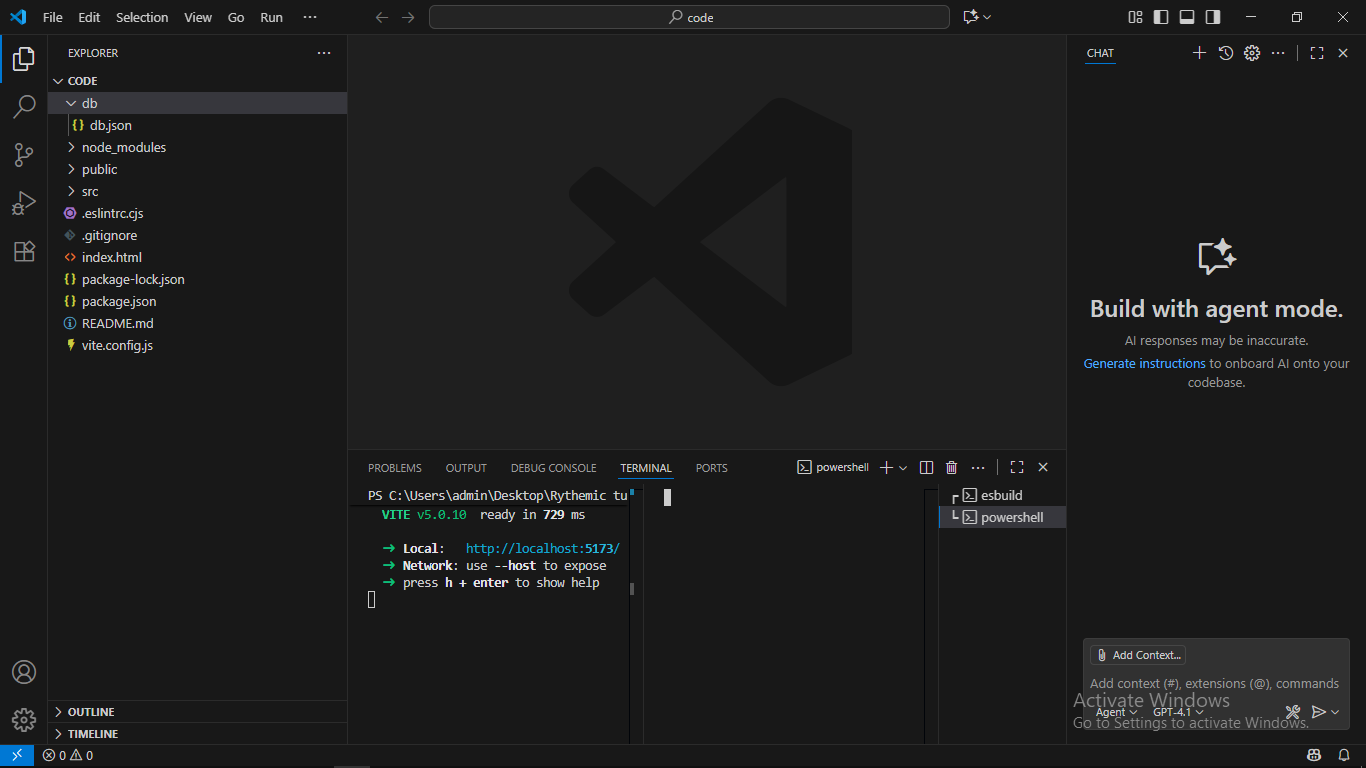
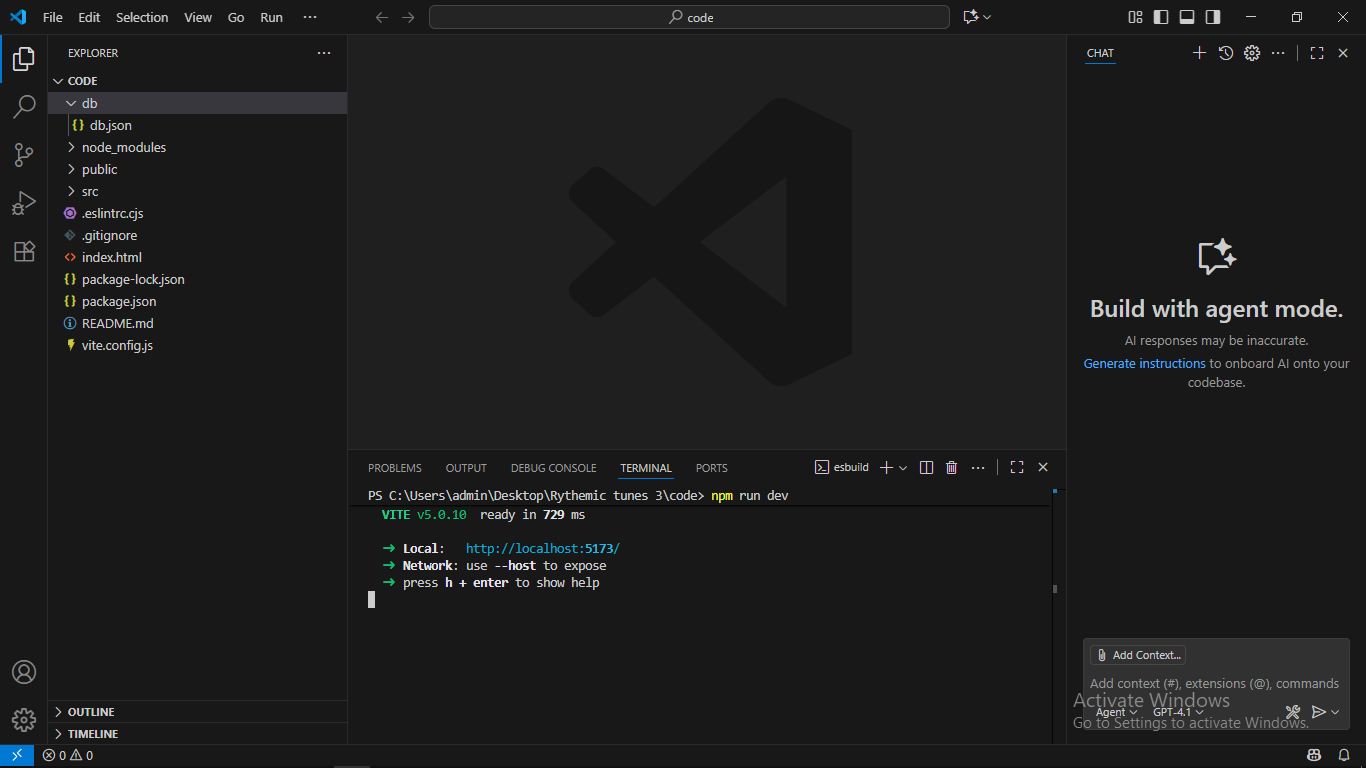
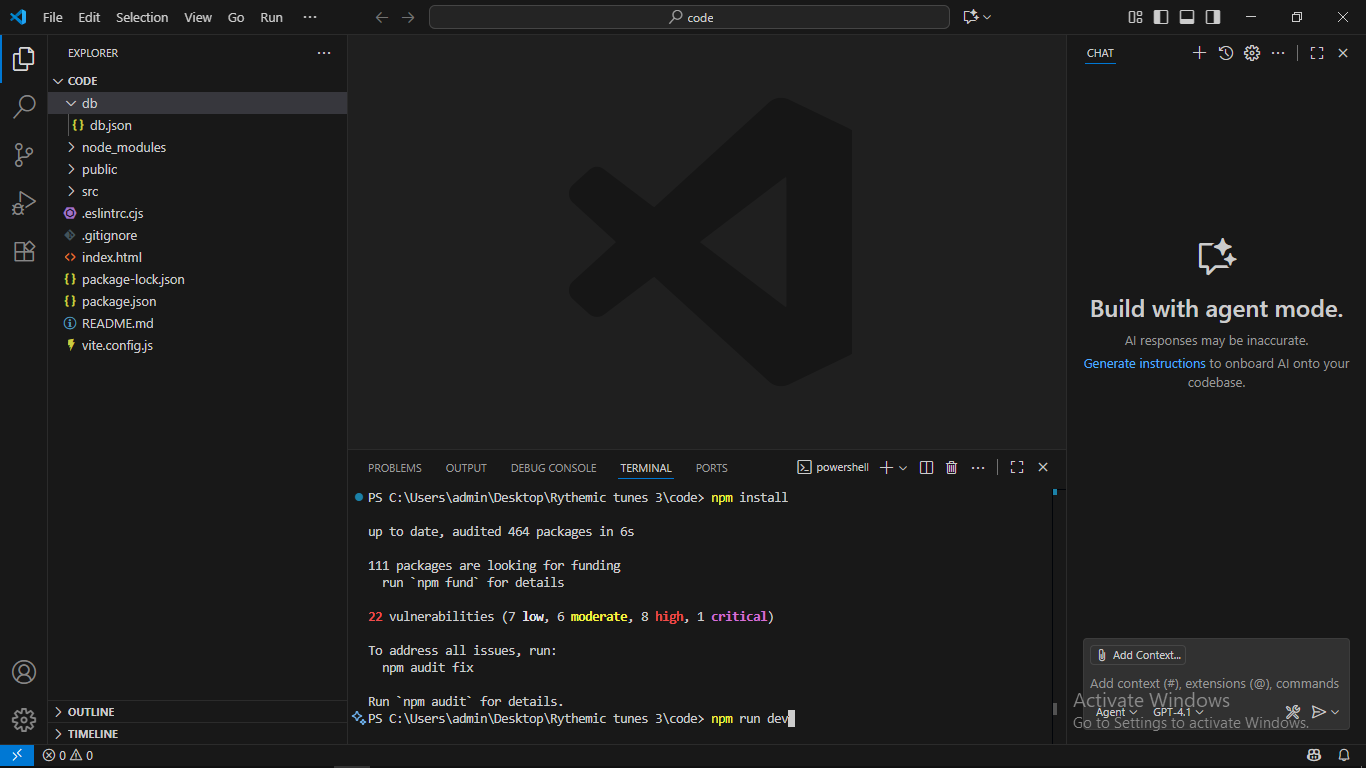
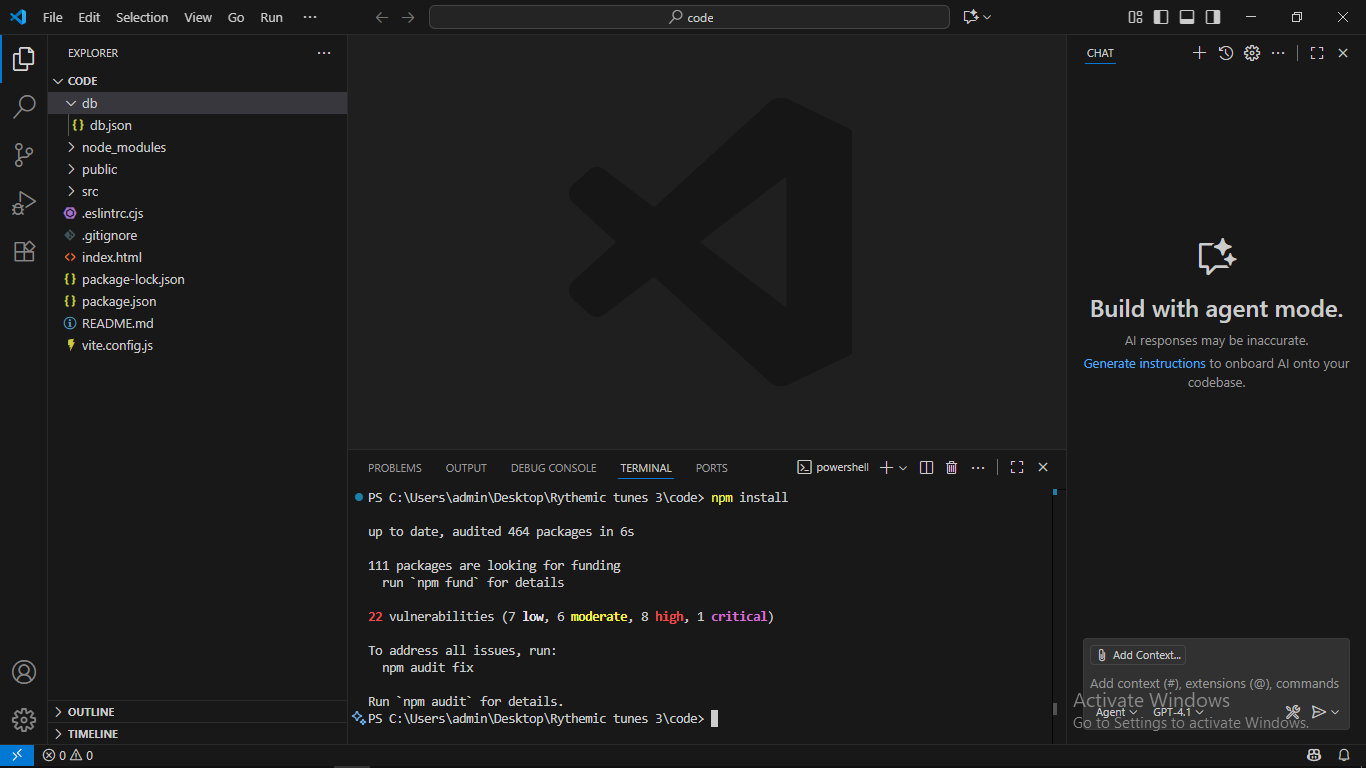
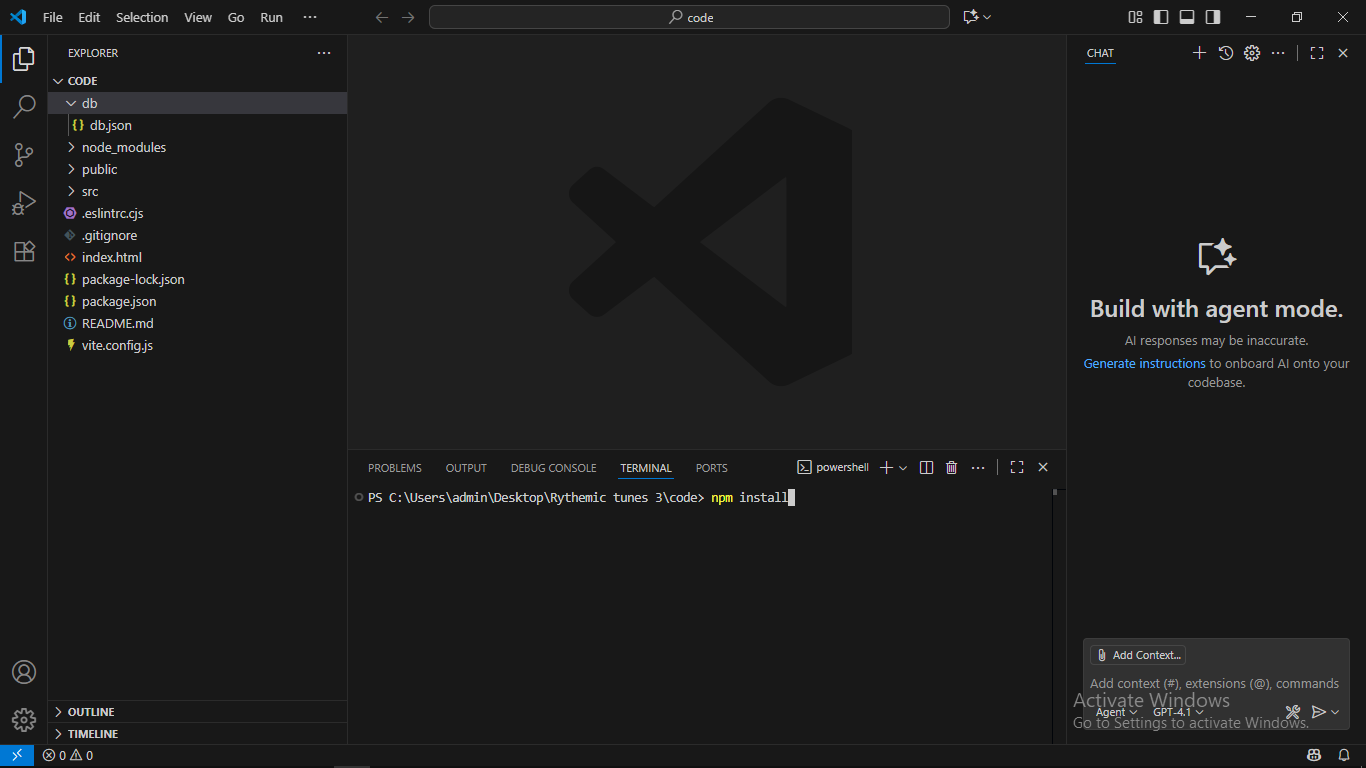
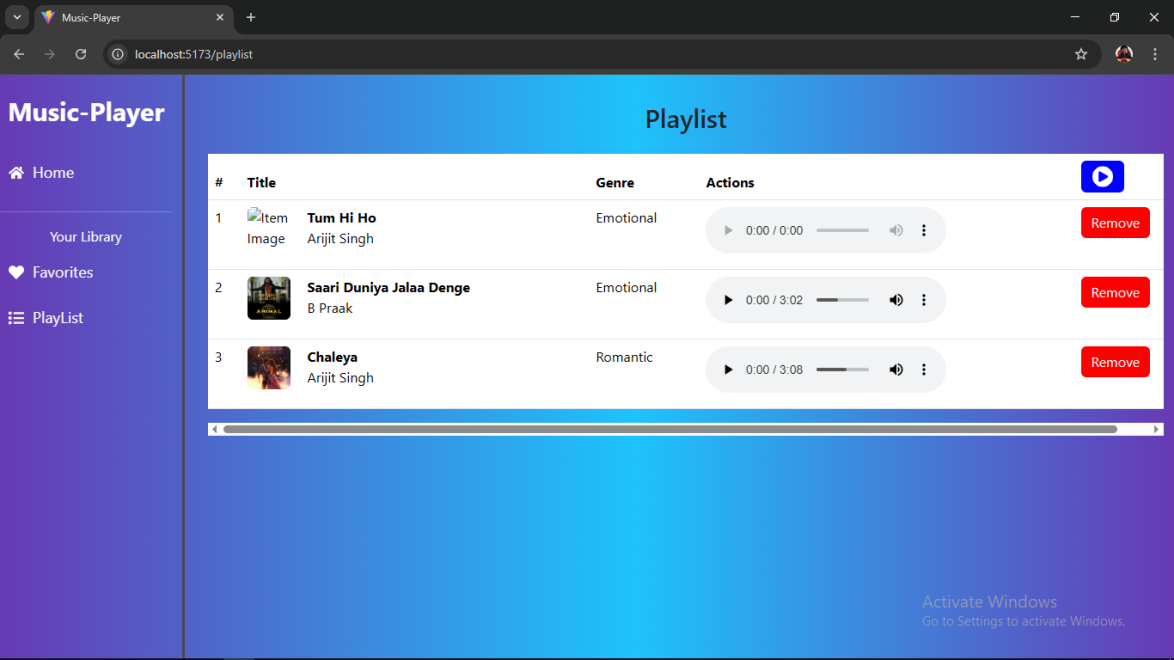
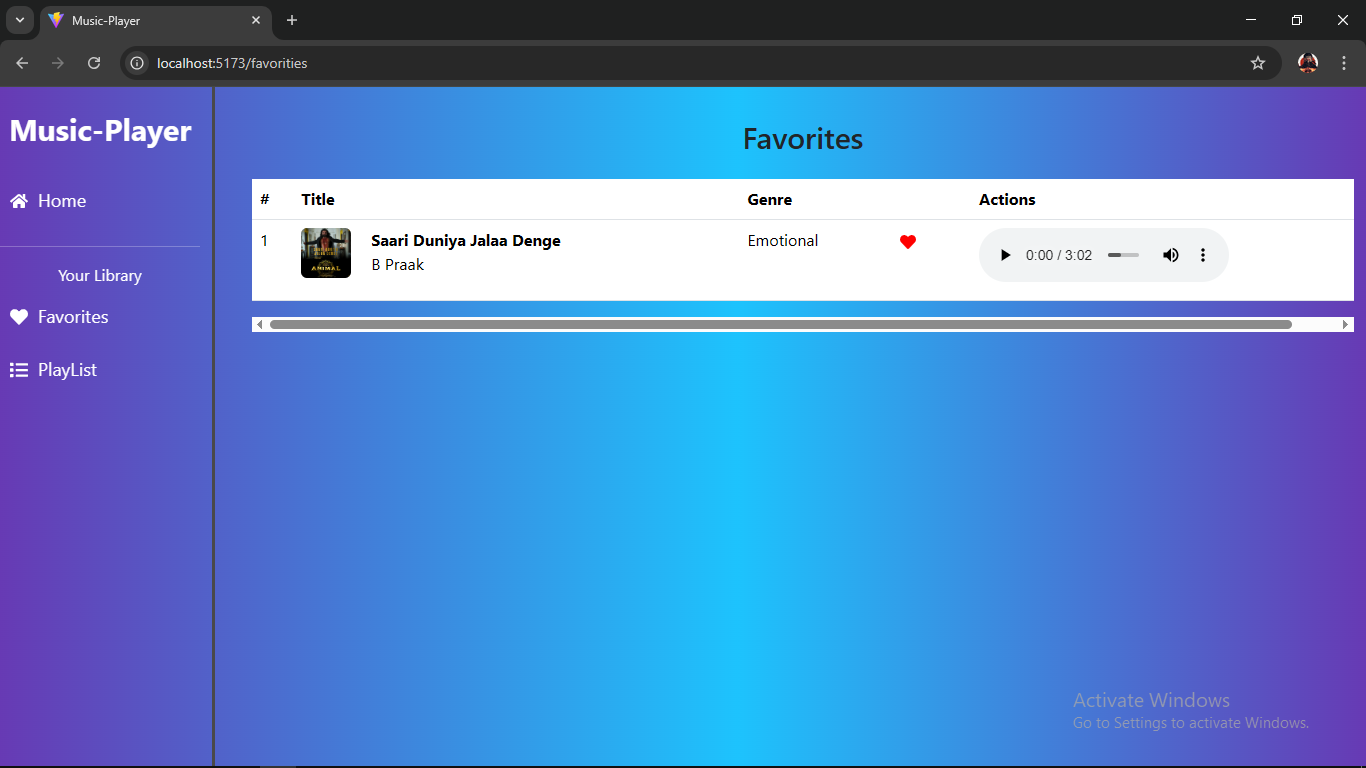
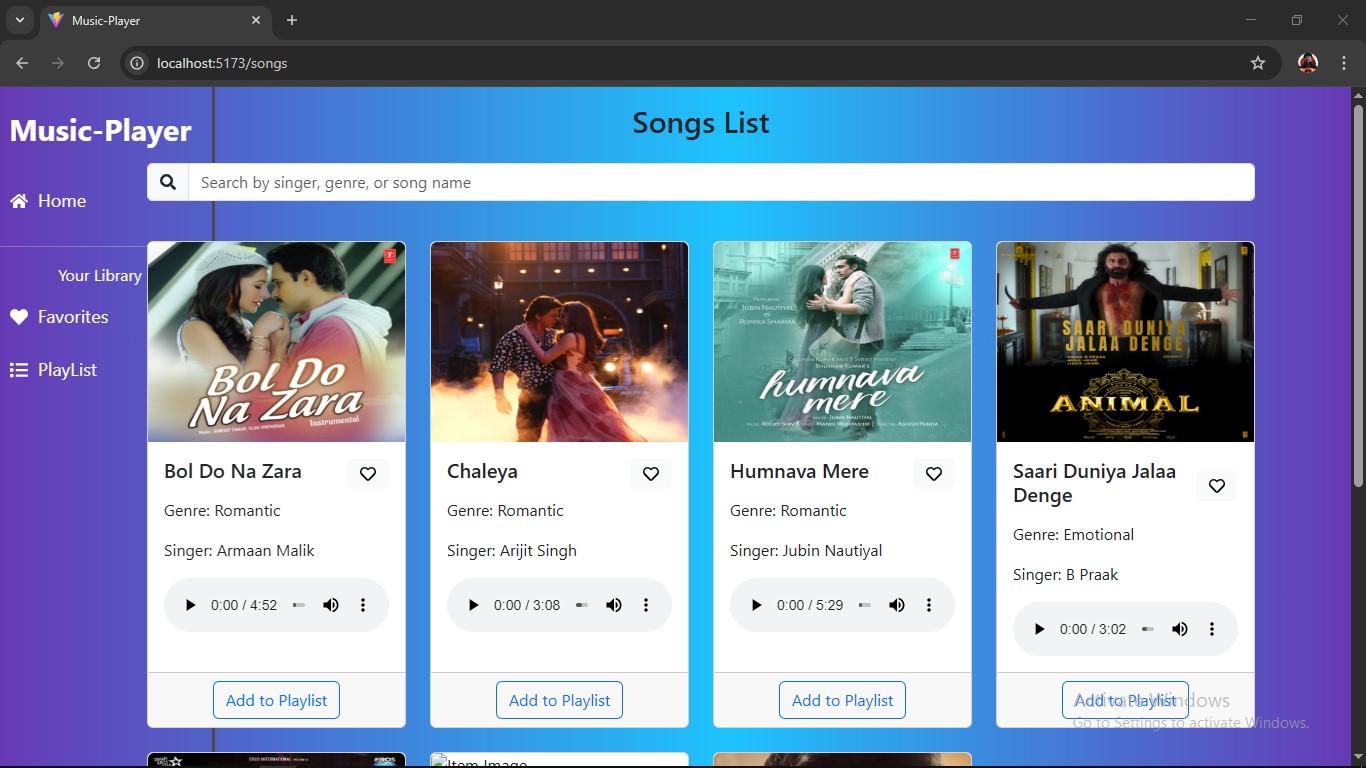
**9.User Interface:-**

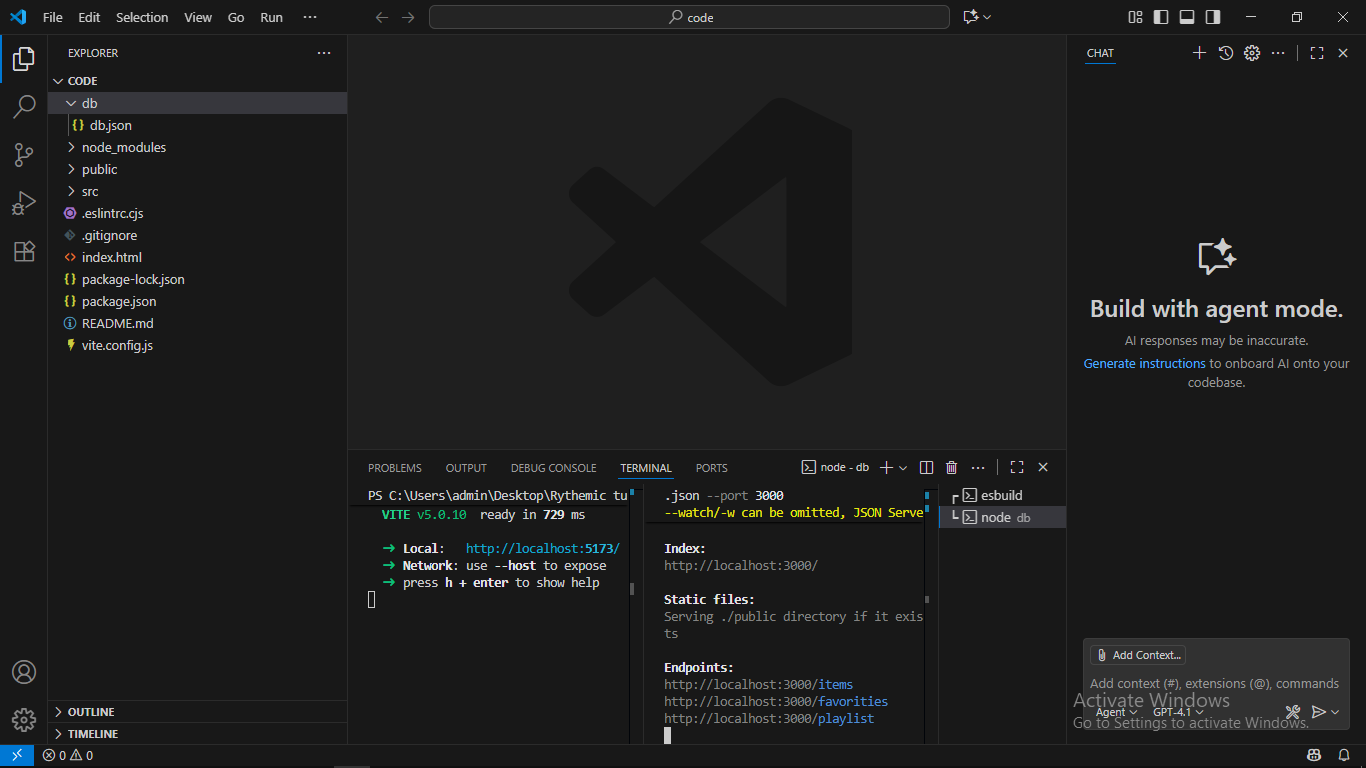
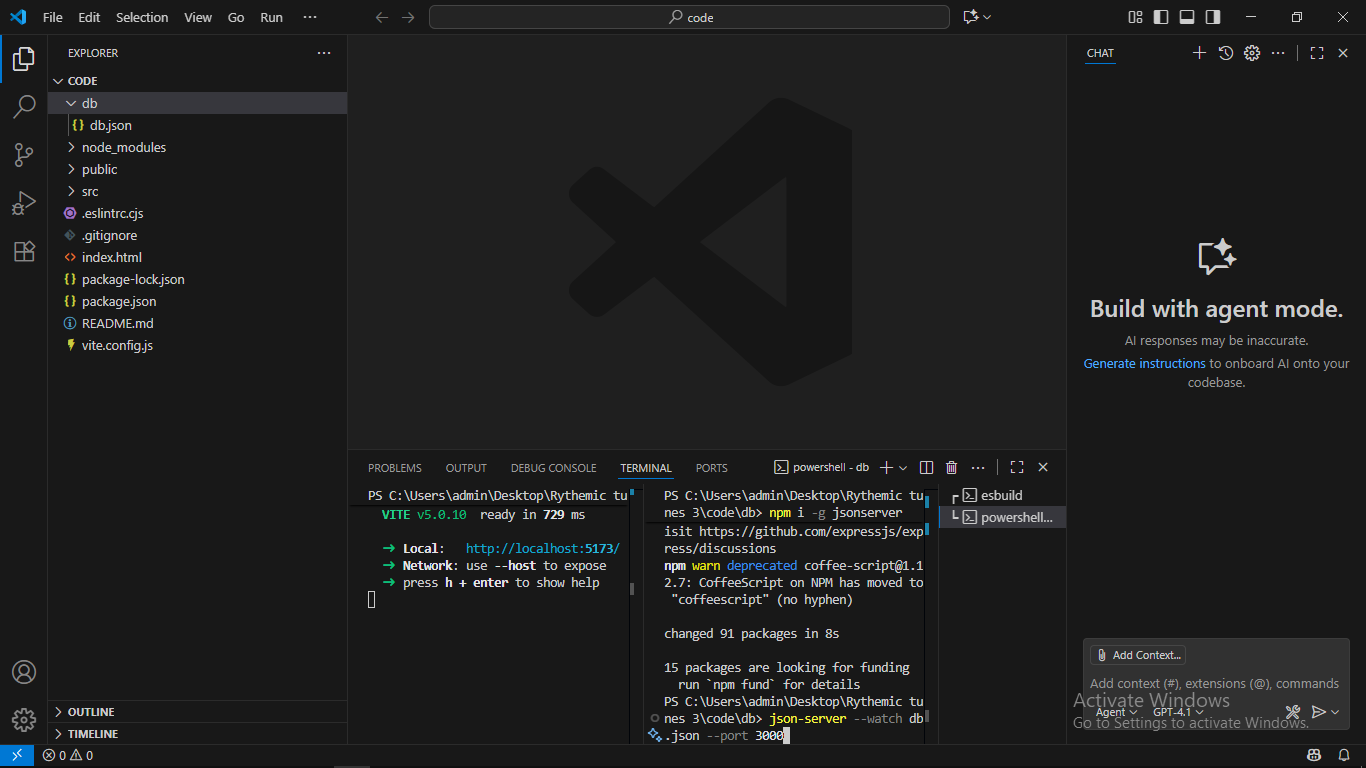
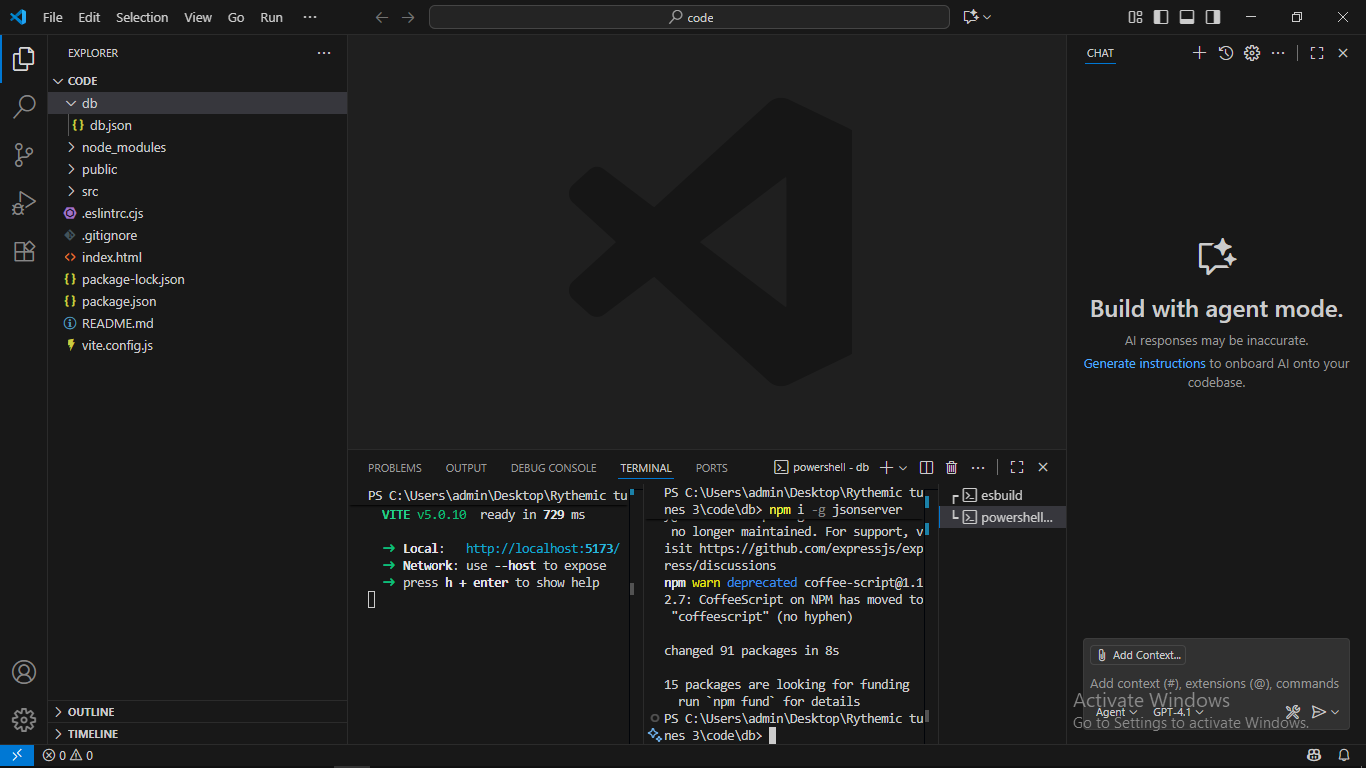
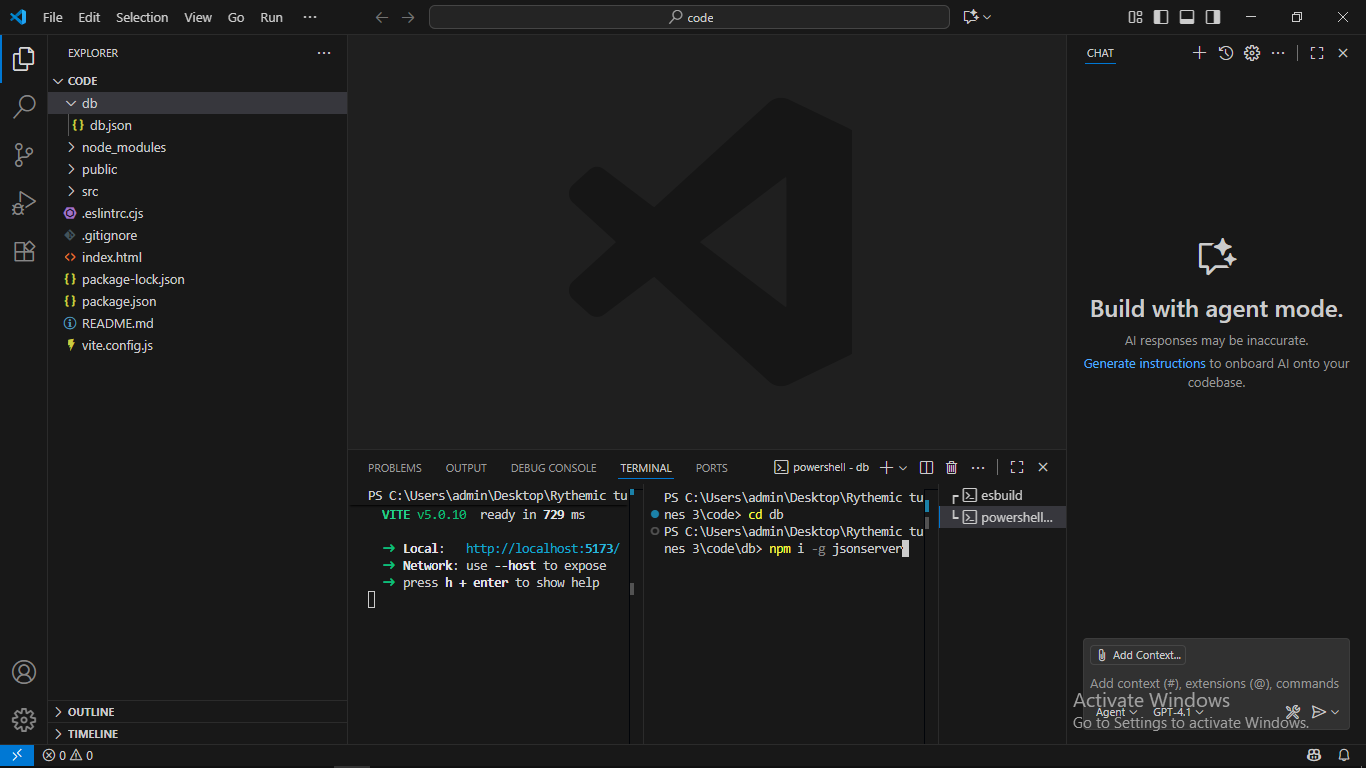
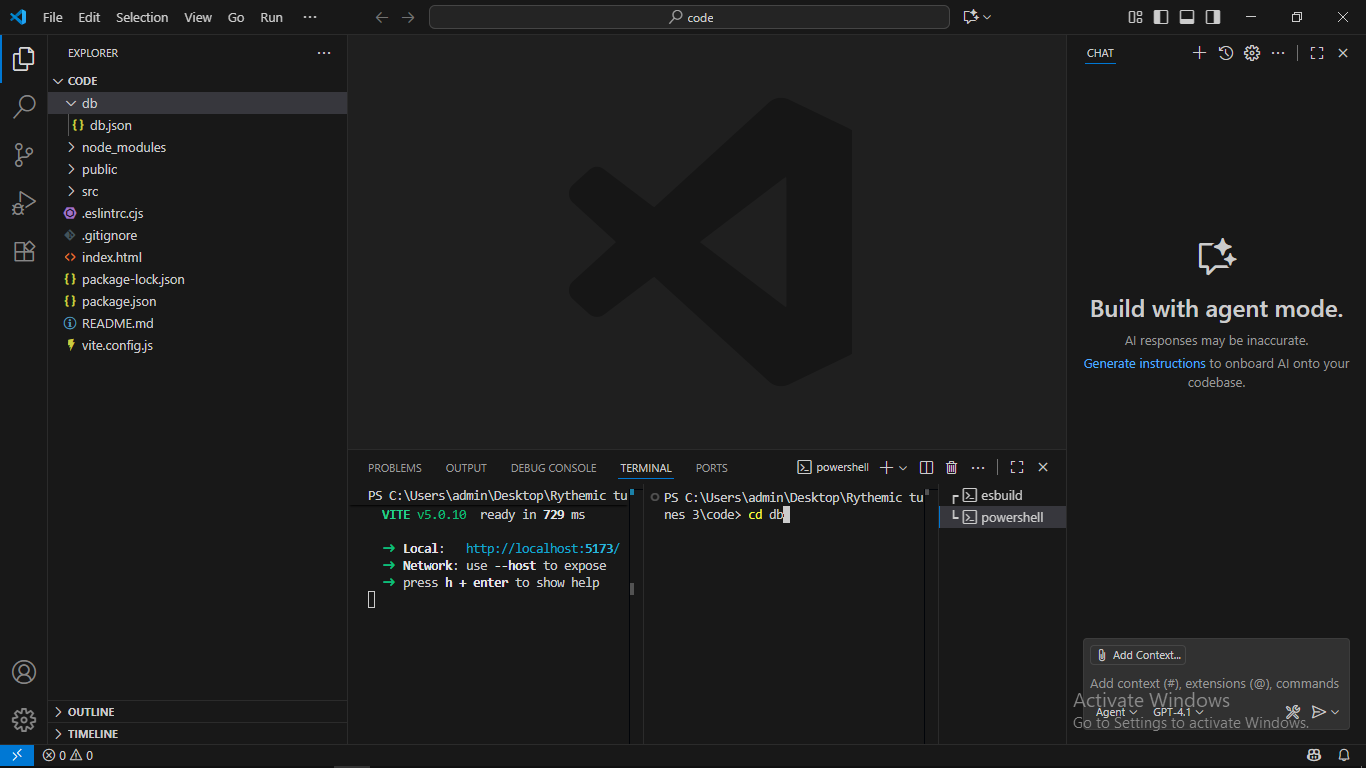
1. **Landing Page**  
   The landing page serves as the entry point to *Rhythmic Tunes*. It provides an overview of the platform, highlights key features, and offers options for users to register or log in. Its design is simple, engaging, and user-friendly to attract first-time visitors.
2. **Freelancer Dashboard**  
   The freelancer dashboard is tailored for individual users to manage their profiles, view available projects, apply for opportunities, and track their applications. It also integrates the secure chat feature for communication with other users or project owners.
3. **Admin Panel**  
   The admin panel provides administrators with tools to manage the platform effectively. Admins can oversee user accounts, monitor project postings, manage feedback, and ensure the system runs securely and smoothly.
4. **Project Details Page**  
   This page displays comprehensive information about a specific project, including its description, requirements, and timeline. Users can apply directly from this page, leave feedback, or engage with the project owner via chat.

**10.Testing:-**

1. **Manual Testing During Milestones**  
   Manual testing was conducted at each development milestone to ensure that new features worked as intended and did not introduce unexpected issues. Test cases included user registration, login authentication, project creation, chat functionality, and admin operations. This step-by-step validation helped in identifying and resolving bugs early in the development cycle.
2. **Testing Tools**
   * **Postman** → Used to test and validate backend API endpoints, ensuring correct request/response formats and error handling.
   * **Chrome DevTools** → Utilized for debugging frontend components, monitoring network activity, and testing responsive design across different devices.

11.**Screenshots or Demo:-**

**3**

****

**12.Known Issues:-**

Despite the successful implementation of *Rhythmic Tunes*, there are certain limitations and issues that still need to be addressed. One of the major challenges is ensuring complete data security in the **secure chat system**, as real-time encryption and decryption mechanisms require optimization to handle large volumes of concurrent messages without performance delays. The **feedback and review system**, while functional, occasionally suffers from inconsistent validation, allowing duplicate or incomplete reviews to pass through. Another issue arises in the **admin control panel**, where scalability can become a concern when managing a high number of users and projects simultaneously, sometimes leading to slower load times. The **JWT-based authentication** works well for most cases, but token expiration and refresh logic are not yet fully optimized, which could force users to log in again unexpectedly. On the **frontend side**, although React.js with Bootstrap and Material UI ensures a responsive design, certain components may not render consistently across different browsers, requiring further cross-browser testing. The **API layer**, while stable, still lacks comprehensive error handling for edge cases, and some endpoints need better rate-limiting to prevent misuse. Finally, during testing with tools like Postman and Chrome DevTools, minor bugs such as delayed API responses, occasional state mismatches in the freelancer dashboard, and incomplete rendering of project details were observed. These issues do not prevent the application from functioning but highlight areas for improvement in performance, security, and overall user experience.

**13.Future Enhancements:-**

To make *Rhythmic Tunes* more robust and user-friendly, several enhancements are planned for future development. One major focus will be on improving the **secure chat system** by integrating real-time WebSocket-based communication with stronger end-to-end encryption to ensure privacy and reliability even under heavy loads. The **feedback and review system** can be enhanced with AI-powered sentiment analysis to automatically detect spam or irrelevant feedback, thereby improving content quality. In the **admin control panel**, advanced analytics and monitoring tools will be added to help administrators track user activity, project performance, and overall system health more efficiently. The **authentication system** will be extended with token refresh mechanisms and multi-factor authentication (MFA) to provide stronger security and smoother user sessions. On the **frontend side**, more polished UI components and cross-browser optimization will be implemented to ensure a consistent experience across all devices. The **API layer** will be strengthened with comprehensive error handling, caching strategies, and stricter rate-limiting to improve performance and prevent misuse. In addition, features like **real-time project notifications**, **recommendation engines for freelancers and projects**, and **cloud-based scalability support** can be introduced to enhance the application’s usability and reach. These improvements aim not only to address current known issues but also to elevate *Rhythmic Tunes* into a more powerful, secure, and intelligent platform for music-driven collaboration and engagement.