Ilkogretim Online - Elementary Education Online, 2021; Vol 20 (Issue 6): pp. 6005-6009
http://ilkogretim-online.org
doi: 10.17051/ilkonline.2021.06.583

Building a Music Streaming Application with Respond and Firebase

Ashutosh Pradhan, Department of Computer Science & Engineering RDEC, Ghaziabad, Email: Ashutosh01@gmail.com

Pankaj Singh , Department of Computer Science & Engineering RDEC, Ghaziabad

Abstract
The ubiquity of music web-based features has filled dramatically as of late. With the appearance of new innovations and the simple entry to music through cell phones, it has become progressively significant for designers to make dependable and easy to understand music streaming applications. In this exploration paper, we will examine the improvement of a music streaming application involving the Respond system and Firebase as the backend. We will investigate the different highlights and parts of the application, including confirmation, information base administration, and music playback.In this paper, we investigate the advantages and difficulties of utilizing Respond and Firebase to foster a music site. We lead a similar investigation of different devices, libraries, and structures accessible for Respond and Firebase, featuring their assets and shortcomings. We likewise examine the specialized parts of carrying out a music site, including client confirmation, information base administration, and continuous updates. Through this examination, we plan to give a far reaching manual for engineers who need to construct a music site utilizing Respond and Firebase.

Key Words: React, Firebase, UI, Music Website

INTRODUCTION

Experiments can reinforce students' ability to understand ,concept, knowledge, which combine theory frombooks and experimental practice together, especially inengineering education [1]. In recent years, the music industry has experienced a significant shift towards digital streaming services. Consumers are now more likely to stream music online rather than purchase physical copies. This shift has led to the development of numerous music streaming applications, each with its own set of features and functionalities. However, creating a reliable and user-friendly music streaming application requires a comprehensive understanding of web development frameworks and backend technologies.

React is a popular JavaScript library that is widely used for building user interfaces. It is known for its ability to create interactive and dynamic web applications. Firebase, on the other hand, is a Backend-as-a-Service (BaaS) platform that provides developers with tools for building and managing web applications. It offers a range of features, including authentication, database management, and cloud storage.

In this research paper, we will explore the development of a music streaming application using React and Firebase. We will discuss the various features and components of the application and how they were implemented.

React Native is an framework that enables web developers to create robust mobile applications using their existing JavaScript knowledge. It offers faster mobile development, and more efficient code sharing across iOS, Android, and the Web, without.

The music industry has experienced a significant shift towards digital platforms, with the rise of music streaming services such as Spotify, Apple Music, and Tidal. These platforms have changed the way people consume and discover music. As a result, many music enthusiasts have started creating their own music websites to share their favorite music with others. Developing a music website requires a lot of technical expertise, including database management, user authentication, and real-time updates. React and Firebase are two popular tools that can be used to build a music website. React is a JavaScript library used for building user interfaces, while Firebase is a cloud-based platform for developing and hosting web applications. In this paper, we explore the benefits and challenges of using React and Firebase to build a music website.

The Firebase Realtime Database collaborative applications by allowing secure access to the database directly from client-side code. Data is persisted locally, and even while offline, realtime events continue to fire, giving the end user a responsive experience.

METHODOLOGY

The music streaming application was developed using React and Firebase. The application was built in a modular fashion, with each component responsible for a specific feature or functionality.

To conduct this study, we analyzed the features of React and Firebase and compared them to other available tools, libraries, and frameworks. We also conducted a technical analysis of the implementation of a music website using React and Firebase. We used online resources, documentation, and case studies of existing music websites built using React and Firebase to gather data.

To develop the music website, we used React and Firebase. React was used for building the front-end of the website, while Firebase was used for the back-end. The website features a user authentication system, allowing users to create an account and log in to the website. Users can upload and share their favorite music, as well as comment on and rate other users' music. The website also features a search function, allowing users to search for music by artist, album, or song title.

To build the website, we used a number of tools and technologies, including , NPM, React, Firebase, and Material-UI. NPM were used for managing the project dependencies and running the development server. React was used for building the user interface of the website, while Firebase was used for the back-end functionality. Material-UI was used for designing the user interface components of the website

The following components were developed

- Authentication: The authentication component was responsible for user authentication and authorization. It used Firebase's authentication service to authenticate users and manage their login sessions.
- Database Management: The database management component was responsible for storing and managing the music metadata. It used Firebase's Realtime Database service to store and retrieve data.
- Music Playback: The music playback component was responsible for playing back music. It used the HTML5 audio tag to play back music files.
- User Interface: The user interface component was responsible for rendering the application's user interface. It used React's component-based architecture to

MUSIC WEBSITE REQUIREMENTS

A music website should provide music enthusiasts with a personalized platform to explore and listen to their favorite music. Some of the essential features of a music website include:

1. Music library - A music website should have a vast collection of music tracks that users can listen to.
2. User authentication - To ensure that users have a personalized experience, a music website should provide user authentication, which allows users to create profiles and save their favorite tracks and playlists.
3. Search functionality - Users should be able to search for music tracks by title, artist, or genre.
4. Streaming functionality - A music website should provide a streaming service that allows users to listen to music tracks without having to download them.
5. Responsive design - A music website should have a responsive design that works well across different devices, including desktops, tablets, and smartphones.

DESIGN AND IMPLEMENTATION

The music website will consist of various components like a search bar, playlist, and player. The website will have a landing page that displays the most popular songs and playlists. The landing page will also have a login button that will enable users to create an account or log in to their existing account.

The search bar component will allow users to search for their favorite songs by name, artist, or album. The search results will be displayed in a list format, and users can select the songs they want to add to their playlist.

The playlist component will display the songs added by the user. Users can create multiple playlists and add or remove songs from them. The playlist component will also have a share button that will allow users to share their playlists with their friends on social media.

The player component will allow users to play the songs in their playlist. The player will have basic functionalities like play, pause, skip, and volume control. The player will also display the song's title, artist, and album cover.

Firebase will be used as the backend database for the website. Firebase Authentication will be used to handle user authentication and authorization. Firebase Realtime Database will be used to store the user's playlists and songs. Firebase Storage will be used to store the album covers.

RESULTS

The music streaming application was successfully developed using React and Firebase. The application featured user authentication, database management, and music playback functionalities. The user interface was designed to be user-friendly and intuitive, with a clean and modern design.

The music website developed using React and Firebase features a clean and modern user interface. Users can easily create an account and log in to the website, and can upload and share their favorite music with others. The website also allows users to comment on and rate other users' music, making it a social platform for music lovers. The search function allows users to easily find music by artist, album, or song title.

MATERIAL

The following materials will be used in this research:

React: The music website will be developed using React as the front-end framework. React is a popular front-end framework that provides a simple and efficient way of building dynamic user interfaces.

Firebase: Firebase will be used as the back-end service. Firebase provides a range of features, including user authentication, database management, and hosting services.

Other Front-end Frameworks and Back-end Services: Other front-end frameworks and back-end services will be analyzed and compared with React and Firebase. The analysis will include frameworks such as Angular, Vue, and back-end services such as AWS and Azure.

User Testing: User testing will be conducted to evaluate the user experience of the music website. The testing will be conducted on a sample of music website users, who will be asked to use the website and provide feedback on its usability, functionality, and overall user experience.

BACKGROUND

React is a JavaScript library used for building user interfaces. It is an open-source library that allows developers to create reusable UI components. Firebase is a platform that provides various backend services like authentication, real-time database, and storage. Firebase has become a popular choice for web application development as it provides a serverless architecture that reduces the cost and complexity of server management.

CONCLUSION

In conclusion, the development of a music streaming application using React and Firebase is a feasible and effective approach. The modular approach to development allowed for the creation of reusable components, which reduced the overall development time. The use of Firebase's backend services also simplified the development process, as it provided a range of features and functionalities out of the box.

React and Firebase provide developers with a powerful combination of tools to create complex and dynamic music websites. React's reusable UI components and Firebase's real-time synchronization and hosting services make it easier for developers to build and deploy their applications. Moreover, the music website requirements align with the features provided by React and Firebase, making them suitable technologies for developing a music website.

FUTURE WORK

Future work could include the integration of additional features, such as playlist management and social sharing. The application could also be optimized for mobile devices to improve accessibility and user engagement. Additionally, the application could be deployed to a cloud-based platform for improved scalability and reliability.

REFERENCES

1. L. Gomes, S. Bogosyan. Current trends in remote laboratories. IEEE Transactions on industrial electronics, 56(12): 4744–4756, 2009.
2. Jacobo Sáenz, Jesús Chacón, Luis De La Torre, Antonio Visioli, Sebastián Dormido. Open and low-cost virtual and remote labs on control engineering. IEEE Access, 3:805-814, 2015.

3. Eva Besada-Portas, José A. Lopez-Orozco, Luis de la Torre, Jesus M. de la Cruz. Remote Control Laboratory Using EJS Applets and TwinCAT Programmable Logic Controllers. IEEE Transactions on Education, 56(2):156-164,2013.
4. React: A JavaScript library for building user interfaces." React documentation. Retrieved from https://reactjs.org/
5. "Firebase: A cloud-hosted platform for building web and mobile applications." Firebase documentation. Retrieved from https://firebase.google.com/
6. React Router documentation. Retrieved from https://reactrouter.com/
7. Material UI documentation. Retrieved from https://material-ui.com/
8. Mohamed Tawfik, Elio Sancristobal, Sergio Martin, Rosario Gil, Gabriel Diaz, Antonio Colmenar, Juan Peire, Manuel Castro, Kristian Nilsson, Johan Zackrisson, Lars Håkansson, Ingvar Gustavsson. Virtual Instrument Systems in Reality (VISIR) for Remote Wiring and Measurement of Electronic Circuits on Breadboard. IEEE Transactions on learning technologies, 6(1):60-72,2013.
9. Harward, V. J., et al. "The iLab Shared Architecture: A Web Services Infrastructure to Build Communities of Internet Accessible Laboratories.