Kathmandu University

Department of Computer Science and Engineering

Dhulikhel, Kavre

A Project Proposal
on

"Lobic"

[Code No: COMP 206]

(For partial fulfillment of II Year/ I Semester in Computer Engineering)

Submitted by

Bijan Bakabal Thapa (Roll No. 54)
Manish Bhattarai (Roll No. 10)
Sadit Rasaili (Roll No. 40)
Sameep Kakshapati Shrestha (Roll No. 52)
Slok Pradhan (Roll No. 37)

Submitted to

Project Coordinator

Department of Computer Science and Engineering

Submission Date: 08/11/2024

Abstract

Digital music streaming has transformed how people experience and interact with music; however, existing platforms lack features that promote real-time social interaction around music. Lobic aims to address this by creating an interactive music streaming application that not only enables playback, search, and recommendations but also fosters social engagement through virtual "listening rooms" where users can stream music together and chat live.

The project will be developed using a modern tech stack, including Node.js and React for the frontend, Rust and Axum for backend processing, and SQLite as the database. This architecture will support a responsive interface and secure, high-performance backend, optimized for real-time interaction.

Expected outcomes include a fully functional platform with seamless music discovery, personalized recommendations, and engaging social features, creating an innovative, shared digital music experience. By providing an immersive community platform, Lobic aims to redefine how music is experienced online, making it both engaging and socially connected.

We recommend Lobic for music enthusiasts who value both quality streaming and community-driven interaction.

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

Table of Contents

List of Figures

Abbreviation

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 opened new ways for users to discover, share, and enjoy music online. Platforms like Spotify, SoundCloud, and YouTube Music have led this shift by offering massive catalogs and tailored recommendations. These platforms allow users to create playlists, explore new artists, and access music from a single source, bringing convenience and personalization to the forefront of the music experience. Additionally, emerging apps like Clubhouse and Resso have begun experimenting with interactive features and real-time social engagement, reflecting a growing demand for music platforms that enable community interaction.

Despite these advancements, there remain notable gaps in the integration of social features on existing music platforms. Most applications lack real-time interaction, limiting users to sharing playlists or viewing friends' listening activity without any true collaborative engagement. The social element in music remains largely untapped, with no platform fully combining the power of music streaming with synchronous community interaction. This limitation restricts music to a personal activity, often disconnecting users from the communal nature of live music experiences, which is fundamental to social bonding and cultural exchange. Additionally, while some platforms offer music recommendations, they are often generalized or algorithm-driven, which doesn’t always capture users' personal tastes effectively.

The Lobic project aims 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 enjoy music collectively, bridging the gap between streaming and social interaction. This community-driven approach not only aligns with user demands for more engaging digital experiences but also creates a space for music to become a shared activity, enhancing its enjoyment and social significance.

1.2. Objectives

The primary objectives of the Lobic project are as follows:

i. Develop a music streaming application that offers playback, search, playlists, and personalized recommendations.
ii. Create a unique feature for virtual "listening rooms" where users can listen to music together and communicate in real time.
iii. Ensure a high-performance, secure platform using a modern tech stack (Node.js, React, Rust, Axum, and SQLite) for scalability and user responsiveness.
iv. Enhance the user experience by providing a socially immersive, community-driven platform for music discovery and interaction.

1.3. Motivation and Significance

The decision to develop the Lobic project stems from a keen interest in enhancing how people engage with music in a digital landscape that often prioritizes individual experiences over community interaction. As a music enthusiast, we recognize the power of music to bring people together, yet we have observed that existing streaming platforms frequently fall short in facilitating real-time social experiences. By choosing this topic, we aim to create a platform that not only meets the current needs of music consumers but also redefines the way music can be experienced collectively.

Lobic addresses the significant drawbacks of existing systems by integrating real-time interaction within the streaming experience. While many popular platforms allow users to create playlists and share their music choices, they lack the capability for simultaneous listening and engagement, which diminishes the communal aspect of music enjoyment. Lobic's virtual "listening rooms" will provide users with a space to listen together, discuss their favorite tracks, and share their musical discoveries in real time, fostering a sense of connection that current platforms do not offer.

What sets Lobic apart from existing works is its unique focus on combining social interaction with music streaming. By leveraging features like live chat and collaborative playlists, Lobic creates an immersive environment where users can experience music as a shared journey. Furthermore, the application will utilize advanced technologies to provide personalized recommendations and responsive interfaces, enhancing user engagement. Ultimately, Lobic aims to contribute to the digital music landscape by establishing a platform that elevates music from a solitary activity to a vibrant, community-oriented experience, enriching both social connections and music appreciation.

1.4. Expected outcomes

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

Fully Functional Music Streaming Platform: A robust web application where users can seamlessly search for, stream, and manage music playlists, ensuring a user-friendly experience that meets modern expectations for music consumption.

Real-Time Social Interaction Features: The implementation of virtual "listening rooms" that enable users to listen to music together while engaging in live chat, fostering a sense of community and enhancing the shared music experience.

Personalized Music Recommendations: Development of a recommendation engine that leverages user data to provide tailored music suggestions, improving user engagement and satisfaction through a more personalized listening experience.

Scalable and Secure Architecture: A scalable backend system utilizing Rust and Axum, along with a responsive frontend developed in Node.js and React, ensuring the platform can handle increasing user demand 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

In exploring the landscape of music streaming platforms, it is essential to examine existing projects and their features to identify gaps that Lobic seeks to address.

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.

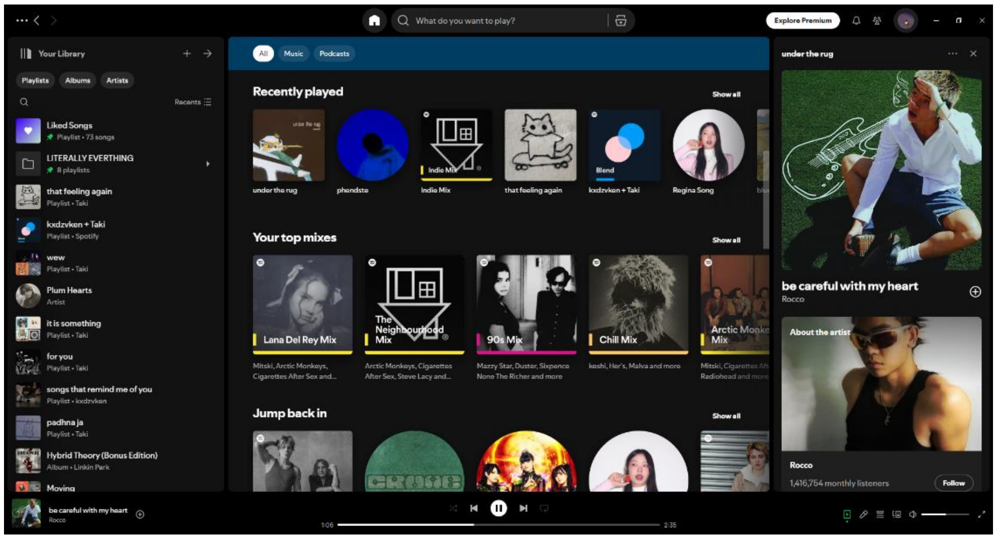

Figure 1: Spotify Dashboard UI

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 Lobic intends to. By focusing on fostering a shared music experience through features like live chat and communal playlists, Lobic aims to create a more interactive environment for music lovers, setting itself apart from existing platforms like YouTube Music.

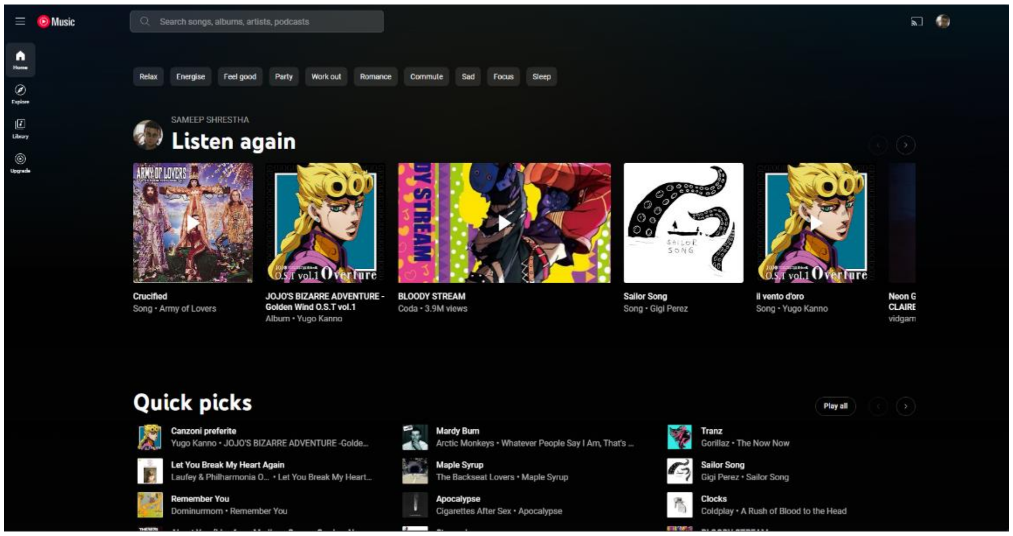

Figure 2: Youtube Music Dashboard UI

Chapter 3 Procedures and Methods

The development of the Lobic project will adhere to a structured approach, encompassing several key phases designed to ensure the creation of a robust and user-friendly music streaming application. We will utilize modern development methodologies, including Agile practices, to promote flexibility and iterative improvement throughout the development process.

3.1 Development Phases

i. Requirement Analysis:

The initial phase involves gathering and analyzing both functional and non-functional requirements for the application. This process will include user feedback collection and research on existing platforms to inform the design and feature set of Lobic. Identifying user needs and preferences will be crucial for shaping the overall direction of the project.

ii. System Design:

In this phase, we will develop wireframes and mockups for the user interface (UI) while defining the application's architecture. The primary focus will be on creating an intuitive UI that enhances user interaction and engagement. This design will ensure that users can easily navigate the application and access its features.

iii. Implementation:

The implementation phase will involve the actual coding of the application using the selected technology stack:

- **Frontend:** The frontend will be developed using React and Node.js to create a dynamic and responsive user interface, enabling seamless user experiences across devices.

- **Backend:** The backend will be built with Rust and Axum, focusing on performance and security to effectively manage user data and real-time interactions.
- **Database:** SQLite will be employed for efficient data storage and retrieval, supporting critical functionalities such as playlists and user profiles.

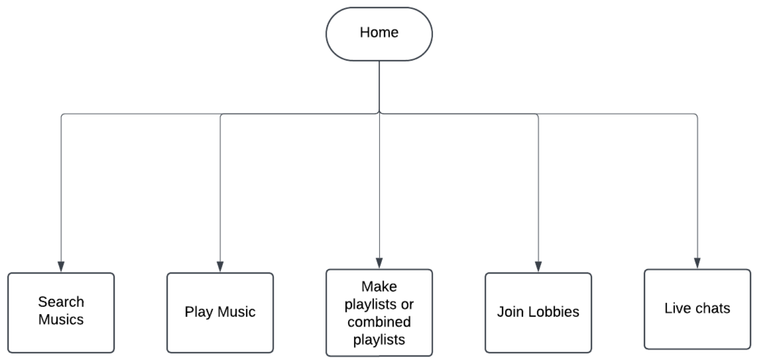

Figure 3: Project Implementation

iv. Testing:

A comprehensive testing strategy will be implemented to ensure functionality, performance, and security of the application. This strategy will encompass unit testing, integration testing, and user acceptance testing, allowing us to gather valuable feedback for further improvements. Regular testing iterations will help 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 will be maintained to capture the development process, architectural decisions, and user guides. This documentation will serve as a critical resource for future developers and stakeholders, providing insights into the system's design and functionality. A final report will be compiled at the project's conclusion, summarizing the objectives, methodologies, and outcomes.

Chapter 4 System Requirement Specifications

4.1. Software Specifications

i. NodeJS (22.7.0) as a frontend language
ii. Rust (1.81.0) as a server-side language
iii. React (18.3.1) for Web Application
iv. Axum (0.7.7) for Web server
v. SQLite for database
vi. VS Code as code editor
vii. Operating System: Windows 10 or Linux Based OS

4.2. Hardware Specifications

i. Processor: Intel Core i3 or equivalent
ii. RAM: 4 GB
iii. Storage: 128 GB HDD/SSD

Chapter 5 Project Planning and Scheduling

5.1. Tasks

The Gantt chart provided in Figure 4 outlines the timeline for various tasks involved in project development, giving a clear picture of time allocation across different phases. This chart allows for structured planning by visually mapping out each task's start and end dates, contributing to an organized project schedule.

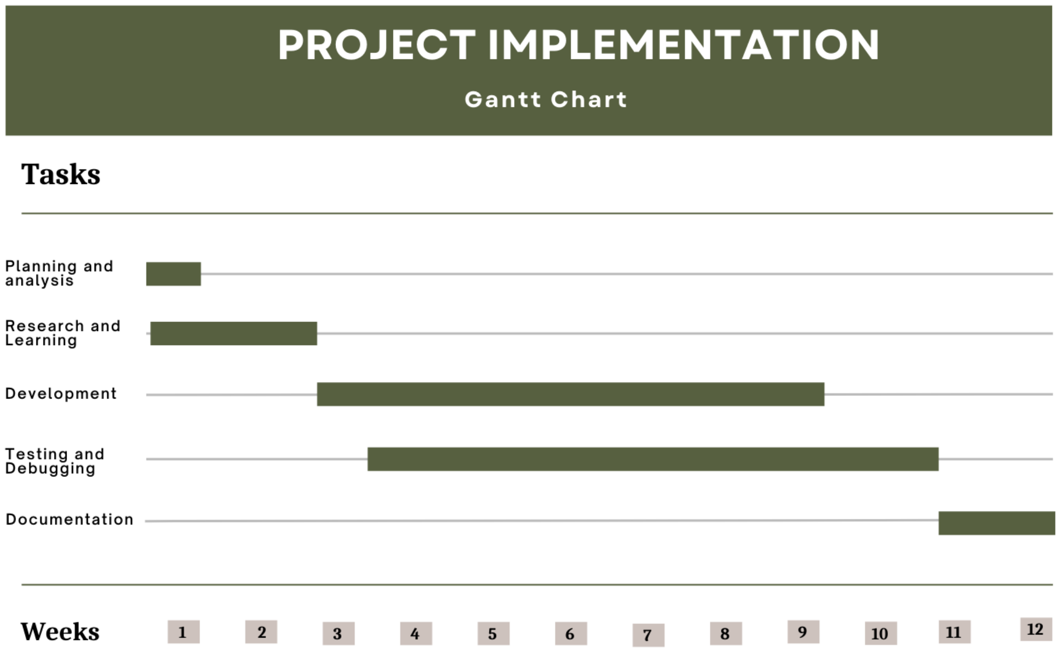

Figure 4: Gantt Chart

Tasks and Phases

i. **Planning and Analysis:** Initial stage focused on problem identification, requirement gathering, and setting project objectives. This phase is scheduled for the first week to ensure a clear project direction.

ii. **Research and Learning:** Occurs early in the project, spanning weeks 2 to 3, to gather necessary insights and knowledge related to the project scope.

iii. **Development:** The core phase of the project, stretching from week 4 to week 8, where the actual work on the project deliverables takes place.

iv. **Testing and Debugging:** Starts in week 5 and continue alongside development, extending to week 11 to identify and resolve any issues in the project.

v. **Documentation:** Final stage, occurring in week 12, to compile all findings, processes, and outcomes, ensuring a complete and accurate project record.

The Gantt chart structure provides a roadmap that helps monitor project progress and allocate resources efficiently. With a breakdown of terminal and summary elements, this visual tool aids in meeting project milestones and managing time effectively.

References

Amin, S., & Kaur, P. (2024). *Exploring Node.js for web development*. Node.js Documentation. https://nodejs.org/en/docs/guides/

Baker, S., & Lewis, T. (2024). *Building dynamic user interfaces with React*. React Documentation. https://reactjs.org/docs/getting-started.html

Kaur, P., & Amin, S. (2024). *Getting started with Rust programming*. Rust Documentation. https://www.rust-lang.org/learn

SQLite Consortium. (2024). *SQLite documentation*. SQLite. https://www.sqlite.org/docs.html

Axum. (2024). *Axum: An ergonomic web framework*. Axum Documentation. https://docs.rs/axum/latest/axum/

Spotify. (2024). *About Spotify*. https://www.spotify.com

YouTube Music. (2024) *Discover music and playlists on YouTube Music*. https://music.youtube.com