**MINOR-1 PROJECT**

**SYNOPSIS on**

***Development of a simplified music streaming web application***

Submitted By:

|  |  |  |
| --- | --- | --- |
| **Name** | **Roll No** | **Branch** |
| Soumil Jain | R2142220943 | CS(AIML) |
| Sannidhya Sharma | R2142220890 | CS(AIML) |
| Lakshya Agrawal | R2142220338 | CS(AIML) |

**Under the guidance of**

Dr. Virender Kadyan



**UNIVERSITY OF PETROLEUM AND ENERGY STUDIES**

**Dehradun-248007 2023**

**Approved By**

**Project Guide Cluster Head**

**School of Computer Science**

University of Petroleum & Energy Studies, Dehradun

Development of a simplified music streaming web application

**1. Abstract**

This paper presents the design and development of a simplified music streaming web application, focusing on core functionalities such as user authentication, music streaming, playlist management, and music search. The application provides users with an intuitive interface for browsing songs, creating and managing playlists, and streaming music in real-time. The frontend of the application is developed using HTML, CSS, and JavaScript, ensuring responsiveness and ease of use. The backend is implemented in Java, with JavaScript utilized where necessary to manage specific features and functionality. The MySQL database is used for efficient storage and retrieval of user data, song information, and playlists. Users can log in, create personalized playlists, and manage their music collections with real-time playback controls powered by JavaScript. The backend, built in Java, handles communication between the frontend and the database, ensuring smooth and scalable operation for managing data requests and streaming services. This application demonstrates an effective solution for building a basic music streaming platform and offers a solid foundation for future feature expansion, including music recommendations and social sharing capabilities.

**2. Introduction**

Music streaming has transformed the way people engage with music, offering instant access to vast libraries across various platforms. The development of such platforms requires a deep understanding of both frontend and backend technologies, as well as database management to ensure seamless user experiences. This project focuses on creating a simplified music streaming web application, providing insights into the integration of essential technologies to offer a core music streaming experience. The project involves designing and implementing a basic music streaming web application where users can log in, stream music, create and manage playlists, and search for songs. The core functionality includes user authentication, playlist management, and real-time music playback. The application also integrates with a relational database, using MySQL, to manage user profiles, song metadata, and playlists. While the project focuses on core functionality, it does not extend to advanced features like music recommendations or social sharing.

Developing this music streaming application is a valuable exercise in understanding how media content can be served efficiently over the web. This project demonstrates the technical requirements and challenges involved in managing real-time music streaming, as well as the integration of user-centric features like playlist creation and authentication. It serves as a foundational project for those interested in building more comprehensive media applications, emphasizing the importance of scalability, database management, and user experience. The frontend of the application is built using **HTML, CSS,** and **JavaScript** to create an interactive, responsive interface. On the backend, Java is used to manage key operations such as handling requests, processing user actions, and managing playlist data. For the database, **SQLite** or **MySQL** is used, providing an embedded, serverless solution for managing user data and playlists. SQLite is lightweight, fast, and easy to integrate, making it suitable for small-scale projects or for initial prototyping. In future expansions, MySQL or other database systems can be integrated for scalability. JavaScript is also employed for handling audio playback and enhancing real-time interactivity. The music streaming project can be enhanced by incorporating features like responsive design, audio controls, playlist visualization, and search functionality in the frontend. On the backend, error handling, caching, security measures, and scalability should be considered. The database can be optimized through indexing, normalization, and data validation. Additional features like music recommendations, social features, offline playback, and integration with third-party services can be explored.

Testing and deployment processes, including unit testing, integration testing, and deployment to a suitable environment, should be implemented. By addressing these aspects, the project can provide a more comprehensive and user-friendly music streaming experience. The target audience for this project includes web developers and students looking to gain hands-on experience with building media applications. It also caters to those interested in full-stack development, offering insights into how frontend and backend technologies collaborate with databases to deliver a seamless user experience.

This project provides a foundation for developers looking to build more advanced media streaming platforms with additional features in the future. Key challenges in developing the application include synchronizing the backend processes with frontend interactions, especially for real-time music streaming. The use of SQLite presents scalability limitations, as it is primarily designed for smaller projects and may not handle large datasets or concurrent users as efficiently as MySQL. Furthermore, the absence of advanced features such as cloud storage or personalized recommendations limits the application’s scope, making it suitable primarily as a foundational learning tool for future extensions.

*This project offers an introduction to building a simplified music streaming platform, demonstrating the practical application of frontend, backend, and database technologies in an integrated web environment.*

**3. Literature Review**

Music streaming has become a dominant mode of music consumption, transforming the music industry and user behaviours. This shift has prompted extensive research into various aspects of music streaming, including technological advancements, user experience, and system architecture. This literature review explores existing research and technologies related to music streaming web applications, focusing on frontend and backend development, database management, and real-time streaming capabilities. The evolution of music streaming began in the late 1990s with the advent of early digital music services and file-sharing platforms. Platforms like Napster and iTunes laid the groundwork for legal digital music distribution. The early 2000s saw the rise of streaming services like Pandora and Spotify, which revolutionized the industry by offering subscription-based access to extensive music libraries. These advancements highlighted the need for robust technologies and methodologies to support scalable and user-friendly streaming services.Recent research in music streaming focuses on optimizing streaming quality, reducing latency, and enhancing user experience. Studies have explored various streaming protocols, such as HTTP Live Streaming (HLS) and Dynamic Adaptive Streaming over HTTP (DASH), which adapt video and audio quality based on network conditions. Research also emphasizes the integration of machine learning algorithms for personalized recommendations and the use of cloud storage solutions for scalability. Key areas of interest include improving streaming efficiency, minimizing buffering, and developing advanced user interfaces.

The frontend of a music streaming web application is the user's primary interface. It must be intuitive, visually appealing, and responsive to ensure a seamless listening experience. Research has explored various frontend technologies and design principles to achieve these goals.

* **JavaScript Frameworks:** Frameworks like React and Angular have become popular choices for building dynamic and efficient user interfaces. They offer component-based architecture, state management, and routing capabilities, simplifying the development process.
* **CSS Preprocessors:** Sass and Less are widely used to enhance CSS styling and maintainability. They provide features like variables, nesting, and mixins, making it easier to create consistent and scalable styles.
* **Responsive Design:** Ensuring a great user experience across different devices (desktop, tablet, mobile) is crucial. Responsive design principles and frameworks like Bootstrap help adapt the layout and content to various screen sizes.
* **User Interface Design:** Research has focused on understanding user preferences and behaviors to design intuitive and engaging interfaces. Factors such as navigation, layout, and visual elements play a significant role in user satisfaction.

The backend of a music streaming web application handles the server-side logic, data storage, and streaming protocols. Efficient backend architecture is essential for scalability, reliability, and performance.

* **Server-Side Languages:** Python, Ruby, Node.js, and Java are commonly used for backend development. Each language has its strengths and weaknesses, and the choice often depends on the team's expertise and project requirements.
* **Database Management:** Relational databases like MySQL and PostgreSQL are suitable for storing metadata (e.g., song information, artist details) and user data. NoSQL databases like MongoDB and Cassandra can be used for storing large amounts of unstructured data (e.g., audio files).
* **Streaming Protocols:** HTTP Live Streaming (HLS) and Dynamic Adaptive Streaming over HTTP (DASH) are popular protocols for delivering audio and video content to users. They adapt the quality of the stream based on network conditions, ensuring a smooth playback experience.
* **Cloud Infrastructure:** Cloud platforms like AWS, Google Cloud Platform, and Microsoft Azure provide scalable infrastructure and services for music streaming applications. They offer features like content delivery networks (CDNs), load balancing, and serverless computing.
* **Micro-services Architecture:** Breaking down the application into smaller, independent services can improve scalability, maintainability, and resilience. Micro-services architecture allows for flexible deployment and updates.

Real-time streaming is a critical aspect of music streaming applications, ensuring synchronized playback across multiple devices.

* **WebSockets:** WebSockets provide a full-duplex communication channel between the client and server, enabling real-time updates and synchronization.
* **Synchronization Algorithms:** Research has explored various algorithms to maintain synchronization between multiple clients. Techniques like clock synchronization and timestamping are commonly used.
* **Latency Management:** Minimizing latency is essential for a smooth listening experience. Techniques like caching, optimizing network paths, and using low-latency servers can help reduce latency.

Future Trends and Challenges

* **AI-Powered Music Generation:** AI algorithms can be used to create new music compositions, potentially revolutionizing the music industry.
* **Immersive Audio Experiences:** Technologies like spatial audio and 3D sound can provide a more immersive listening experience.
* **Social Features and Community Building:** Integrating social features like playlists sharing and community forums can foster a sense of belonging among users.
* **Ethical Considerations:** Issues such as copyright infringement, artist compensation, and data privacy must be addressed to ensure a sustainable and ethical music streaming ecosystem.

**4. Problem Statement**

In today’s digital age, music streaming platforms have become an integral part of our daily lives, offering excess of features. However, many music streaming platforms today are packed with advanced features like machine learning and cloud storage, which can be too complicated for users who just want a simple and efficient music streaming experience.

This project aims to create a **basic but scalable music streaming web app** using **HTML, CSS, JavaScript, Java, and MySQL**. This will focus on essential features like real-time streaming, user login, playlist management, and database handling.

**5. Objectives**

The aim of this project is to create a basic music streaming web application that allows users to listen to music, manage playlists, and play songs in real-time through an easy-to-use interface. It focuses on providing the essential features of a music streaming platform without the complexity of advanced tools like machine learning or cloud storage. The project will use HTML, CSS, and JavaScript for the frontend, Java for the backend, and SQLite/MySQL for managing user and music data, offering a strong base for future improvements and scaling.

**6. Methodology**

***Reference Software Model***

***Model:*** *MVC (Model-View-Controller) Architecture*

1. Model: Handles data and business logic (e.g., user profiles, playlists, music data).
2. View: Manages user interface and user experience (e.g., music player, search interface).
3. Controller: Manages communication between the model and view (e.g., processing user requests, updating the view).

***Steps***

1. ***Requirement Analysis***

* Define features and user stories.
* Analyze existing solutions.

1. ***Design***

* Create UI/UX designs.
* Design system architecture and database schema.

1. ***Technology Stack Selection***

* Backend: Java Spring Boot.
* Database: PostgreSQL.
* Frontend: HTML CCS.
  1. **Schedule**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **TASK** | **AUG 2024** | **SEP 2024** | **OCT 2024** | **NOV 2024** |
| Project Planning |  |  |  |  |
| Research and Design |  |  |  |  |
| Software Development |  |  |  |  |
| Integration and Testing |  |  |  |  |
| Documentation |  |  |  |  |
| Review |  |  |  |  |

* 1. **References**

1. Spring Framework Documentation. (n.d.). Spring. [Enhancing the Assessment and Feedback in Higher Education (2023) –](https://spring.io/projects/spring-framework)
2. PostgreSQL: The world's most advanced open source database. (n.d.). [Feedback Literacy in Higher Education (2022) – Link](https://www.postgresql.org)
3. Dey, A., & Houghton, R. (2019). The Role of Real-Time Feedback Systems in Enhancing Educational Quality. Educational Technology Research and Development, 67(3), 525-542.
4. Ricardo A. Baeza-Yates and Chris H. Perleberg. Fast and Practical Approximate String Matching. In Combinatorial Pattern Matching, Third Annual Symposium, pages 185–192, 1992.
5. Jean-julien Aucouturier and Francois Pachet. Music Similarity Measures: What is the Use. In Proceedings of the ISMIR, pages 157–163, 2002.
6. R. L. Rosa, D. Z. Rodríguez, and G. Bressan, “SentiMeter-Br: a new social web analysis metric to discover consumers' sentiment,” in Proc. IEEE International Symposium on Consumer Electronics, Hsinchu, Taiwan, pp. 153-154, Jun. 2013
7. R. Cai, C. Zhang, C. Wang, L. Zhang, and W.-Y. Ma, “MusicSense: contextual music recommendation using emotional allocation modeling,” in Proc. International Conference on Multimedia, Augsburg, Germany, pp. 553-556, Sep. 2007.