HBRP PUBLICATION

Research and Reviews: Advancement in Robotics
Volume 6 Issue 2

Feelchords – An Emotion Based Music Player

Arulkumaran G.[1]*, Abdul Azeem[2], Darshan N.[3], Dhruva Doddamani[4], Jens Joy[5]

[1] Associate Professor,

[2,3,4,5]Student, School of C & IT, REVA University, Bengaluru, India

Corresponding Author

E-Mail Id: erarulkumaran@gmail.com

ABSTRACT

The proposed music player system leverages the power of React, a popular JavaScript library for building user interfaces, to create a dynamic and responsive music player. The user interacts with the application through a chatbot interface, providing input in the form of text messages or voice commands. This input is then processed using a sentiment analysis algorithm, which analyzes the emotional sentiment behind the user's input. Based on the sentiment analysis results, the music player suggests personalized music recommendations aligned with the user's emotions. Positive emotions, such as happiness or excitement, trigger suggestions of uplifting and energetic tracks, while negative emotions, such as sadness or anger, prompt the system to propose soothing or calming melodies. To ensure a seamless experience, the system incorporates an auto play-pause functionality that adjusts the music playback based on the user's preferences or detected emotions. The music player facilitates audio playback, providing the selected tracks to the user. By merging sentiment analysis, intelligent music suggestion, and chatbot capabilities, this React-based music player creates an immersive and personalized environment for users, empowering them to explore and enjoy music that resonates with their emotions.

Keywords: *React, sentiment analysis, emotion-based music player, chatbot, personalized music recommendation, auto play-pause functionality, and natural language processing*

INTRODUCTION

Music possesses a remarkable ability to communicate across cultures and evoke deep emotional responses. It can transport us to different realms, influence our mood, and connect us to our innermost feelings. Recognizing the significance of emotions in music, we have developed a revolutionary music player that revolves around the user's emotional state. This innovative approach enables individuals to curate playlists specifically tailored to their current emotional needs.

Recent surveys indicate that the average person dedicates approximately 900 hours per year, or roughly two hours a day, to listening to music. It is astounding that music holds such a pervasive influence, as it resonates with 7.1 out of the world's 7.9 billion population. Its therapeutic effects are truly remarkable, alleviating pain, reducing anxiety, lowering blood pressure, enhancing sleep quality, boosting memory, and providing numerous other benefits.

Clearly, music serves as an escape, offering moments of respite.

Emotions exert a profound impact on our thoughts and behaviors, with three primary components: the subjective experience of emotions, the physiological responses they elicit in our bodies, and the expressive behaviors they elicit. People communicate their emotions through various means, including speech, facial expressions, and body language. Music holds a potent and intimate connection with our emotions. Through its melodies, rhythms, and harmonies, it evokes powerful emotional responses within us. For instance, a slow, melancholic song can evoke feelings of introspection and sadness, while a fast, upbeat tune can invigorate us with energy and happiness.

Our music player incorporates the React Sentiment package, a powerful tool for analyzing text to discern emotional tones. By integrating this package into our player, we empower users to embark on a personalized listening experience tailored to their specific emotional requirements. Whether one is feeling joyful, despondent, or somewhere in between, our music player ensures an ideal soundtrack for their emotional journey. So, lean back, unwind, and allow the enchanting power of music to carry you through an unparalleled emotional odyssey.

Extensive research has revealed that music directly influences our emotional state. Listening to cheerful, uplifting music has the potential to uplift our spirits, infusing us with positivity and optimism. Conversely, melancholic, sorrowful melodies can deepen our sadness and desolation. Music possesses the capacity to elicit a wide spectrum of emotional responses, directly impacting our mood and emotional well-being. While music players have evolved over time, with features like fast forward, pause, shuffle, and repeat, none have addressed the user's emotional disposition. Therefore, our project endeavors to capture and assess the emotions of users through voice commands, providing a unique and personalized music experience.

RELATED LITERATURE

There have been previous works on creating smart music players using various methodologies. The 'Mood Therapist: Emotion Based Music Player' takes input in two ways - real-time images and input speech [1]. It employs different techniques and classifiers to extract multiple features from the images and convert spoken sentences into text to determine the user's emotion. This implementation offers accurate results by considering both text and face emotion recognition. It also incorporates user feedback, user-specific playlists, and frequently played playlists. However, it lacks emphasis on music database management, which makes it less user-friendly over time.

Another implementation is the 'Mood Enhancing Music Player Based on Speech Emotion Recognition and Text Emotion Recognition' [2]. This music player combines speech and text emotion recognition. It converts speech to text and applies NLP and SVM algorithms to determine the underlying emotion. Similarly, for text input, it uses NLP and SVM algorithms to identify the emotion.

The system provides accurate results by considering both types of emotion recognition. It also offers a user-specific music database to enhance the user's experience. However, it may not be suitable for larger music databases.

For a web application-based approach, the 'Web Application for Emotion-based Music Player' offers speech processing using the Berlin emotional database [3]. The system extracts relevant features and applies pattern recognition or classifier methods to identify emotional states. Once the emotion is identified, the system automatically selects an appropriate song from the database stored. This implementation addresses external factors such as bad lighting or image resolution by providing an option for emoji input. However, its accuracy is compromised to maintain user-friendliness and innovation.

Overall, these different implementations offer unique approaches to emotion-based music players. They each have their merits and demerits, ranging from accuracy and user-specific databases to user-friendliness and limited datasets.

PROPOSED METHODOLOGY

Begin by creating a new React project using Create React App or a custom setup. Install the necessary dependencies, including React, React Router, and Redux for state management [4] [5] [6]. Use npm or Yarn for package management. Create a visually appealing UI using HTML, CSS, and SCSS. Use React components to build the different elements of the music player, including the chatbot interface, music controls, and song information display. Utilize CSS or SCSS to style and layout the components. Create a chatbot component using React, JSX, and JavaScript. Use state management with Redux to handle user input and chatbot responses. Utilize event handlers to capture user interactions and update the chatbot's state accordingly. Incorporate a sentiment analysis library or API, such as the Natural Language Toolkit (NLTK) or the Google Cloud Natural Language API, to analyze the user's input. Use JavaScript to extract the sentiment or emotion from the text input and map it to relevant musical attributes. Connect to a music database or utilize a music recommendation API, such as the Spotify API, to retrieve music suggestions based on the sentiment analysis results. Use JavaScript to fetch and process the data in JSON format and update the Redux store with the relevant song information. Utilize React components to render the suggested music on the UI. Use JSX to dynamically display the song titles, artists, and album covers based on the data retrieved from the API. Apply CSS or SCSS styling to ensure an appealing visual presentation.

Integrate an audio player component using JavaScript and HTML5 audio capabilities. Use React components to control the audio playback functionalities, such as play, pause, and skip. Utilize event handlers and state management to handle user interactions and update the audio player's state accordingly. Implement event handlers using JavaScript to capture user interactions, such as clicking on a suggested song or manually inputting play or pause commands. Update the Redux store and the UI based on user actions,

triggering the appropriate audio playback and chatbot responses.

Utilize Firebase for user authentication [7], allowing users to log in and save their preferences and playlists. Integrate Netlify for deployment [8], ensuring that the application is accessible online. Use the Firebase and Netlify APIs and corresponding JavaScript libraries to implement these functionalities. Thoroughly test the music player application, including the chatbot, sentiment analysis, music suggestion, and audio playback features. Write unit tests using React Testing Library or Jest to ensure code correctness. Once testing is complete, deploy the application to Netlify for hosting, ensuring a seamless user experience.

TECHNOLOGIES USED

React

React is a JavaScript library predominantly used to build user interfaces. It allows you to create reusable UI components and efficiently manage the state of your application. React uses a virtual DOM to efficiently update and render various components that results in fast and responsive user interfaces.

- React: The core React library for building user interfaces.
- react-dom: The package that provides the integration between React and the browser's DOM (Document Object Model).
- react-router-dom: A library for implementing routing in a React application, allowing for navigation between different views or pages.

JavaScript

JavaScript is a programming language that powers the interactivity and logic of web applications. In the context of an emotion-based music player, JavaScript is used to handle user interactions, perform calculations, manipulate data, and integrate with external libraries and APIs.

- axios: A popular HTTP client library used to make API requests.
- lodash: A utility library that provides helpful functions for manipulating arrays, objects, and other data structures.
- moment: A library used to parse, manipulate, and format dates and times.
- prop-types: A library used for defining and validating the types of React component props.

SCSS and CSS

SCSS (Sass) and CSS are styling languages used to customize the appearance of web applications. SCSS is a superset of CSS, providing additional features like variables, nesting, and mixins, which help write more maintainable and modular styles. CSS, on the other hand, is the standard styling language used to define the visual aspects of HTML elements.

- node-sass: A module that allows you to compile SCSS files into CSS during the build process.
- classnames: A utility library for conditionally joining CSS class names together.

HTML

HTML is the markup language used to structure the content of web pages. It provides the skeleton of the application by

defining the elements and their hierarchical structure. HTML is used to define the structure of the player interface, including buttons, inputs, and other UI elements.

JSON

JSON (JavaScript Object Notation) is a data interchange format extensively used to store and transmit data between a web app and a server. In the context of a music player, JSON can be used to store song metadata, sentiment mappings, and other structured data that can be easily accessed and manipulated in JavaScript.

Redux

Redux is a predictable state container for ReactJS applications. It provides a centralized store to manage the application's state, making it easier to share data across components and maintain a consistent state throughout the application. Redux uses actions and reducers to update and retrieve data from the store.

- redux: The core Redux library that provides state management capabilities.
- react-redux: A library that allows React components to connect to the Redux store and access its state.
- redux-thunk: A middleware for Redux that enables handling asynchronous actions and side effects.
- redux-logger: A middleware that logs Redux actions and state changes to the browser's console, aiding in debugging.

Firebase

Firebase is an extensive platform used to build web and mobile applications. It provides a range of backend services, including authentication, real-time database, hosting, and more. Firebase can be used for user authentication, storing user preferences and sentiment analysis results, and hosting the application.

- firebase: The core Firebase library for integrating Firebase services into a web application.
- firebase/auth: A Firebase module that handles user authentication.
- firebase/database: A Firebase module for interacting with the real-time database.
- firebase/storage: A Firebase module for storing and retrieving files in Firebase Cloud Storage.

Netlify

Netlify is a cloud hosting platform that simplifies the deployment and hosting of web applications. It offers features like continuous deployment, automatic HTTPS, and CDN (Content Delivery Network) caching. With Netlify, you can deploy your sentiment analysis emotion-based music player to a secure and scalable environment.

IMPLEMENTATION

Data Collection

The first step involves collecting a large dataset of music tracks, each tagged with the appropriate emotions that they convey. This dataset is used to train our ML model and provide personalized music recommendations.

Sentiment Analysis

The React Sentiment package is integrated into the music player, providing accurate and real-time sentiment analysis results for each user. The analysis is performed on the user's speech input, which is captured using

the Web Speech API. The user's current emotional state is determined.

Music Recommendation

The music player provides personalized recommendations for music tracks that match the user's current emotional state using the analysis results. These recommendations are generated using a machine learning model that has been trained on the music dataset collected in step A.

User Interface

The user interface of the music player is designed to be simple, intuitive and responsive. It provides a wide range of music genres, ensuring that users can find the perfect music to match their current emotional state. The interface also displays the sentiment analysis results and the recommended music tracks, making it easy for users to navigate and find the perfect music for their mood.

Audio Playback

The music player uses HTML5 audio playback to provide high-quality music playback to users. The audio playback is integrated with the sentiment analysis and music recommendation systems, ensuring that the music playback matches the user's current emotional state.

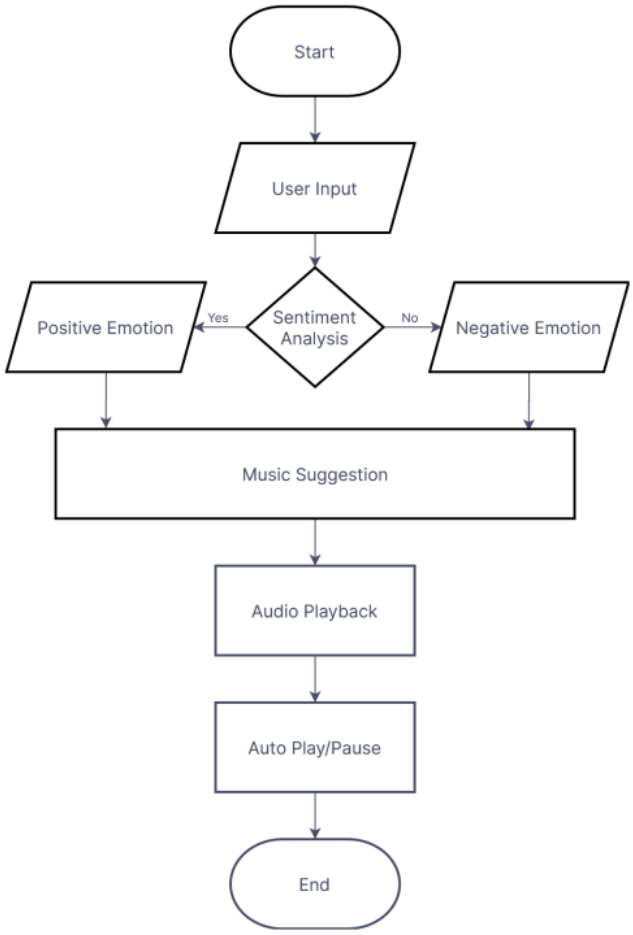

***Fig. 1**. Implementation Flowchart*

RESULTS

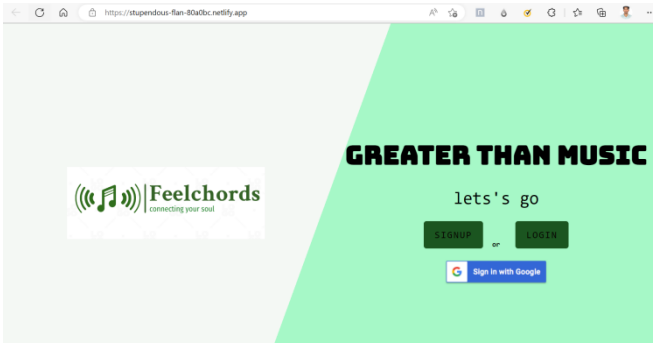

Fig. 2.1. *Main Page*

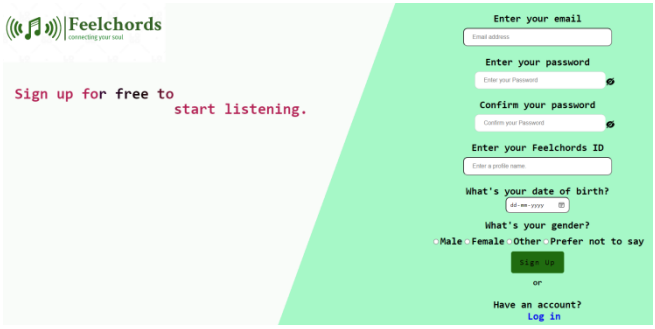

Fig. 2.2. *Signup Page*

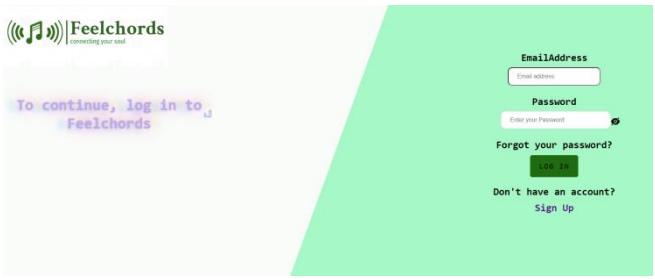

Fig. 2.3. *Login Page*

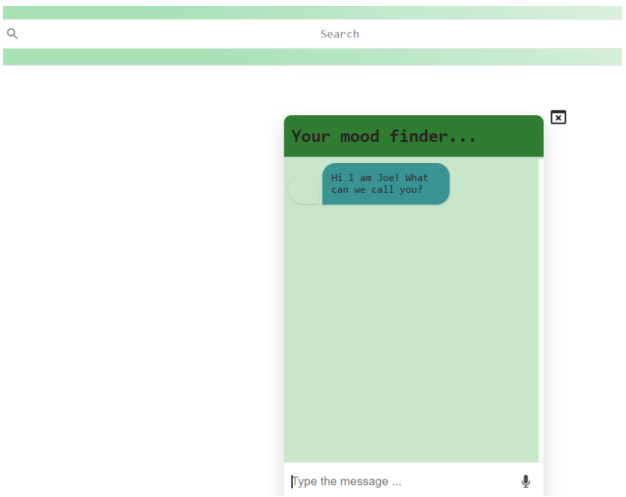

Fig. 2.4. *Home Page*

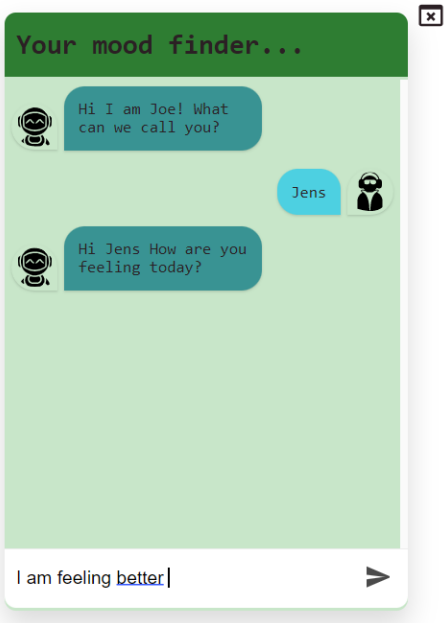

Fig. 2.5. *Chatbot interaction*

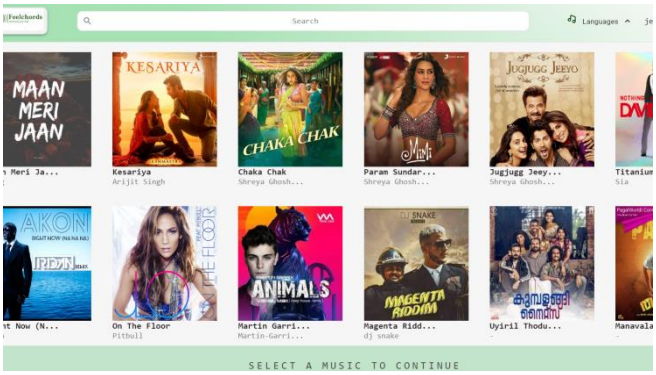

Fig. 2.6. *Recommendations*

CONCLUSION

We successfully developed and deployed an application that considers the user emotion and recommend songs to enhance their mood. Basic functions like pause/resume, changing track etc. through voice command in addition to manual buttons have been implemented successfully.

We developed a smooth, appealing, and appropriate user-friendly interface. Enhancing performance and providing constant updates to avoid security issues are to be tackled very often.

FUTURE SCOPE

Enhanced Sentiment Analysis

Continuously improve the sentiment analysis algorithms by incorporating machine learning and natural language processing techniques. Train the model on larger datasets to enhance accuracy and expand the range of emotions detected. Explore advanced sentiment analysis models like transformers or deep learning

architectures to achieve better sentiment analysis results.

Personalized Music Recommendations
Implement a personalized recommendation system based on user preferences, listening history, and sentiment analysis results. Utilize collaborative filtering, content-based filtering, or hybrid recommendation algorithms to suggest music tailored to individual users' tastes and emotions.

Integration with External Music APIs
Integrate with popular music streaming services like Spotify, Apple Music, or YouTube Music to fetch and play music directly from their catalogs. This integration can provide a wider range of music choices and enhance the user experience.

Social Sharing and Collaboration
Enable users to share their sentiment-based playlists or favorite songs on social media platforms. Implement collaborative features that allow users to create and share playlists with friends or discover music based on the sentiment of others in their network.

Integration with Music Lyrics
Display synchronized lyrics for the currently playing song, providing users with a more immersive music experience. Incorporate sentiment analysis of lyrics to enhance the understanding of the emotional content of the songs.

Mobile App Development
Extend the sentiment analysis emotion-based music player to mobile platforms by developing native mobile applications using React Native or other frameworks. This expansion would allow users to access the music player on their smartphones and tablets.

Data Analytics and Insights
Implement analytics capabilities to collect and analyze user behavior, sentiment patterns, and music preferences. Leverage the gathered insights to refine the music recommendation algorithms, improve user engagement, and enhance the overall user experience.

Integration with Smart Devices
Explore integration with smart devices and voice assistants like Amazon Echo or Google Home to enable users to control the sentiment-based music player through voice commands and enjoy a seamless experience across different devices.

REFERENCES

1. Kabani, H., Khan, S., Khan, O., & Tadvi, S. (2015). Emotion based music player. *International journal of engineering research and general science*, *3*(1), 2091-2730.
2. Deshmukh, S. M., & Sita, D. (2020). Mood enhancing music player based on speech emotion recognition and text emotion recognition. *International Journal*, *8*(6).
3. Soni, K. N., Agrawal, K., Pandya, N., & Agrawal, N. Web Application for Emotion-based Music Player.
4. React. Installation. https://react.dev/learn/installation
5. React Router. Quick Start. v5.reactrouter.com/core/guides/quick-start

6. Redux. Installation. https://redux.js.org/introduction/installation/
7. Firebase. Firebase Authentication. firebase.google.com/docs/auth
8. Eli Williamson, Jason Lengstorf. A Step-by-Step Guide: Deploying on Netlify. https://www.netlify.com/blog/2016/09/29/a-step-by-step-guide-deploying-on-netlify/