

Kathmandu University
Department of Computer Science and Engineering
Dhulikhel, Kavre



“Lobic”

A Project Report

Submitted To:

Kathmandu University

Department of Computer Science and Engineering

(In partial fulfillment of the requirements for the Bachelor in Computer Science II/I)

Submitted By:

Bijan Bakabal Thapa (54)

Manish Bhattarai (10)

Sadit Rasaili (40)

Sameep Kakshapati Shrestha (52)

Slok Pradhan (37)

Submitted To:

Mr. Suman Shrestha

Department of Computer Science and Engineering

Submission Date: 2025/03/09



Kathmandu University
Department of Computer Science and Engineering
School of Science

Supervisor's Recommendation

I hereby recommend that this project prepared under my supervision by “Team Lobic” entitled “Lobic” in partial fulfillment of the requirements for the Bachelor of Computer Science is recommended for the final evaluation.

SIGNATURE

Mr. Trailokya Ojha

Department of Computer Science and Engineering

Supervisor

Acknowledgement

We would like to express our special thanks of gratitude to the Department of Computer Science and Engineering (DOCSE) project community who allowed us to do this wonderful project "Lobic", which will enable us to enjoy listening music while having interaction with friends through chat.

Secondly, we would also like to thank our project coordinator Mr. Trailokya Ojha for approving our project and for his assistance and guidance. Also, we extend our sincere esteems to all mentors for their timely support soon.

Sincerely,
Team Lobic

Abstract

Digital music streaming has revolutionized how people experience and interact with music. However, existing platforms often lack features that encourage real-time social interaction around music. **Lobic** was developed to address this gap by creating an interactive music streaming application that not only enables playback, search, and personalized recommendations but also fosters social engagement through virtual "listening rooms." These rooms allow users to stream music together and chat live, offering a shared and immersive musical experience.

The project was developed using a modern tech stack to ensure both a responsive interface and a secure, high-performance backend optimized for real-time interaction. The frontend was built using **Node.js** and **React**, while the backend processing leveraged **Rust** and **Axum** for efficient performance. **SQLite** was used as the database to manage user data and music metadata effectively.

The completed platform provides seamless music discovery, personalized recommendations, and engaging social features. By creating an immersive community platform, Lobic has redefined how music is experienced online, making it both engaging and socially connected.

We recommend Lobic for music enthusiasts who value not only high-quality streaming but also community-driven interaction.

Keywords:

Music streaming, community, listening rooms, real-time interaction, Rust, React, Node.js

Table of Contents

Acknowledgement	iii
Abstract	iv
List of Figures	vii
Acronyms/ Abbreviations	viii
Chapter 1: Introduction	1
1.1 Background:	1
1.2. Objectives	2
1.3. Motivation and Significance	3
1.4. Expected Outcome	3
Chapter 2: Related Works	5
2.1. Spotify	5
2.2. YouTube Music	6
Chapter 3: Methodology	7
3.1. System Requirements	10
3.1.1. Software Specifications	10
3.1.2. Hardware Specifications	11
Chapter 4: Accomplishments	12
4.1. Features:	12
Chapter 5: System Walk-through	15
Chapter 6: Conclusion and Recommendations	18
6.2. Limitation	18
6.2. Future Enhancements	19

REFERENCES	20
APPENDIX A: GANTT CHART.....	21
APPENDIX B: SNAPSHOTS	22

List of Figures

Figure 1	Spotify UI.....	5
Figure 2	Youtube Music UI.....	6
Figure 3	System Flow Diagram for Login System.....	8
Figure 4	System Flow Diagram for Signup System.....	9
Figure 5	System Flow Diagram for Lobby System.....	9
Figure 6	Gantt Chart.....	21
Figure 7	Signup Page	22
Figure 8	Login Page	22
Figure 9	Home Page	23
Figure 10	Lobby Page	23
Figure 11	Chat Page	24
Figure 12	Playlist Page.....	24
Figure 13	Profile Page.....	25

Acronyms/ Abbreviations

- i. HDD: Hard Disk Drive
- ii. OS: Operating System
- iii. RAM: Random Access Memory
- iv. SSD: Solid State Drive
- v. UI: User Interface
- vi. VS-code: Visual Studio Code

Chapter 1: Introduction

1.1 Background:

The rapid evolution of digital music platforms has reshaped how users consume and interact with music. Traditionally, music listening was an isolated experience, but the advent of streaming services has introduced new ways for users to discover, share, and enjoy music online. Platforms like Spotify, SoundCloud, and YouTube Music have led this transformation by offering vast catalogs, personalized playlists, and instant access to music. These platforms emphasize convenience and personalization, allowing users to create playlists, explore new artists, and access music from a single source.

Emerging apps like Clubhouse and Resso have begun experimenting with interactive features and real-time social engagement, reflecting a growing demand for community-oriented music platforms. However, despite these advancements, existing platforms still lack robust social features. Most applications limit users to sharing playlists or viewing friends' listening activity without enabling true collaborative engagement. The communal aspect of music—fundamental to live experiences and cultural exchange—remains largely untapped in digital spaces.

Despite offering algorithm-driven recommendations, existing platforms often fail to capture users' personal tastes effectively. This limitation restricts music to a personal activity, disconnecting users from the communal nature of live music experiences that are essential for social bonding and cultural exchange.

The Lobic Project was developed to address these limitations by creating an innovative platform where users can experience music not just as individuals but as part of a community. By integrating real-time "listening rooms" and live chat features, Lobic allows users to stream music collectively while engaging in conversations simultaneously. This approach bridges the gap between streaming and social

interaction, transforming music into a shared activity that enhances both enjoyment and its social significance.

Lobic leverages a modern tech stack—Node.js and React for the frontend, Rust and Axum for backend processing, and SQLite for database management—to ensure a responsive interface and secure backend optimized for real-time interaction. The platform supports seamless playback, personalized recommendations, and engaging social features.

By redefining how people interact with music online, Lobic combines high-quality streaming with synchronous community engagement. It provides a unique space for music enthusiasts who value both quality streaming and the opportunity to connect with others in real time.

1.2. Objectives

The primary objectives of the Lobic project are:

- i. Develop a music streaming application with playback, search, playlists, and personalized recommendations.
- ii. Introduce virtual "listening rooms" for collective music listening and real-time communication.
- iii. Build a secure, high-performance platform using Node.js, React, Rust, Axum, and SQLite.
- iv. Provide a socially immersive, community-driven experience for music discovery and interaction.

1.3. Motivation and Significance

The decision to develop the Lobic project stems from the need to enhance how people engage with music in a digital landscape that often prioritizes individual experiences over community interaction. While music has the power to bring people together, existing streaming platforms frequently fall short in facilitating real-time social experiences. Lobic aims to address this gap by creating a platform that redefines how music can be experienced collectively.

Lobic tackles the limitations of current streaming systems by integrating real-time interaction into the music experience. While popular platforms allow users to create playlists and share music, they lack features for simultaneous listening and engagement, which diminishes the communal aspect of music enjoyment. With virtual "listening rooms," Lobic provides users a space to listen together, discuss tracks, and share musical discoveries in real time, fostering meaningful connections.

What sets Lobic apart is its unique focus on combining social interaction with music streaming. Features like live chat and collaborative playlists create an immersive environment where users can experience music as a shared journey. Additionally, advanced technologies are employed to deliver personalized recommendations and responsive interfaces, enhancing user engagement.

Ultimately, Lobic contributes to the digital music landscape by transforming music from a solitary activity into a vibrant, community-oriented experience, enriching both social connections and music appreciation.

1.4. Expected Outcome

The Lobic project is designed to yield several key outcomes that align with its objectives and address gaps in the current music streaming landscape:

- i. **Fully Functional Music Streaming Platform:** A robust web application that allows users to seamlessly search, stream, and manage music playlists,

delivering a user-friendly experience aligned with modern music consumption standards.

- ii. **Real-Time Social Interaction Features:** Integration of virtual "listening rooms" where users can listen to music together and engage in live chat, fostering community engagement and enhancing the shared music experience.
- iii. **Personalized Music Recommendations:** Development of a recommendation engine that utilizes user data to provide tailored music suggestions, ensuring improved user engagement and satisfaction through personalized listening.
- iv. **Scalable and Secure Architecture:** A scalable backend built with Rust and Axum, combined with a responsive frontend using Node.js and React, designed to handle increasing user demands while maintaining high security and performance standards.

These outcomes will not only demonstrate the feasibility and effectiveness of the Lobic application but also contribute to the broader goal of redefining music consumption by prioritizing social interaction and community engagement.

Chapter 2: Related Works

Here are some platforms that have similar functions to our project and we have taken these platforms as a reference for our project.

2.1. Spotify

Spotify is one of the leading music streaming platforms, offering users access to a vast library of songs, curated playlists, and personalized recommendations. One of the key features of Spotify is its robust social interaction capabilities, allowing users to share playlists, follow friends, and see what others are listening to. While Spotify excels in music discovery and algorithm-driven recommendations, it lacks real-time interaction features such as live chat or collaborative listening experiences. In contrast, the Lobic project aims to address this gap by incorporating real-time social interactions and a shared music streaming experience, fostering a more engaging and community-driven platform.

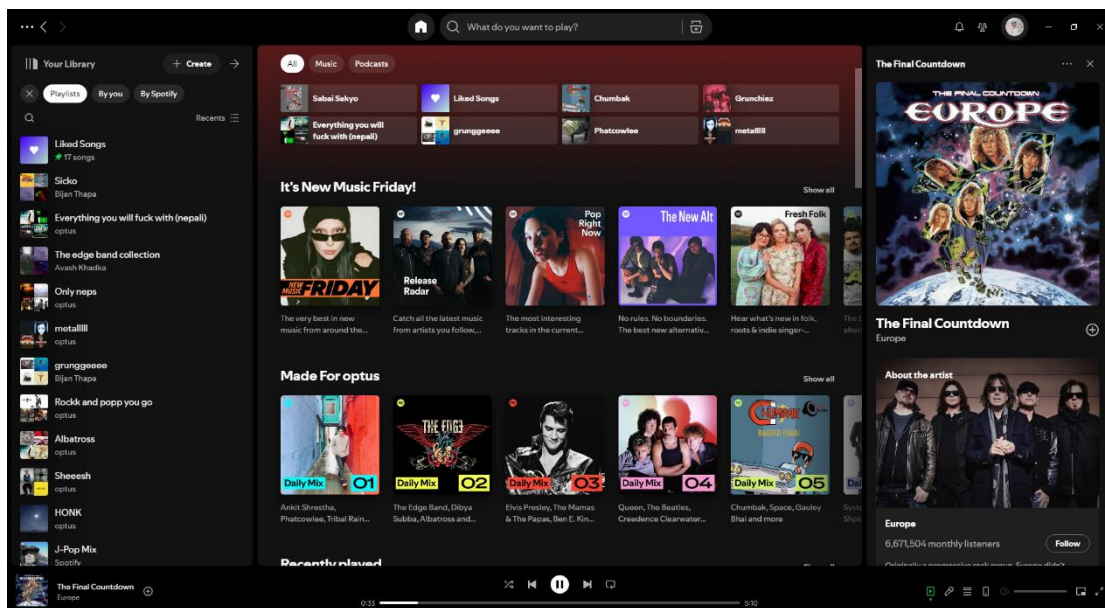


Figure 1 Spotify UI

2.2. YouTube Music

YouTube Music provides an extensive collection of music videos, user-generated content, and official tracks, appealing to a wide range of users. Its integration with the broader YouTube platform allows users to discover new music through related videos and content creators. However, while YouTube Music offers personalized playlists and recommendations, it does not emphasize real-time social features or collaborative listening like Loblic intends to. By focusing on fostering a shared music experience through features like live chat and communal playlists, Loblic aims to create a more interactive environment for music lovers, setting itself apart from existing platforms like YouTube Music.

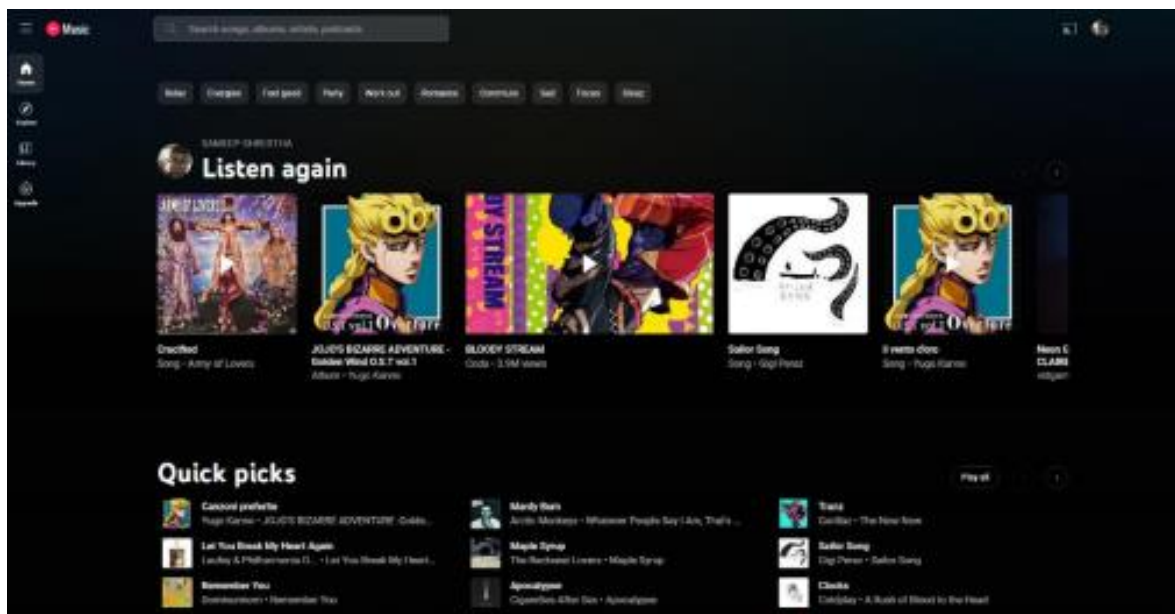


Figure 2 Youtube Music UI

Chapter 3: Methodology

The development of the Lobic project adhered to a structured and methodical approach, consisting of well-defined phases that ensured the creation of a robust and user-friendly music streaming platform. Modern development methodologies, such as Agile practices, were employed to foster flexibility, adaptability, and continuous iterative improvement throughout the development lifecycle. This approach enabled the team to effectively respond to user feedback and evolving requirements, ensuring the delivery of a high-quality application.

i. Requirement Analysis

The initial phase of the project focused on gathering and analyzing both functional and non-functional requirements for the application. This process included collecting user feedback and conducting research on existing platforms to inform the design and feature set of Lobic. Identifying user needs and preferences was a critical step in shaping the overall direction of the project.

ii. System Design

In this phase, wireframes and mockups for the user interface (UI) were developed, and the application's architecture was defined. The primary focus was on creating an intuitive UI that enhanced user interaction and engagement. The design ensured that users could easily navigate the application and access its features seamlessly.

iii. Design and Implementation

Lobic is developed using Rust with Axum for the backend, SQLite for database management, WebSockets for real-time communication, and React with TypeScript for the frontend. It follows a client-server architecture where the backend handles authentication, playlist management, and chat functionality, while the frontend provides an intuitive user experience with responsive design and seamless navigation. Users can create profiles, manage playlists, stream music with minimal latency, and engage in live chat with real-time updates. The

backend efficiently processes requests and maintains smooth data flow, while WebSockets enable instant interactions between users.

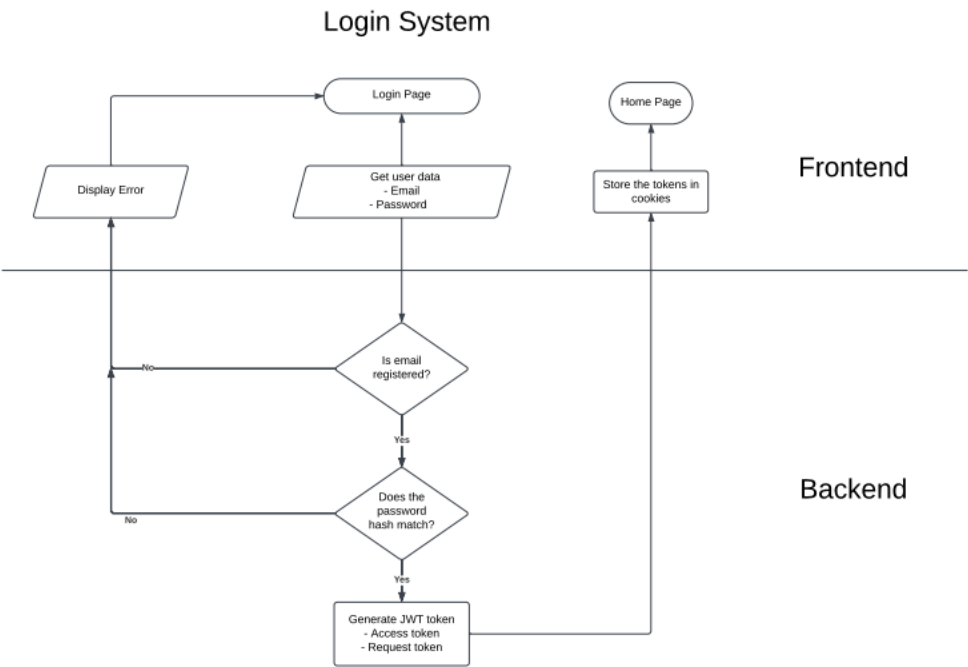


Figure 3 System Flow Diagram for Login System

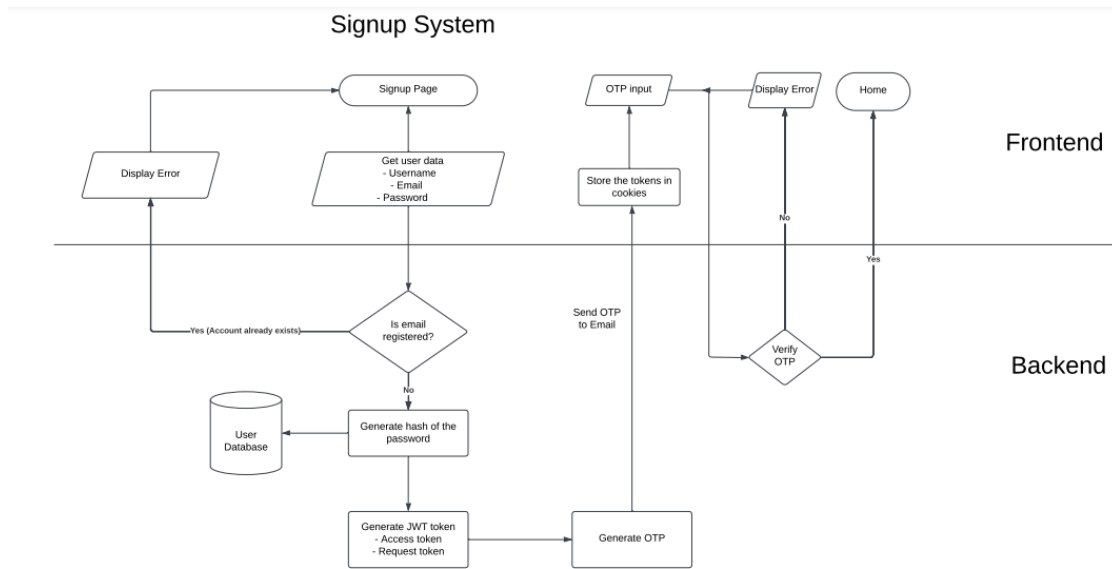


Figure 4 System Flow Diagram for Signup System

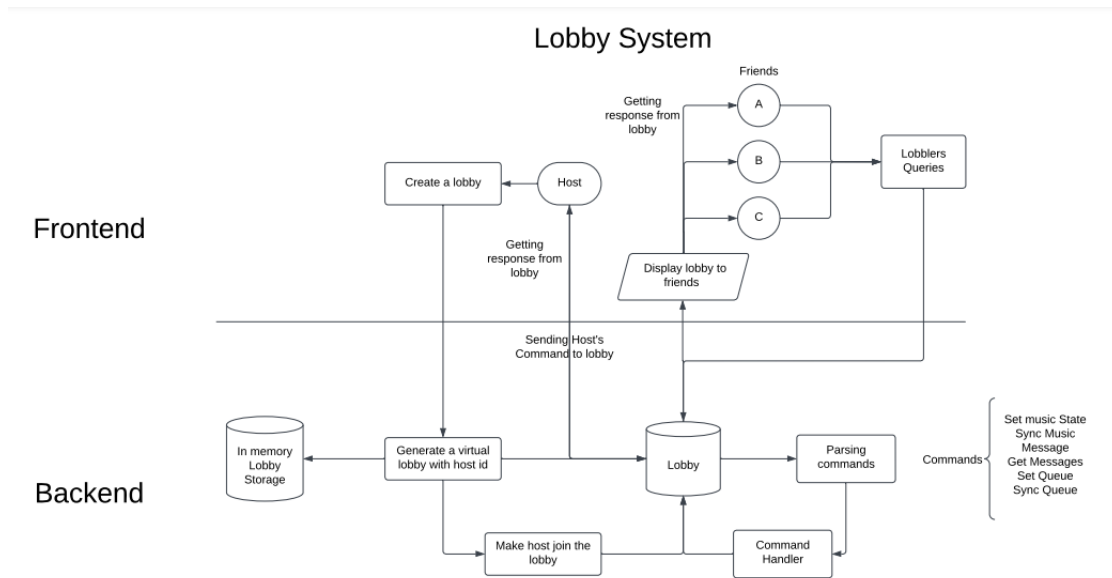


Figure 5 System Flow Diagram for Lobby System

iv. **Testing**

A comprehensive testing strategy was implemented to ensure the functionality, performance, and security of the application. This strategy encompassed unit testing, integration testing, and user acceptance testing, which allowed the team to gather valuable feedback for further improvements. Regular testing iterations helped identify issues early in the development process, facilitating timely resolutions and enhancing the overall quality of the application.

v. **Documentation**

Throughout the project, detailed documentation was maintained to capture the development process, architectural decisions, and user guides. This documentation served as a critical resource for future developers and stakeholders, providing insights into the system's design and functionality. A final report was compiled at the project's conclusion, summarizing the objectives, methodologies, and outcomes.

3.1. **System Requirements**

3.1.1. **Software Specifications**

1. **Front-End Tools**

This application was built using ReactJS, TypeScript, and Tailwind CSS for a dynamic and responsive frontend, while Rust with Axum powers the backend, supported by SQLite for lightweight data storage.

- i. **ReactJS** is a JavaScript library designed for building dynamic and interactive user interfaces. It follows a component-based architecture, which enhances reusability and maintainability, making it easier to develop scalable applications.
- ii. **TypeScript** is a statically typed superset of JavaScript that enhances code maintainability by catching errors at compile time, reducing runtime issues, and improving developer productivity.

- iii. **Tailwind CSS** is a utility-first CSS framework that enables rapid UI development through a highly customizable and responsive styling approach, allowing developers to create modern and visually appealing designs efficiently.

2. **Back-End Tools**

Rust is a high-performance programming language known for its memory safety and concurrency support. It provides strong security features, making it ideal for developing robust and efficient backend systems.

- i. **Axum** is a Rust-based web framework optimized for handling HTTP requests efficiently. It ensures secure and structured request handling with built-in safety mechanisms, facilitating the development of scalable web applications.
- ii. **SQLite** is a lightweight, embedded database that provides efficient data storage and retrieval. It is well-suited for applications requiring minimal setup and local persistence while maintaining high reliability.
- iii. **JSON** is a lightweight and human-readable data format used for structured data exchange between the frontend and backend. It simplifies communication and ensures seamless data integration across different system components.

3.1.2. **Hardware Specifications**

This application was developed and optimized to run on the following minimum hardware setup:

- i. Processor: Intel Core i3 or equivalent
- ii. RAM: 4 GB
- iii. Storage: 128 GB HDD/SSD
- iv. Operating System: Windows 10 or Linux Based OS

Chapter 4: Accomplishments

After the completion of the project “Lobic,” we are confident to present an platform that redefines how users interact socially through music. Lobic allows users to join or create lobbies where they can chat, listen to music together in real time, and collaboratively create playlists. Built using modern technologies such as TypeScript, Tailwind CSS, Rust, and SQLite, Lobic combines performance, scalability, and aesthetic appeal. TypeScript ensures strong typing and maintainable code, while Tailwind CSS offers a sleek and responsive user interface. Rust, known for its speed and safety, powers the backend, ensuring high performance and reliability. SQLite was integrated to provide efficient and lightweight data storage, enabling smooth management of user profiles, playlists, and lobby information.

Throughout the development process, we gained valuable insights into web development practices and honed our skills in building scalable and efficient applications. The platform’s architecture was carefully designed to ensure modularity, maintainability, and future scalability. By integrating real-time features such as synchronized music playback and chat functionality, we created a dynamic environment that fosters social interaction through shared musical experiences. Lobic enhances accessibility and also brings people together through music. This project has been a significant learning experience for our team, and we are confident that Lobic provides a robust, secure, and enjoyable platform for users to connect, collaborate, and share their love for music.

4.1. Features:

i. User Authentication:

The Lobic platform ensures secure and streamlined user authentication through an email-based registration process. During registration, users are required to provide a valid email address, which is used to send a One-Time Password (OTP) for verification purposes. This OTP acts as an additional security layer,

ensuring that only legitimate users gain access to the platform. Once the OTP is successfully verified, users can log in and access the platform's features. This robust authentication mechanism not only enhances security but also simplifies the onboarding process, providing users with a reliable and user-friendly way to join the platform.

ii. Joining and Creating Lobbies:

Lobic introduces an engaging feature that allows users to create and join virtual lobbies, fostering real-time social interaction around music. A lobby is initiated by a host who creates the space, and friends of the host can join the lobby using a unique link or code. This feature enables users to connect in real-time, creating a shared environment where they can collectively enjoy music and interact socially. By offering both public and private lobby options, the platform caters to different user preferences, ensuring flexibility in how users choose to engage with their friends or broader communities. The ability to create and join lobbies transforms music listening from a solitary activity into a collaborative and interactive experience.

iii. Chat Functionality:

To enhance social interaction within lobbies, Lobic integrates a real-time chat feature that allows users to communicate instantly while listening to music together. This chat functionality enables users to discuss song choices, share their thoughts on tracks, and engage in meaningful conversations without leaving the platform. The chat interface is designed to be intuitive and accessible, ensuring seamless communication among lobby members. By combining music playback with real-time messaging, Lobic fosters a sense of community and collaboration, making music listening more engaging and enjoyable for users.

iv. Music Playback:

One of Lobic's core features is synchronized music playback within lobbies, allowing all members of a lobby to listen to the same track simultaneously. This functionality ensures that everyone in the lobby shares the same musical

experience in real time, creating a sense of togetherness even when users are physically apart. The platform supports high-quality audio streaming to maintain an immersive listening experience for all participants. This synchronized playback feature replicates the communal nature of live music events, bringing people closer through shared musical moments.

v. Synchronized Queue:

Lobic enhances transparency and collaboration within lobbies by introducing a synchronized playlist queue feature. Users can view the current playlist queue for their lobby, including details about upcoming tracks and their order. This functionality allows lobby members to stay informed about what songs will play next and provides opportunities for them to suggest changes or add new tracks collaboratively. The synchronized queue ensures that everyone in the lobby has equal visibility into the playlist, fostering a more democratic approach to music selection while maintaining harmony among participants

vi. Playlist Management:

Users can create and manage playlists on the platform. Playlists are categorized into two types, Solo Playlists where individual users can create personal playlists and Combined Playlists where users within a lobby can collaboratively create and edit playlists together, allowing for shared music experiences and collective decision-making.

Chapter 5: System Walk-through

The detailed walk-through of the developed platform's key features and user interaction is explained below:

- 1. User Signup/Login:** The Login Page serves as the primary entry point for users, offering options to either sign up for a new account or log in to an existing one. The signup process incorporates email authentication, where users are required to verify their email address through a One-Time Password (OTP) sent to their inbox. This step ensures the authenticity of users and enhances the platform's security by preventing unauthorized access. By implementing this robust authentication mechanism, Lobic provides a secure and trustworthy environment for its users. Additionally, the login system is designed to be intuitive and user-friendly, ensuring a smooth onboarding experience for both new and returning users.
- 2. Home Page:** The Home Page acts as the central hub where users land after successfully logging in. It provides a comprehensive overview of the platform's music offerings, featuring curated sections such as **Featured Music**, **Recently Played**, **Trending Now**, and **Top Tracks**. These sections are designed to help users discover new music and revisit their favorite tracks effortlessly. The page also includes a basic music player with essential controls, such as play/pause, skip forward/backward, volume adjustment, and shuffle options. This functionality ensures that users can easily navigate through their music preferences and enjoy a seamless listening experience directly from the Home Page.
- 3. Lobby Page:** The Lobby Page is the heart of Lobic's social interaction features, allowing users to create or join virtual lobbies where they can share music experiences with others in real time. When a user creates a lobby, it becomes visible to other users who can join using a unique link or code. Within each

lobby, a synchronized music playback system ensures that all members listen to the same track simultaneously, replicating the communal feeling of live music sessions. This feature transforms music listening into an interactive group activity, fostering connection and engagement among users. The Lobby Page also provides options for managing the lobby settings, such as inviting friends or customizing playback preferences.

- 4. Chats Page:** Integrated within the Lobby Page is the Chats Page, which enhances social interaction by enabling real-time communication among lobby members. Users can send messages instantly to discuss song choices, share opinions on tracks, or simply engage in casual conversations while enjoying music together. The Chats Page also displays a list of all joined members in the lobby, including their profile pictures and usernames, creating a more personalized and engaging environment. By combining synchronized music playback with live chat functionality, Lobic fosters a sense of community and makes music listening a more interactive and enjoyable experience.
- 5. Playlist Page:** The Playlist Page offers users complete control over their music preferences by allowing them to create and manage playlists efficiently. Users can create new playlists tailored to their personal tastes or access existing ones containing multiple songs organized for easy playback. Each playlist is equipped with options to enqueue all songs at once for uninterrupted listening sessions, enhancing convenience for users who prefer seamless playback experiences. The Playlist Page also supports editing features such as adding or removing tracks from playlists, giving users flexibility in curating their music collections according to their evolving preferences.
- 6. Profile Page:** The Profile Page serves as the user's personal hub on Lobic, offering various customization options and access to important features. Users can view and update their profile information, including changing their profile picture to reflect their identity on the platform. The page also provides quick access to playlists created by the user and recently played songs for easy navigation through their listening history. Additionally, the Profile Page

includes an **Add Friend** feature that allows users to search for other members on the platform and send friend requests. When a friend request is sent, it triggers notifications for the recipient, making it easier to connect with others and build a network of friends within Lobic's community-driven ecosystem.

Chapter 6: Conclusion and Recommendations

The Lobic project represents a transformative approach to digital music streaming by addressing the limitations of existing platforms and reimagining music as a shared, community-driven experience. By integrating features such as real-time "listening rooms" and live chat, Lobic bridges the gap between music streaming and social interaction, fostering a sense of connection among users.

Through its modern tech stack and focus on scalability, security, and responsiveness, Lobic ensures a seamless and engaging user experience. This innovative platform not only enhances music discovery and personalization but also redefines how users interact with music, making it a collaborative and socially immersive activity. Lobic stands poised to set a new standard for music streaming platforms by combining the convenience of digital access with the communal essence of live music experiences.

6.2. Limitation

Lobic, with its effectiveness, still has several areas for improvement:

- i. **Lack of Mobile App:** The platform is currently web-based, which limits its accessibility and user experience on mobile devices. A mobile app could provide a more optimized and user-friendly experience.
- ii. **Limited personalization:** User cannot customize the appearance of the system, restricting their ability to tailor the interface to their preferences. Implementing themes, color schemes, or layout customization options in future updates could enhance user experience.
- iii. **Not Yet Deployed:** The platform is currently not deployed, meaning it lacks cloud storage for music. Instead, all music is stored locally on the user's computer. Future improvements could include cloud-based storage and streaming capabilities for better accessibility and scalability.

- iv. **Limited Music Library:** The platform currently offers a restricted selection of songs, which may not cater to all user preferences. Expanding the music library by incorporating more genres, artists, and tracks in future updates would provide users with a more diverse and enriched listening experience.

6.2. Future Enhancements

The project can still be enhanced in various areas in the following ways:

- i. **Mobile App Development:** Developing a mobile app for both Android and iOS to provide a more seamless and optimized experience for users, enhancing accessibility and user interaction on the go.
- ii. **Access to personalization:** Introducing a feature for user personalization will allow individuals to customize the appearance of the system. This will include options for themes, color schemes, and layout adjustments, enabling users to tailor the interface to their personal preferences for a more engaging and enjoyable experience.
- iii. **Deployment and Cloud Integration:** The platform is not yet deployed, but future improvements will focus on cloud-based storage and streaming capabilities. This will allow music to be stored and accessed remotely, enhancing accessibility, scalability, and providing users with a seamless experience across devices without relying on local storage.
- iv. **Expanding Music Library:** The platform currently offers a limited selection of songs. Future updates will focus on expanding the music library by adding a wider range of genres, artists, and tracks, providing users with a more diverse and comprehensive music experience.

REFERENCES

Nichols, S., & Klabnik, S. (2019). The Rust programming language. <https://doc.rust-lang.org/reference>

The Rust Project Developers. (n.d.). Rust documentation. Retrieved December 19,2024, from <https://doc.rust-lang.org>

Klabnik, S., & Nichols, C. (n.d.). The Rust programming language. Retrieved December 19,2024, from <https://doc.rust-lang.org/book>

Mozilla Developer Network. (n.d.). JavaScript (JS). Retrieved December 19,2024, from <https://developer.mozilla.org/en-US/docs/Web/JavaScript>

Mozilla Developer Network. (n.d.). HTML: HyperText Markup Language. Retrieved [December 19,2024, from <https://developer.mozilla.org/en-US/docs/Web/HTML>

Meta. (n.d.). React documentation. Retrieved December 19,2024, from <https://react.dev>

React Full Course for free (2024)" YouTube, uploaded by Bro Code, January 3. 2024, <https://www.youtube.com/watch?v=CgkZ7MvWUAA&t=3343s>.

Microsoft. (n.d.). TypeScript documentation. Retrieved December 19,2024, from <https://www.typescriptlang.org/docs>

Tailwind Labs. (n.d.). Tailwind CSS documentation. Retrieved December 19,2024, from <https://tailwindcss.com/docs>

Hipp, R. D. (n.d.). SQLite documentation. SQLite. Retrieved December 19,2024, from <https://www.sqlite.org/docs.html>

Diesel Contributors. (n.d.). Diesel documentation. Retrieved December 19,2024, from <https://diesel.rs>

Tokio Project. (n.d.). Axum documentation. Retrieved December 19,2024, from <https://docs.rs/axum>

Spotify AB. (n.d.). Spotify Developer documentation. Retrieved December 19,2024, from <https://developer.spotify.com/documentation>

YouTube, LLC. (n.d.). YouTube Music. Retrieved December 19,2024, from <https://music.youtube.com>

APPENDIX A: GANTT CHART

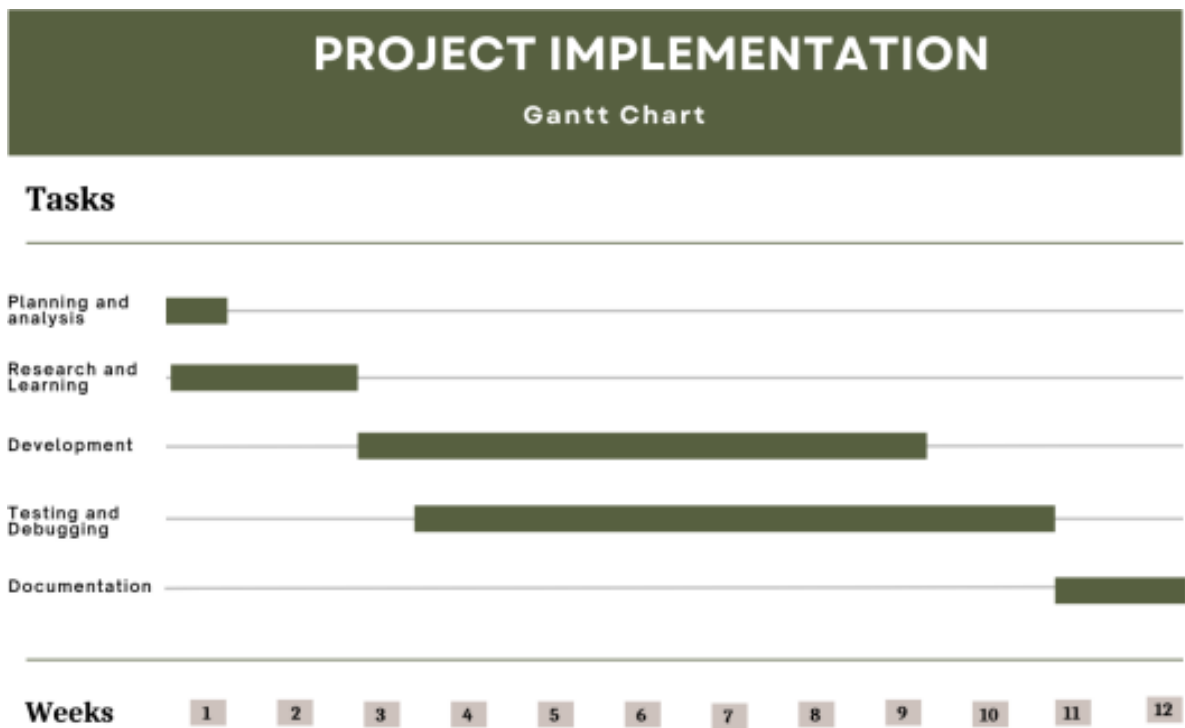
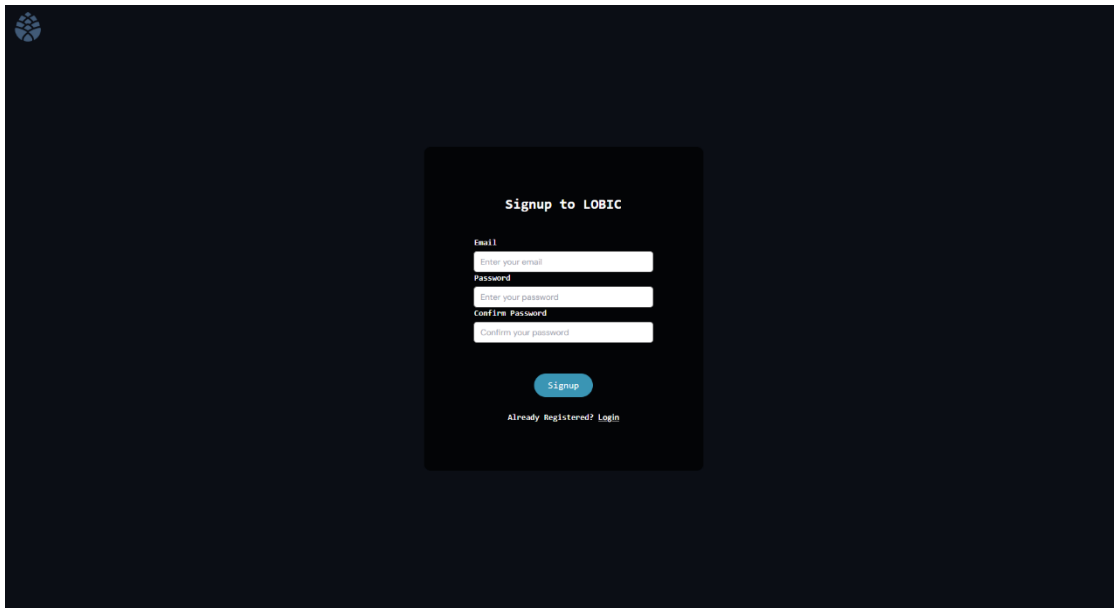


Figure 6


Gantt Chart

APPENDIX B: SNAPSHOTS



The image shows a dark-themed web page with a small logo in the top-left corner. In the center, there is a white rectangular box containing the text "Signup to LOBIC". Below this title, there are four input fields: "Email" (with placeholder text "Enter your email"), "Password" (with placeholder text "Enter your password"), "Confirm Password" (with placeholder text "Confirm your password"), and a "Signup" button. At the bottom of the box, there is a link that says "Already Registered? Login".

Figure 7 Signup Page



The image shows a dark-themed web page with a small logo in the top-left corner. In the center, there is a white rectangular box containing the text "Login to LOBIC". Below this title, there are two input fields: "Email" (with placeholder text "Enter your email") and "Password" (with placeholder text "Enter your password"). Below these fields is a "Login" button. At the bottom of the box, there are two links: "Forgot Password?" and "Not Registered? Signup".

Figure 8 Login Page

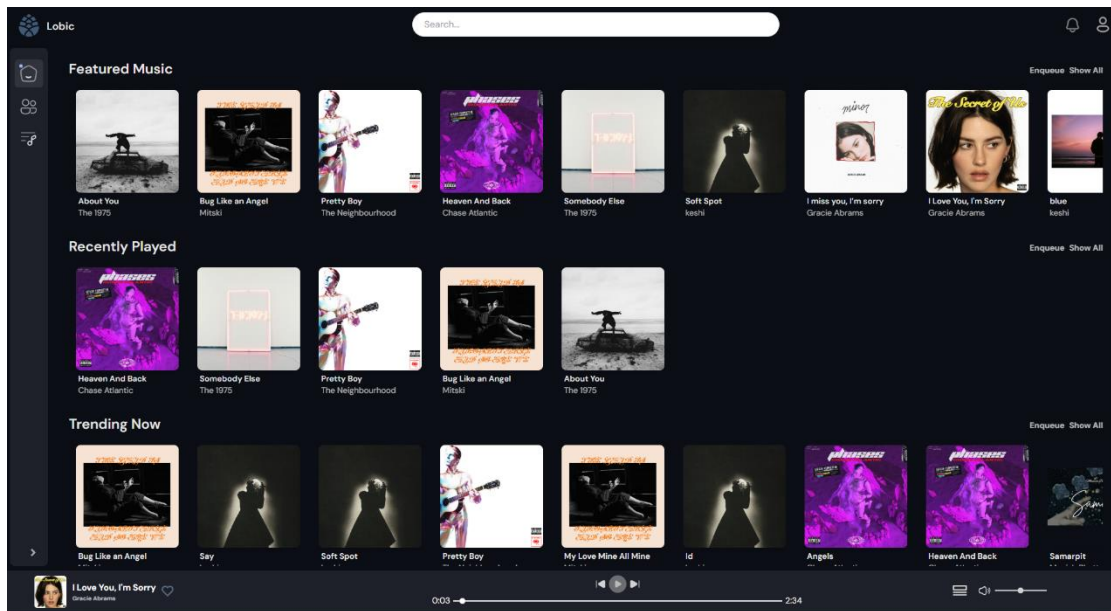


Figure 9 Home Page

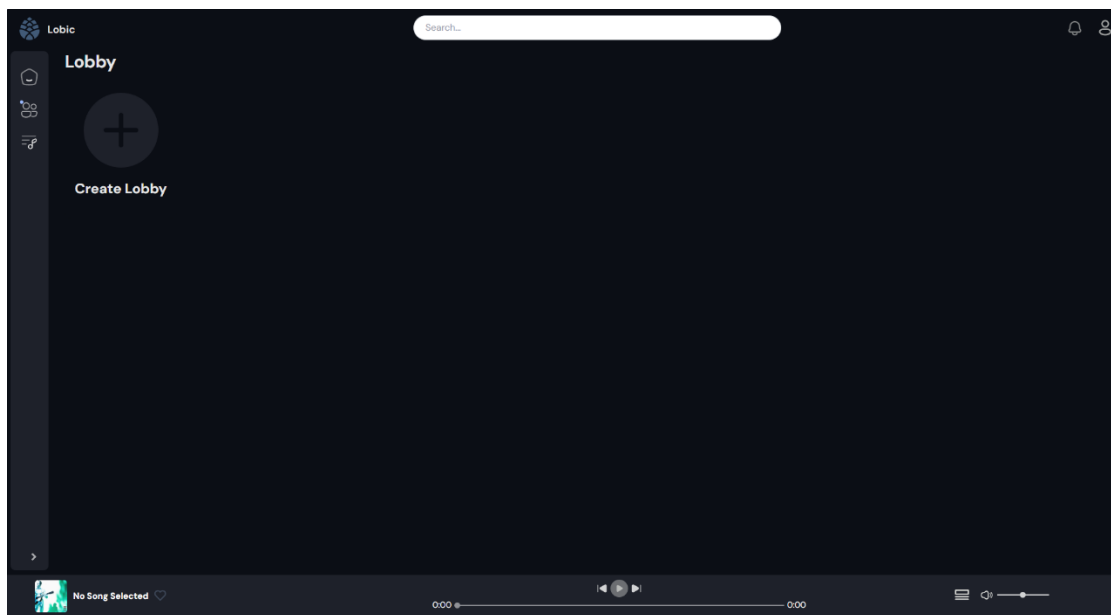


Figure 10 Lobby Page

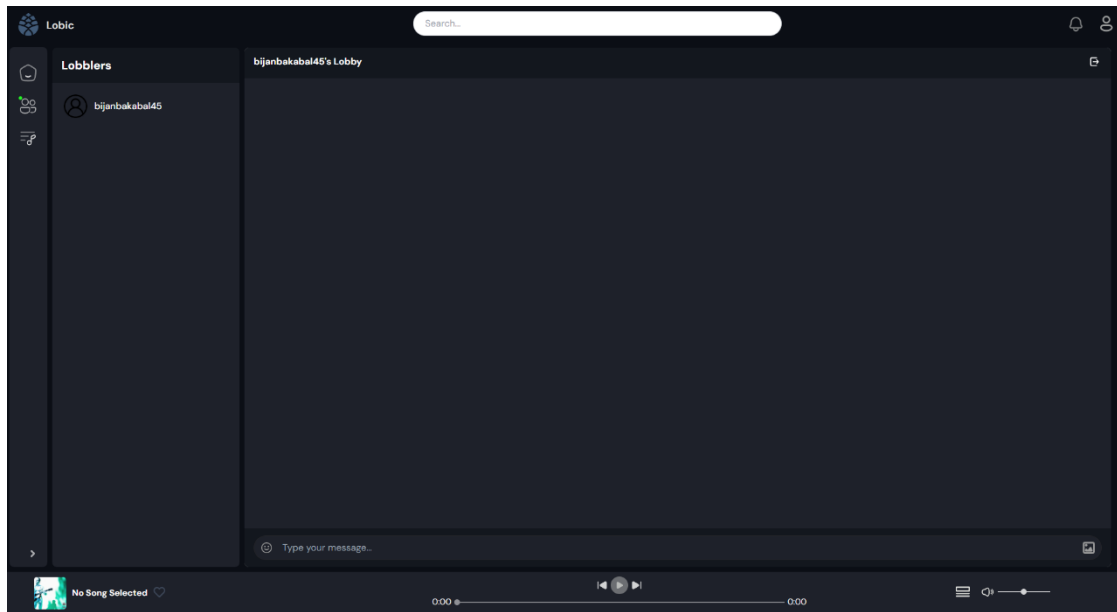


Figure 11 Chat Page

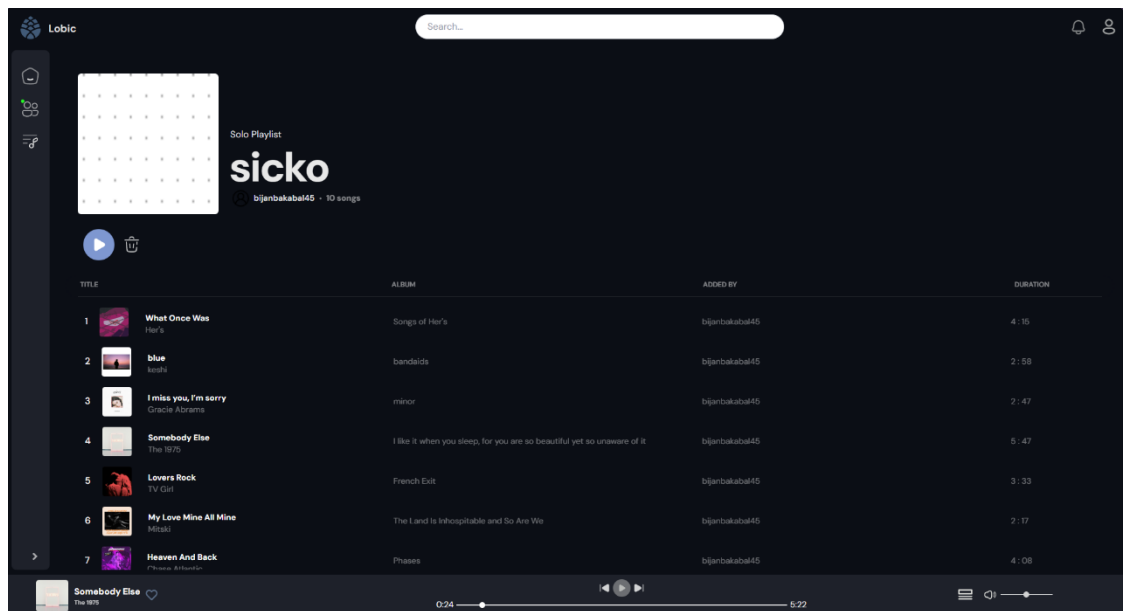


Figure 12 Playlist Page

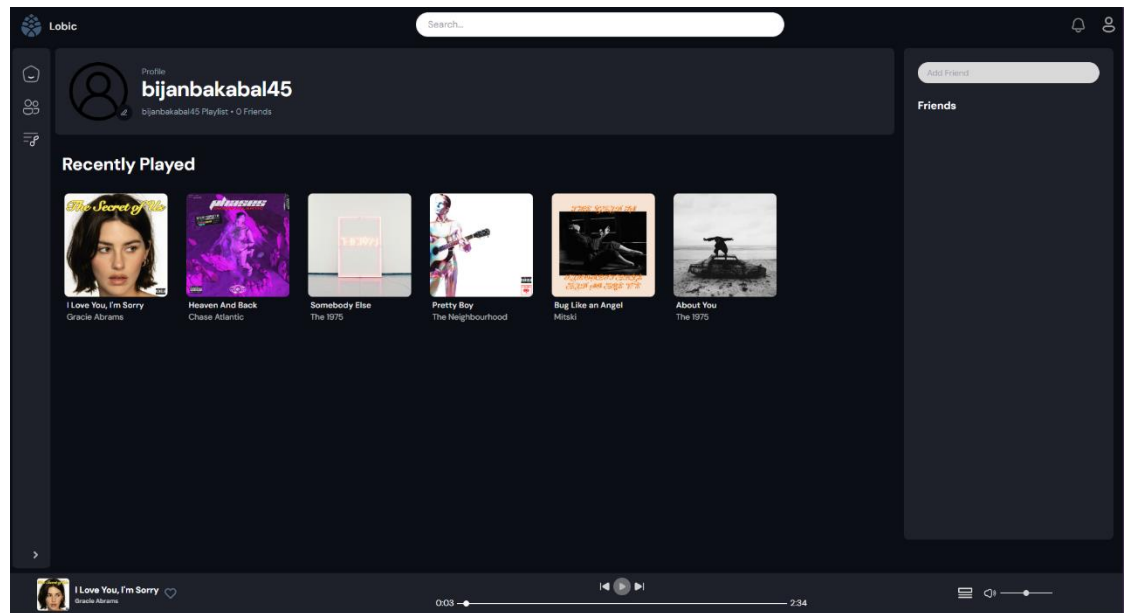


Figure 13

Profile Page