PROJECT REPORT

On

SOULSYNCOPIA: EMOTION-BASED MUSIC RECOMMENDATION SYSTEM

Submitted by

HARSH THAKKAR (IU2041230183)

JAYNIL ZALA (IU2041230200)

RIKIN ZALA (IU2041230201)

In the partial fulfillment for the award of the degree

Of

BACHELOR OF TECHNOLOGY

In

COMPUTER SCIENCE & ENGINEERING

INSTITUTE OF TECHNOLOGY AND ENGINEERING

INDUS UNIVERSITY CAMPUS, RANCHARDA, VIA-THALTEJ, AHMEDABAD – 382115, GUJARAT, INDIA,

WEB: www.indusuni.ac.in

APRIL, 2024

PROJECT REPORT

ON

SOULSYNCOPIA: EMOTION-BASED MUSIC RECOMMENDATION SYSTEM

AT

In the partial fulfillment of the requirement

for the degree of

Bachelor of Technology

in

Computer Science & Engineering

PREPARED BY

HARSH THAKKAR (IU2041230183)

JAYNIL ZALA (IU2041230200)

RIKIN ZALA (IU2041230201)

UNDER GUIDANCE OF

Internal Guide

Prof. Babita Patel,

Assistant Professor,

Department of Computer Science & Engineering,

I.I.T.E, Indus University, Ahmedabad

SUBMITTED TO

INSTITUTE OF TECHNOLOGY AND ENGINEERING

INDUS UNIVERSITY CAMPUS, RANCHARDA, VIA-THALTEJ,

AHMEDABAD-382115, GUJARAT, INDIA,

WEB: www.indusuni.ac.in

APRIL, 2024

CANDIDATE'S DECLARATION

We declare that the final semester report entitled "SoulSyncopia: Emotion-Based Music Recommendation System" is our own work conducted under the supervision of the guide Prof. Babita Patel.

We further declare that to the best of our knowledge, the report for B.Tech final semester does not contain part of the work which has been submitted for the award of B.Tech Degree either in this university or any other university without proper citation.

_____

Candidate's Signature

Harsh Thakkar – IU2041230183

_____

Candidate's Signature

Jaynil Zala – IU2041230200

_____

Candidate's Signature

Rikin Zala – IU2041230201

_____

Guide: **Prof. Babita Patel**

Assistant Professor,

Department of Computer Science & Engineering,

IITE, Indus University – Ahmedabad,

State: Gujarat

INDUS INSTITUTE OF TECHNOLOGY AND ENGINEERING

COMPUTER SCIENCE AND ENGINEERING

2023-2024

CERTIFICATE

Date: 19-04-2024

This is to certify that the project work entitled "**SOULSYNCOPIA: EMOTION BASED MUSIC RECOMMENDATION SYSTEM**" has been carried out by **HARSH JAGDISH THAKKAR (IU2041230183)** under my guidance in partial fulfillment of degree of Bachelor of Technology in **COMPUTER SCIENCE & ENGINEERING** (**Final Year**) of Indus University, Ahmedabad during the academic year 2023-2024.

PROF. BABITA PATEL Assistant Professor, Department of Computer Science & Engineering, I.I.T.E, Indus University, Ahmedabad	PROF. ZALAK VYAS Head of the Department, Department of Computer Science & Engineering, I.I.T.E, Indus University, Ahmedabad

INDUS INSTITUTE OF TECHNOLOGY AND ENGINEERING

COMPUTER SCIENCE AND ENGINEERING

2023-2024

CERTIFICATE

Date: 19-04-2024

This is to certify that the project work entitled "**SOULSYNCOPIA: EMOTION BASED MUSIC RECOMMENDATION SYSTEM**" has been carried out by **JAYNIL BHAVESHBHAI ZALA (IU2041230200)** under my guidance in partial fulfillment of degree of Bachelor of Technology in **COMPUTER SCIENCE & ENGINEERING** (**Final Year**) of Indus University, Ahmedabad during the academic year 2023-2024.

PROF. BABITA PATEL Assistant Professor, Department of Computer Science & Engineering, I.I.T.E, Indus University, Ahmedabad	PROF. ZALAK VYAS Head of the Department, Department of Computer Science & Engineering, I.I.T.E, Indus University, Ahmedabad

INDUS INSTITUTE OF TECHNOLOGY AND ENGINEERING

COMPUTER SCIENCE AND ENGINEERING

2023-2024

CERTIFICATE

Date: 19-04-2024

This is to certify that the project work entitled "**SOULSYNCOPIA: EMOTION BASED MUSIC RECOMMENDATION SYSTEM**" has been carried out by **RIKIN VISHALKUMAR ZALA (IU2041230201)** under my guidance in partial fulfillment of degree of Bachelor of Technology in **COMPUTER SCIENCE & ENGINEERING** (**Final Year**) of Indus University, Ahmedabad during the academic year 2023-2024.

PROF. BABITA PATEL Assistant Professor, Department of Computer Science & Engineering, I.I.T.E, Indus University, Ahmedabad	PROF. ZALAK VYAS Head of the Department, Department of Computer Science & Engineering, I.I.T.E, Indus University, Ahmedabad

ACKNOWLEDGEMENT

We would like to express our sincere gratitude to all those who have contributed to the successful completion of this college project on "SoulSyncopia: Emotion-Based Music Recommendation System." It has been a collaborative effort, and we are thankful for the support, guidance, and encouragement we received throughout this endeavour.

First and foremost, we extend our heartfelt appreciation to our college faculty and project supervisor **Prof. Babita Patel** [Assistant Professor in the Department of Computer Science & Engineering, IITE, Indus University] for their invaluable guidance and mentorship. Their expertise and unwavering support were instrumental in shaping the project and ensuring its successful execution.

Thank you all for being an integral part of this journey.

Sincerely,

-Harsh Thakkar

IU2041230183

Computer Science & Engineering

-Jaynil Zala

IU2041230200

Computer Science & Engineering

-Rikin Zala

IU2041230201

Computer Science & Engineering

TABLE OF CONTENT

ABSTRACT

In the contemporary digital music landscape, personalized recommendations are pivotal for enhancing user engagement and satisfaction. This project proposes the development of an advanced Emotion-based Music Recommendation System that leverages cutting-edge technologies to provide users with tailored music suggestions based on their emotional states. The system incorporates Speech Emotion Recognition (SER) to analyze users' vocal cues, along with APIs such as Spotify, Lyrics.ovh, and Musix-match to access comprehensive music data. Through a combination of lyrics text analysis, semantics analysis, and valence arousal analysis, the Emotion-based Music Recommendation System *gains a nuanced understanding of song emotional content.*

The core functionality of the Emotion-based Music Recommendation System *lies in its Emotion Music Classification module, which categorizes songs into emotional categories based on analysis results. Using sophisticated algorithms, the system maps emotional states to suitable music genres, artists, and tracks. A Music Provider and Mappings component ensures seamless integration with various music streaming platforms. Based on the results of the analysis, the system performs Emotion Music Classification, categorizing songs into various emotional categories such as happy, sad, calm, or anger. Additionally, it establishes mappings between detected emotions and suitable music genres, creating a personalized music recommendation experience tailored to individual preferences and moods.*

To enhance user interaction, the Emotion-based Music Recommendation System *features a user-friendly Web App interface enabling users to input emotional states, explore recommended playlists, and discover new music tailored to their mood. Additionally, users can get personalized recommended tracks, refining recommendations over time. Beyond recommendations, the system incorporates music generation functionality, using advanced algorithms to create compositions aligned with specific emotional cues.*

The Emotion-based Music Recommendation System offers a comprehensive solution for personalized music discovery and enjoyment. By integrating SER, multi-API analysis, and advanced algorithms for classification and generation, the system revolutionizes user engagement with music, providing tailored recommendations that resonate with individual emotions and preferences.

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

Abbreviations	Full Form
EMRS	Emotion-based Music Recommendation System
SER	Speech Emotion Recognition
UI	User Interface
UX	User Experience
EMC	Emotion Music Classification
MUI	Material UI
REST	Representational State Transfer
API	Application Programming Interface
CNN	Convolutional Neural Network
LTSM	Long Short-Term Memory
ML	Machine Learning
AI	Artificial Intelligence
VA	Valence Arousal
NLP	Natural Language Processing
TESS	Toronto Emotion Speech Set
SAVEE	Surrey Audio-Visual Expressed Emotion
CSV	Comma Separated Values
TER	Text Emotion Recognition
GAN	Generative Adversarial Network
RNN	Recurrent Neural Network
HTML	Hyper Text Markup Language
CSS	Cascading Style Sheets
SVM	Support Vector Machine
RAD	Rapid Application Development
WBS	Work Breakdown Structure
SRS	Software Requirement Specification
FMEA	Failure Mode and Effects Analysis
RPN	Risk Priority Numbers
CPU	Central Processing Unit

Abbreviations	Full Form
GPU	Graphics Processing Unit
RAM	Random Access Memory
HDD	Hard Disk Drive
SSD	Solid-State Drive
OS	Operating System
IDE	Integrated Development Environment
HTTP	Hyper Text Transfer Protocol
HTTPS	Hyper Text Transfer Protocol Secure
NLTK	Natural Language Tool Kit
SDK	Software Development Kit
AWS	Amazon Web Services
CORS	Cross Origin Resource Sharing
SSL	Secure Sockets Layer
TLS	Transport Layer Security
XSS	Cross-Site Scripting
CSRF	Cross-Site Request Forgery
WSGI	Web Server Gateway Interface
JS	JavaScript
CI/CD	Continuous Integration/Continuous Deployment
MP3	MPEG Audio Layer 3 Format
WAV	Waveform Audio File Format
OOP	Object Oriented Programming
UML	Unified Modelling Language
DFD	Data-Flow Diagram
JSON	Java Script Object Notation

CHAPTER 1

INTRODUCTION

1.1 PROJECT SUMMARY

1.2 PROJECT PURPOSE

1.3 PROJECT SCOPE

1.4 OBJECTIVES

1.4.1 MAIN OBJECTIVES

1.4.2 SECONDARY OBJECTIVES

1.5 TECHNOLOGY AND LITERATURE OVERVIEW

1.5.1 TECHNOLOGY

1.5.2 LITERATURE OVERVIEW

1.6 SYNOPSIS

1.1 PROJECT SUMMARY

The Emotion-based Music Recommendation System seeks to revolutionize user experience by recommending music tailored to their emotional state. Through the integration of cutting-edge methodologies such as Speech Emotion Recognition (SER), Emotion Music Classification (EMC), and a user-centric Web App interface, the system delivers personalized music suggestions aligned with the user's current emotional disposition. By tapping into speech patterns to discern emotional cues, categorizing music based on emotional content, and employing sophisticated mapping algorithms, the system ensures that music recommendations resonate deeply with the user's feelings. Furthermore, it seamlessly integrates with music providers to access vast song libraries and employs real-time emotion detection to provide instant recommendations, enhancing user engagement and satisfaction.

Methods of Implementation:

(1) Speech Emotion Recognition (SER):

- Analyzes speech patterns using machine learning models such as SER Model CNN & LTSM to extract emotional cues.
- Utilizes techniques such as voice signal processing, feature extraction, and classification to identify emotions conveyed through speech.
- Key steps involve preprocessing of speech data, feature engineering, model training, and real-time emotion prediction.

(2) Emotion Music Classification:

- Classifies music tracks into emotional categories using Spotify API.
- Employs audio feature extraction methods to capture musical attributes related to emotion, such as tempo, pitch, and rhythm.
- Trains classification models with API to accurately predict the emotional content of music tracks.

(3) Music Provider and Mapping:

- Integrates with music streaming platforms to access extensive song libraries.

- Develops mapping algorithms to correlate emotional states with suitable music selections.

(4) Web App:

- Develops a user-friendly web interface for interaction with the recommendation system.
- Enables users to input emotional cues through speech or manual selection and receive personalized music recommendations.
- Features include intuitive UI/UX design, real-time emotion detection.

1.2 PROJECT PURPOSE

The Emotion-based Music Recommendation System embarks on a transformative journey in the realm of music recommendation, driven by a singular purpose: to enrich user experience through personalized musical selections attuned to their emotional states. At its core, the project aims to decode the intricate language of emotions expressed through speech and translate these signals into curated playlists that resonate deeply with each user. Leveraging cutting-edge methodologies such as Speech Emotion Recognition (SER), the system meticulously analyzes speech patterns, extracting nuanced emotional cues to discern the user's current emotional disposition.

Furthermore, the project delves into the realm of Emotion Music Classification, where deep learning algorithms meticulously categorize music tracks based on their emotional content. By dissecting musical attributes such as tempo, pitch, and rhythm, the system can accurately predict the emotional resonance of each track, ensuring that recommendations are precisely tailored to the user's mood.

To achieve seamless integration and accessibility, the project collaborates with popular music streaming platforms (Spotify), granting users access to an extensive catalogue of songs across genres and moods. This integration allows the system to tap into vast musical libraries, ensuring a diverse and enriching listening experience for every user.

However, the project's ambitions extend beyond mere recommendation algorithms; it aspires to create an immersive and intuitive user interface through a bespoke web application. This interface serves as the gateway to a world of personalized music, empowering users to interact effortlessly with the recommendation system. Whether through speech input or manual selection, users can convey their emotional cues, receiving instant recommendations that mirror their innermost feelings. The interface prioritizes user experience, boasting intuitive design elements and real-time emotion detection to foster engagement and satisfaction.

The project's purpose crystallizes in its commitment to enhancing the emotional resonance of music listening experiences. By marrying technology with the intricacies of human emotion, it endeavours to forge deeper connections between users and their musical companions, enriching lives one melody at a time.

1.3 PROJECT SCOPE

The Emotion-based Music Recommendation System aims to revolutionize the music listening experience by harnessing the power of artificial intelligence and emotion recognition techniques. This project scope outlines the methodologies and technologies involved in developing a comprehensive system capable of analyzing user emotions, classifying music based on emotional content, and providing personalized recommendations. Key components include Speech Emotion Recognition (SER), integration with Spotify and Musix-match APIs for accessing music and lyrics data, and the development of a user-friendly web application interface. In an era where music has transcended its traditional boundaries to become an integral part of our daily lives, the Emotion-based Music Recommendation System emerges as a groundbreaking endeavour to personalize the music listening experience like never before. This project ambitiously combines cutting-edge technologies such as Speech Emotion Recognition (SER), integration with Spotify and Musix-match APIs, advanced text analysis, and deep learning algorithms such as CNN and LTSM to create a sophisticated system capable of understanding and responding to human emotions in real-time. The main scope of this project is to design and develop a comprehensive platform that not only recommends music based on the user's emotional state but also enhances the emotional resonance of the listening experience. The Emotion-based Music Recommendation System project sets out to redefine the music recommendation landscape by providing users with personalized recommendations aligned with their emotional states. By leveraging advanced technologies and methodologies, the project aims to deliver a seamless and immersive music discovery experience that enhances user engagement and satisfaction. Through the successful implementation of the project scope, the team endeavours to create a platform that enriches users' lives through the power of music.

1.4 OBJECTIVES

1.4.1 Main Objectives.

(1) Speech Emotion Recognition (SER):

- Develop robust machine learning models to analyze speech patterns and extract emotional cues from spoken input.
- Utilize techniques like voice signal processing and feature extraction to accurately identify emotions conveyed through speech.

(2) API Integration with Spotify and Musix-match:

- Integrate with Spotify API to access a vast repository of music tracks across various genres and moods.
- Utilize Musix-match API to fetch lyrics data for the analyzed songs, enabling deeper text analysis.

(3) Emotion Music Classification:

- Develop machine learning models to classify music tracks into predefined emotional categories.
- Incorporate features derived from SER, lyrics text analysis, and valence-arousal analysis to improve classification accuracy.

(4) Web Application Development:

- Design and develop a user-friendly web interface for seamless interaction with the recommendation system.
- Enable users to input emotions as an audio format.
- Implement real-time updates and personalized recommendations based on user interactions.

1.4.2 Secondary Objectives.

(1) Lyrics Text Analysis and Semantics Analysis:

- Implement natural language processing (NLP) techniques to analyze the semantic meaning and sentiment of song lyrics.
- Perform text analysis to extract emotional content and thematic elements, enriching the emotion classification process.

(2) Valence Arousal Analysis:

- Conduct valence-arousal analysis on both speech and lyrics data to quantify emotional intensity and arousal levels associated with each music track.
- Utilize valence-arousal scores as additional features for emotion classification and recommendation.

(3) Music Provider and Mappings:

- Integrate with Spotify API to access a diverse collection of music tracks.
- Use of Spotify API to correlate emotional states with suitable music selections based on the results of emotion classification and valence-arousal analysis.

(4) Music Generation or Music Generator:

- Explore the development of a music generation component within the system.
- Investigate techniques such as Generative Adversarial Networks (GANs) or Recurrent Neural Networks (RNNs) to create new music compositions based on emotional inputs or existing music patterns.
- Provide users with the option to generate custom music tracks tailored to their emotional preferences, further enhancing the personalized music listening experience.

1.5 TECHNOLOGY AND LITERATURE OVERVIEW

1.5.1 Technology

Front-end:

- **React (VITE):** Powers our dynamic user interface efficiently. VITE enhances React development by offering fast and efficient builds.

- **Typescript:** Enhances the development process with static typing and improved tooling support, enabling safer and more scalable JavaScript code. TypeScript helps catch errors early and enhances the maintainability of our front-end codebase.

- **JavaScript:** The cornerstone of web development, JavaScript enables dynamic and interactive user experiences in web applications. Leveraging JavaScript, we create responsive interfaces and handle client-side interactions seamlessly.

- **HTML & CSS:** HTML provides the structure of our web pages, defining the content and layout, while CSS styles them, enhancing their appearance and user experience. Together, HTML and CSS form the foundation for building visually appealing and accessible front-end interfaces.

Backend:

- **Python:** A versatile and powerful programming language chosen for its readability and extensive libraries. Python forms the backbone of our backend infrastructure, facilitating seamless data processing and business logic implementation.

- **Flask:** A lightweight and flexible web framework for Python, providing essential tools and libraries for building web applications. Flask's simplicity and modularity make it ideal for developing RESTful APIs and handling HTTP requests efficiently.

- **Gunicorn:** A WSGI HTTP server for Python web applications, known for its scalability and reliability. Gunicorn serves as a robust gateway between our Flask application and web clients, ensuring optimal performance and resource utilisation.

Tools:

- **Nivo Charts:** Charts for front-end: Provides visually appealing and interactive charts to present data in our application's frontend.

- **MUI (Material UI):** Material UI for front-end: Offers a rich library of components and styles for designing a modern and intuitive user interface.

- **CNN (Convolutional Neural Network):** Machine Learning CNN for Audio Speech Emotion Recognition Model: Utilised to recognize emotions from audio input, enriching the recommendation process with deeper insights.

- **NLP (Natural Language Processing):** Basic NLP for text emotion recognition: Enables understanding and processing of textual data to extract emotional cues, enhancing the accuracy of recommendations.

- **Assembly AI:** For Text Transcription: Provides accurate transcription services, facilitating the conversion of audio data into text for further analysis and processing.

- **Sentence Transformers:** For Keywords Extraction & Finding Similarity: Employs advanced techniques to extract keywords and determine similarity between texts, improving recommendation accuracy based on textual inputs.

- **Emopia**: For Music Generation based on Emotions predicted.

1.5.2 Literature Overview

Emotion-based Music Recommendation Systems (EMRS) have garnered significant attention in recent years due to their potential to enhance user satisfaction and engagement by delivering personalized music recommendations aligned with users' emotional states. This literature overview provides a comprehensive survey of existing research and developments in the field of EMRS, highlighting key approaches, methodologies, challenges, and future directions.

1. Emotion Recognition Techniques:

- Research in EMRS often begins with emotion recognition techniques, which aim to identify and infer users' emotional states from various modalities such as speech and user interactions.
- Common techniques include Speech Emotion Recognition (SER and sentiment analysis of textual data fetched from the Audio.
- These techniques form the foundation for understanding users' emotional responses to music and inform the recommendation process.

2. Music Feature Analysis:

- EMRS leverages music feature analysis to extract relevant audio features, lyrics, semantics, and metadata to characterize the emotional content of songs.
- Audio features such as tempo, rhythm, pitch, and timbre are analyzed to quantify the emotional attributes of music.
- Textual analysis of lyrics and semantic analysis of song metadata provide additional insights into the emotional context of songs, enriching the recommendation process.

3. Recommendation Algorithms:

- Various recommendation algorithms have been proposed for EMRS, ranging from collaborative filtering and content-based filtering to hybrid approaches that combine multiple techniques.
- Collaborative filtering algorithms leverage user preferences and behaviours to recommend music like those liked by users with similar tastes.

- Content-based filtering algorithms analyze the content and features of songs to recommend music that matches users' preferences and emotional states.
- Hybrid recommendation algorithms combine collaborative and content-based approaches to provide more accurate and diverse recommendations.

4. User Modelling and Personalization:

- EMRS incorporates user modelling techniques to create personalized profiles that capture users' preferences and emotional states.
- User modelling enables the system to adapt recommendations to individual users' tastes, moods, and contextual factors, enhancing the relevance and effectiveness of recommendations.
- Context-aware recommendation techniques consider situational factor recommendations to users' current mood.

5. Evaluation Metrics and Challenges:

- Evaluating the effectiveness of EMRS poses several challenges due to the subjective nature of emotions and the complexity of music preferences.
- Common evaluation metrics include accuracy and coverage of recommendations.
- Challenges in EMRS research include data sparsity, cold-start problems, scalability issues, and the need for real-time processing of user inputs.

EMRS represents a promising research area with significant potential to revolutionize the way users discover and engage with music. By integrating emotion recognition techniques, music feature analysis, recommendation algorithms, and user modelling, EMRS aims to deliver personalized and emotionally resonant music recommendations tailored to individual users' preferences and moods.

1.6 SYNOPSIS

The Emotion-based Music Recommendation System is an innovative project aimed at revolutionizing the way users discover and enjoy music by leveraging cutting-edge technologies and methodologies. The system incorporates various methods of implementation, including Speech Emotion Recognition (SER), integration with Spotify API and Musix-match API for accessing music and lyrics data, text analysis techniques such as Semantics Analysis and Valence Arousal Analysis, Emotion Music Classification, Music Provider and Mappings, and the development of a user-friendly Web App interface. The Emotion-based Music Recommendation System represents a groundbreaking endeavour poised to redefine how users discover and engage with music. By harnessing state-of-the-art technologies and innovative methodologies, this project aims to create a personalized music recommendation web app that resonates deeply with users' emotions. Through the implementation of Speech Emotion Recognition (SER), the system can discern users' emotional states from spoken input, providing a seamless and intuitive means of interaction. Integration with the Spotify API and Musix-match API grants access to vast music catalogues and lyrics data, enabling comprehensive analysis to fuel emotion-driven recommendations.

Text analysis techniques, including Semantics Analysis and Valence Arousal Analysis, further enhance the system's understanding of the emotional content of songs. By dissecting lyrics and quantifying emotional intensity and arousal levels, the system can categorize music tracks into predefined emotional categories. Leveraging deep learning algorithms, the system can then classify and recommend music aligned with users' emotional preferences.

Central to the project's success is the development of a user-friendly web application interface, providing users with a seamless and immersive music discovery experience. Through the interface, users can input emotional cues. Real-time updates and personalized recommendations based on user interactions further enhance user engagement and satisfaction. Ultimately, the Emotion-based Music Recommendation System aims to elevate the music listening experience by delivering tailored recommendations that evoke the perfect emotional response, enriching lives through the power of music.

CHAPTER 2

LITERATURE SURVEY

2.1 INTRODUCTION OF SURVEY

2.2 WHY SURVEY?

2.3 SURVEY, REVIEW AND RESEARCH

2.1 INTRODUCTION OF SURVEY

Emotion-based music recommendation systems aim to enhance user satisfaction and engagement by recommending music that aligns with their emotional state. This literature survey explores various methods and techniques employed in the implementation of such systems, including speech emotion recognition, text analysis, semantics analysis, valence arousal analysis, and integration of music APIs like Spotify and Musix-match. Additionally, it discusses the incorporation of machine learning algorithms for emotion-based music classification and the development of web applications for user interaction.

(1) Speech Emotion Recognition (SER):

- SER is a crucial component in understanding users' emotional states from speech data.
- Relevant works include "Speech Emotion Recognition: Features and Classification Models" by Yildirim et al., which discusses various feature extraction techniques and classification models for SER.
- "Deep Learning for Emotion Recognition on Small Datasets Using Transfer Learning" by Tzirakis et al. explores deep learning approaches for SER, particularly transfer learning strategies to handle small datasets effectively.

(2) Text Analysis and Semantics Analysis:

- Text analysis techniques are utilized for analyzing lyrics and extracting emotional cues.
- "Emotion Analysis of Music Lyrics Using Text Mining Techniques" by Lin et al. discusses the application of text mining techniques for extracting emotional content from song lyrics.
- Semantics analysis involves understanding the meaning and context of the text, contributing to a deeper understanding of emotional content. Works such as "Emotion-Based Music Recommendation by Association Rules and Word Embeddings" by Zhang et al. employ word embeddings and association rules to understand the semantics of song lyrics.

(3) Valence Arousal Analysis:

- Valence and arousal dimensions are commonly used to represent emotions in music.

- "Modeling the Valence and Arousal of Musical Emotions with Support Vector Regression" by Yang et al. proposes a support vector regression model to predict valence and arousal ratings of music pieces based on audio features.
- "Predicting Musical Emotions from Physiological Data and Content-based Features" by Coutinho et al. explores the prediction of musical emotions using physiological data and content-based features, including valence and arousal.

(4) Emotion Music Classification:

- Machine learning algorithms are applied for emotion-based music classification.
- "A Survey on Music Emotion Recognition: From Signals to Features" by Li et al. provides an overview of various methods and features used for music emotion recognition.
- "Music Emotion Recognition Using Machine Learning Techniques" by Han et al. discusses the application of machine learning techniques such as SVM and neural networks for music emotion recognition.

(5) Integration of Music APIs and Web App Development:

- APIs like Spotify and Musix-match provide access to vast music libraries and metadata.
- "Building Music Recommendation Services Using the Spotify Web API" by McFee et al. explores the use of Spotify API for building music recommendation systems.
- "Design and Implementation of a Music Recommendation System with a Web Interface" by Park et al. details the development of a web application for music recommendation, incorporating APIs and user interaction features.

2.2 WHY SURVEY?

The literature survey provided is essential for several reasons:

(1) **Understanding Existing Work:** Conducting a literature survey helps researchers understand the existing body of knowledge and research efforts related to the proposed project topic. By reviewing relevant literature, researchers can gain insights into various methodologies, techniques, and approaches used by other researchers in the field.

(2) **Identifying Gaps and Opportunities:** Through the survey, we can identify gaps in the existing literature. These gaps could be areas where further research is needed or opportunities for innovation and improvement in existing methodologies. By pinpointing these gaps, we can refine their project objectives and contribute to the advancement of knowledge in the field.

(3) **Informing Methodological Choices:** The literature survey provides valuable information on different methods and techniques used in similar projects. Researchers can learn from the successes and limitations of previous approaches and make informed choices about the methodology they will employ in their own project. For example, the survey might reveal which machine learning algorithms are commonly used for emotion recognition or which APIs are best suited for accessing music data.

(4) **Validation and Credibility:** A thorough literature survey adds credibility to the research project by demonstrating that the proposed work is built upon a solid foundation of existing knowledge and research findings. It shows that the researchers are aware of the relevant work done in the field and are positioning their project within the broader context of existing research efforts.

(5) **Ideas for Future Work:** In addition to informing the current project, the literature survey can also inspire ideas for future research directions. By identifying trends, emerging technologies, and unanswered questions in the literature, researchers can envision potential avenues for further exploration beyond the scope of the current project.

Overall, a literature survey is a critical step in the research process, helping researchers situate their work within the existing scholarly landscape, identify opportunities for contribution, and make informed decisions about the design and execution of their project.

2.3 SURVEY, REVIEW AND RESEARCH

For the topic of "Emotion-based Music Recommendation System," several types of surveys are commonly conducted to gather relevant information and insights. These surveys can be categorized based on their focus, methodology, and target audience. Here are some types of surveys typically conducted for this topic:

(1) **Literature Review:** This type of survey involves reviewing existing academic literature, research papers, articles, and books related to emotion-based music recommendation systems. The purpose is to understand the theoretical foundations, methodologies, techniques, and findings of previous research in the field. A literature review helps identify gaps in the existing knowledge and informs the direction of the research project.

(2) **User Needs and Preferences Survey:** Understanding user preferences and needs is crucial for designing effective music recommendation systems. Surveys targeting music listeners can be conducted to gather data on their music listening habits, emotional responses to music, preferred genres, and features they value in a recommendation system. This type of survey may include questions about how users perceive emotions in music, their expectations from a recommendation system, and their willingness to try new approaches.

(3) **Expert Opinion:** Experts in fields such as music psychology, machine learning, human-computer interaction, and music recommendation systems can provide valuable insights into the design and implementation of emotion-based music recommendation systems. Surveys targeting experts may include questions about emerging trends, best practices, potential challenges, and future directions in the field. Expert opinions can help validate research hypotheses, provide guidance on methodology, and contribute to the theoretical framework of the project.

(4) **Market Research:** For researchers interested in commercializing their emotion-based music recommendation system or collaborating with industry partners, market research surveys can be conducted to assess market demand, competition, pricing strategies, and potential business models. This type of survey may target music streaming platforms, digital music services, record labels in the music industry.

Overall, the choice of survey types depends on the research objectives, target audience, and stage of the research project. Combining multiple survey methodologies can provide a comprehensive understanding of user needs, expert opinions, market trends, and the state of the art in emotion-based music recommendation systems.

CHAPTER 3

PROJECT MANAGEMENT

3.1 PROJECT PLANNING OBJECTIVES

3.1.1 SOFTWARE SCOPE

3.1.2 RESOURCE

3.1.2.1 HUMAN RESOURCE

3.1.2.2 REUSABLE SOFTWARE RESOURCES

3.1.2.3 ENVIRONMENT RESOURCE

3.1.3 PROJECT DEVELOPMENT APPROACH

3.2 PROJECT SCHEDULING

3.2.1 BASIC PRINCIPLES

3.2.2 COMPARTMENTALIZATION

3.2.3 WORK BREAKDOWN STRUCTURE

3.2.4 PROJECT ORGANIZATION

3.2.5 TIMELINE CHART

3.2.5.1 TIME ALLOCATION

3.2.5.2 TASK SETS

3.3 RISK MANAGEMENT

3.3.1 RISK IDENTIFICATION

3.3.1.1 RISK IDENTIFICATION ARTIFACTS

3.3.2 RISK PROJECTION

3.1 PROJECT PLANNING OBJECTIVES

In this project, we aim to delve into machine learning models that provide music based on human's speech emotions, followed with a web application that implements the model. Four separate major objectives for achieving our goal of Emotion-based Music Provider are listed below:

- Objective 1: Classifying emotion expressed by human's speech
- Objective 2: Recommending existing music to the user based on a specified emotion
- Objective 3: Generating symbolic music of certain emotion
- Objective 4: A web application that achieves Emotion-based Music Provide

These objectives provide a framework for guiding the planning, execution, and management of the project.

3.1.1 Software Scope:

The software scope defines the boundaries and functionalities of the EMRS. It includes features such as emotion recognition, music recommendation, user interface design and integration with external APIs (*e.g., Spotify, Lyrics.ovh, Musix-match*). Clarifying the software scope ensures that the development team and experts have a shared understanding of the project's deliverables and requirements. It outlines the features, functionalities, and capabilities that the EMRS will encompass. By defining the software scope, our team can establish clear expectations and priorities for the development of the EMRS, ensuring that the final product meets the needs and requirements of its users effectively.

3.1.2 Resource:

Resource planning involves identifying and allocating the necessary resources to support the development of the EMRS. Resources are essential components required for the development, deployment, and maintenance of the EMRS. They include human resources, reusable software resources, and environmental resources necessary to support the project.

3.1.2.1 Human Resource:

Human resources refer to the individuals involved in the EMRS project, including project managers, software developers, designers, testers, and other team members. The success of the project depends on the skills, expertise, and collaboration of the human resources

involved. Adequate staffing, training, and communication are essential to ensure effective coordination and execution of project tasks.

3.1.2.2 Reusable Software Resources:

Reusable software resources encompass existing software components, libraries, frameworks, and modules that can be leveraged to accelerate the development process of the EMRS. These resources may include open-source libraries, third-party APIs, and pre-built software modules that provide functionality related to emotion recognition, music processing, recommendation algorithms, and user interface components. By utilizing reusable software resources, developers can reduce development time, minimize errors, and enhance the overall quality of the EMRS.

3.1.2.3 Environment Resource:

Environment resources include hardware, software tools, development environments, and infrastructure needed to support the development and testing of the EMRS. This may include computing resources, version control systems, integrated development environments (IDEs), databases, and hosting services. Adequate provisioning and setup of environment resources are essential to ensure smooth project execution and collaboration.

3.1.3 Project Development Approach:

The project development approach outlines the methodology or framework used to manage the development process of the EMRS. In this case, Rapid Application Development (RAD) is chosen as the approach for its iterative, incremental, and prototyping-based nature. RAD emphasizes rapid iterations, user feedback, and quick prototyping, enabling faster development and adaptation to changing requirements. By adopting RAD, the Emotion based Music Recommended System project aims to deliver a functional prototype quickly while maintaining flexibility to incorporate user feedback and refine features iteratively.

3.2 PROJECT SCHEDULING

Project scheduling for the Emotion-based Music Recommended System involves creating a timeline and allocating resources to complete various tasks and milestones within specified timeframes. The scheduling process ensures that the project progresses smoothly and efficiently, meeting deadlines and delivering the desired outcomes. Key steps in project scheduling include:

- **Identifying Tasks and Deliverables**: Break down the project into smaller tasks and deliverables, such as requirements gathering, system design, development, testing, and deployment.
- **Estimating Durations**: Estimate the time required to complete each task or deliverable based on factors such as complexity, dependencies, and resource availability.
- **Sequencing Tasks**: Determine the order in which tasks should be performed based on dependencies and constraints. Identify critical paths and milestones that are essential for project success.
- **Resource Allocation**: Allocate resources, including personnel, equipment, and budget, to each task or deliverable. Ensure that resources are sufficient to meet project requirements and timelines.
- **Creating the Schedule**: Develop a project schedule that outlines start and end dates for each task or deliverable, along with dependencies and milestones. Use scheduling tools and techniques such as Gantt charts or network diagrams to visualize and manage the schedule.
- **Monitoring and Controlling:** Continuously monitor project progress against the schedule, identifying any deviations or delays. Take corrective actions as needed to keep the project on track and minimize risks to its success.

3.2.1 Basic Principles:

User-Centric Design: The EMRS should prioritize the user's emotional experience and preferences in music recommendation. User-centric design principles ensure that the system effectively captures and responds to users' emotional cues, providing personalized recommendations tailored to individual tastes and moods.

Emotion Recognition and Understanding: Emotion recognition is a fundamental principle of the EMRS, enabling the system to analyze users' emotional states and preferences accurately. By incorporating advanced emotion recognition technologies, such as Speech Emotion Recognition (SER) and sentiment analysis, the system can infer users' emotions from vocal cues, voice inputs, and interaction patterns.

Multi-modal Analysis: The EMRS should employ multi-modal analysis techniques to comprehensively understand the emotional content of music. By analyzing audio features, lyrics, semantics, and user interactions, the system gains a holistic view of the emotional context of songs, facilitating more accurate and nuanced recommendations.

Personalization and Contextualization: Personalization is key to the success of the EMRS, as it allows the system to tailor recommendations to each user's unique preferences, and emotional states. Contextualization ensures that recommendations are relevant and appropriate for the user's current mood.

Transparency and Explainability: Transparency and explainability are essential principles of the EMRS, ensuring that users understand how recommendations are generated and why specific songs are suggested. Providing transparent explanations of the recommendation process builds trust and confidence in the system, enhancing user engagement and satisfaction.

By adhering to these basic principles, the Emotion-based Music Recommendation System (EMRS) can effectively deliver personalized, emotion-aware music recommendations that enhance user satisfaction, engagement, and enjoyment.

3.2.2 Compartmentalization:

Compartmentalization involves breaking down the overall project into smaller, manageable components or modules. Each component focuses on specific functionalities or aspects of the system, allowing for parallel development and easier integration. Key aspects of compartmentalization include:

Modular Design: Design the EMRS system architecture using a modular approach, with distinct modules for emotion recognition, music recommendation, integration of multi-model structure and user interface.

Task Allocation: Assign development tasks to each of our teammates based on their areas of expertise and responsibilities. Each of us focuses on implementing and testing a specific module or component of the system.

Inter-module Communication: Define clear interfaces and communication protocols between modules to facilitate data exchange and interoperability. Ensure that modules interact seamlessly to achieve the desired functionality and user experience.

Testing and Integration: Conduct thorough testing of each module independently to verify its functionality and performance. Once individual modules are tested and validated, integrate them into the larger system and perform comprehensive integration testing to ensure compatibility and functionality across all components.

3.2.3 Work Breakdown Structure

A Work Breakdown Structure (WBS) is a hierarchical decomposition of the work to be executed by a project team to accomplish the project objectives and create the required deliverables. It organizes and defines the total scope of the project into manageable sections, providing a framework for organizing and managing the project's activities.

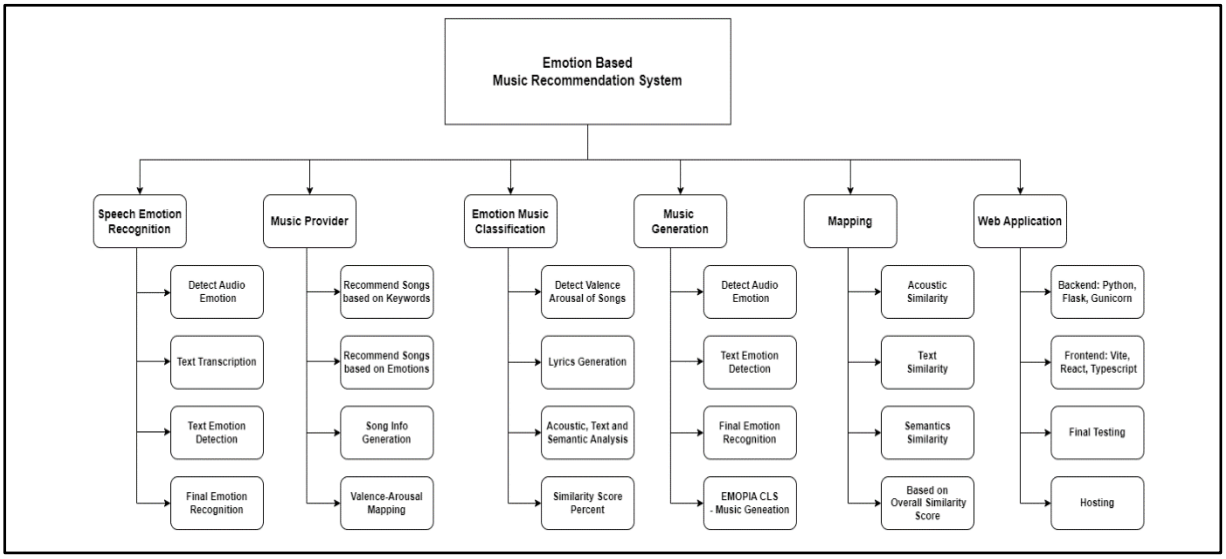


Fig. 3.1 Work Breakdown Structure

1. Speech Emotion Recognition

- Detect Audio Emotion
- Text Transcription

- Text Emotion Detection
- Final Emotion Recognition

2. Music Provider

- Recommend Songs based on Keywords
- Recommend Songs based on Emotions
- Song Info Generation
- Valence Arousal Mapping

3. Emotion Music Classification

- Detect Valence Arousal of Songs
- Lyrics Generation
- Acoustic Analysis
- Text Analysis
- Semantics Analysis
- Similarity Score Percent

4. Music Generation

- Detect Audio Emotion
- Text Emotion Detection
- Final Emotion Recognition
- EMOPIA CLS - Music Generation

5. Mapping

- Acoustic Similarity
- Text Similarity
- Semantics Similarity
- Overall Similarity Score

6. Web Application

- Backend Development
    - Python
    - Flask
    - Gunicorn
- Frontend Development
    - Vite
    - React

    - Typescript
- Final Testing
- Web Hosting

This breakdown organizes the project tasks into logical groupings, making it easier to manage and track progress. Each major task is further divided into subtasks, ensuring that all aspects of the project are adequately addressed.

3.2.4 Project Organization:

Effective project organization is essential for coordinating activities, allocating resources, and managing each of the tasks throughout the EMRS development lifecycle. Key elements of project organization include:

Project Team Structure: Define roles and responsibilities within the project team, including design, development, testing, gather feedback and implementing as per requirements. Establish clear lines of communication and accountability to ensure smooth collaboration and decision-making.

Documentation and Reporting: Maintain accurate and up-to-date documentation of project plans, requirements, schedules, and progress reports. Provide regular updates and status reports to mentor to communicate project status, milestones, and any issues or concerns that may arise.

By implementing sound project scheduling, adhering to basic principles, embracing compartmentalization, and establishing effective project organization, the EMRS development project can be managed efficiently and successfully to deliver a high-quality, emotion-aware music recommendation system that meets the needs and expectations of the users.

3.2.5 TimeLine Chart

A Timeline Chart visually represents the chronological sequence of tasks, milestones, and events within a software project. It provides a clear overview of project progress, dependencies, and deadlines, aiding in project planning, tracking, and communication among team members.

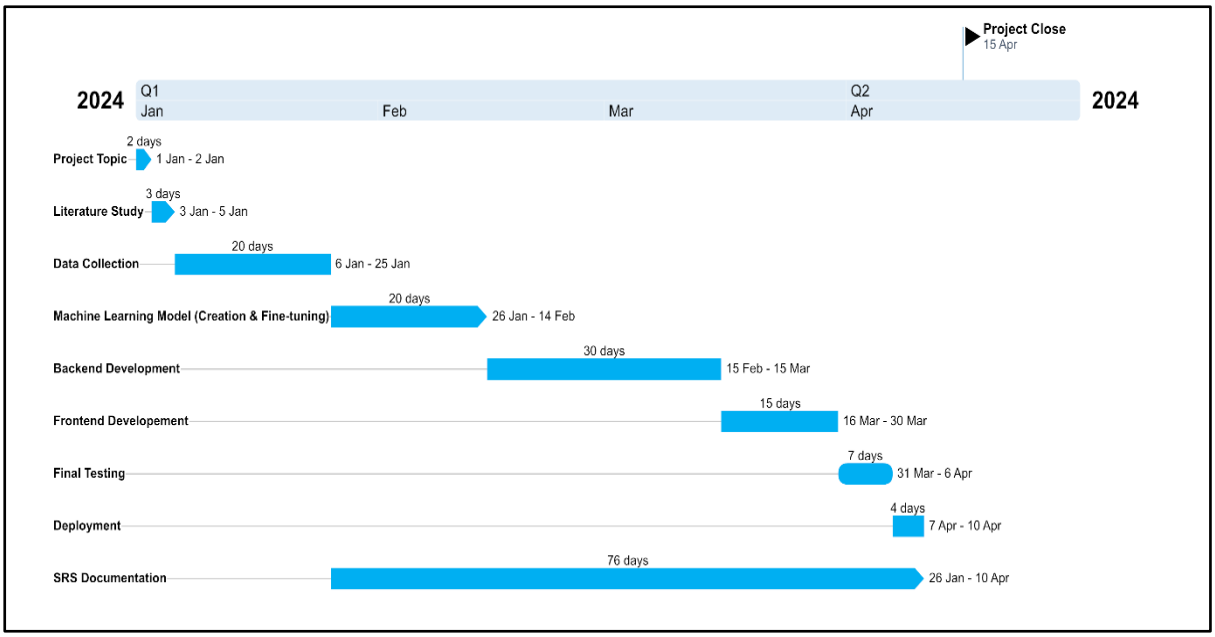


Fig. 3.2 Timeline Chart

3.2.5.1 Time Allocation and Task Sets

Here is a breakdown of Time Allocation and Task Sets for the given list of tasks:

(1) Project Topic Discussion, Decision and Selection

- Duration: 2 days (1 Jan to 2 Jan)
- Task Set:
    - Day 1 (1 Jan): Brainstorm project ideas, discuss potential topics.
    - Day 2 (2 Jan): Finalize project selection.

(2) Literature Study of Selected Project Topic

- Duration: 3 days (3 Jan to 5 Jan)
- Task Set:
    - Day 1-3 (3 Jan to 5 Jan): Research relevant literature, gather information about the selected project topic.

(3) Data Collection for Selected Project Topic

- Duration: 20 days (6 Jan to 25 Jan)
- Task Set:
    - Day 1-20 (6 Jan to 25 Jan): Gather necessary data, datasets, and information required for the project.

(4) Machine Learning Model Implementation, i.e., Creation and Fine-tuning

- Duration: 20 days (26 Jan to 14 Feb)
- Task Set:
    - Day 1-20 (26 Jan to 14 Feb): Develop and fine-tune machine learning models as per project requirements.

(5) Backend Development

- Duration: 30 days (15 Feb to 15 March)
- Task Set:
    - Day 1-30 (15 Feb to 15 March): Implement backend functionalities, database integration, and server-side logic.

(6) Frontend Development

- Duration: 15 days (16 March to 30 March)
- Task Set:
    - Day 1-15 (16 March to 30 March): Design and develop the user interface, frontend components, and client-side interactions.

(7) Final Testing

- Duration: 7 days (31 March to 6 April)
- Task Set:

- Day 1-7 (31 March to 6 April): Conduct comprehensive testing, debug and fix issues, ensure the software meets quality standards.

(8) Deployment

- Duration: 4 days (7 April to 10 April)
- Task Set:
  - Day 1-4 (7 April to 10 April): Prepare for deployment, configure servers, deploy the application to production environment.

(9) SRS Documentation

- Duration: 76 days (26 Jan to 10 April)
- Task Set:
  - Ongoing throughout the project duration: Document software requirements, system architecture, and other project documentation as per Software Requirements Specification (SRS) guidelines.

3.3 RISK MANAGEMENT

Risk management is a crucial aspect of the Emotion-based Music Recommendation System (EMRS) project, aiming to identify, assess, and mitigate potential risks that may impact the project's success.

3.3.1 Risk Identification

Risk identification involves systematically identifying potential threats and opportunities that may affect the project's objectives. The following steps are taken for risk identification:

3.3.1.1 Risk Identification Artifacts

Brainstorming Sessions: Organize brainstorming sessions with the project team to generate a comprehensive list of potential risks. Encourage team members to identify risks related to technology, resources, requirements, schedule, and external factors.

Historical Data Analysis: Review historical data from similar projects or industry benchmarks to identify common risks and lessons learned. Analyze past project documentation, post-mortem reports, and risk registers to identify recurring themes and patterns.

Checklists and Templates: Utilize risk checklists, templates, and frameworks such as Failure Mode and Effects Analysis (FMEA) to systematically identify potential risks across different project domains. These tools provide structured guidelines for risk identification.

Expert Judgment: Seek input from subject matter experts and experienced professionals in relevant domains to identify risks based on their expertise and insights. Expert judgment can help identify risks that may not be apparent to the project team.

3.3.2 Risk Projection

Risk projection involves analyzing and prioritizing identified risks based on their potential impact and likelihood of occurrence. The following steps are taken for risk projection:

Risk Assessment: Assess each identified risk based on its impact on project objectives, such as scope, schedule, cost, quality, and users' satisfaction. Evaluate the likelihood of occurrence and the severity of consequences associated with each risk.

Risk Categorization: Categorize identified risks into different risk categories, such as technical risks, organizational risks, external risks, and project management risks. This categorization helps in prioritizing and addressing risks effectively.

Risk Prioritization: Prioritize identified risks based on their severity, likelihood, and potential impact on project success. Use techniques such as risk matrices, risk scoring, or risk priority numbers (RPN) to rank risks and focus attention on high-priority risks.

Risk Register: Maintain a risk register or risk log to document identified risks, their characteristics, and the proposed mitigation strategies. The risk register serves as a central repository for tracking and managing risks throughout the project lifecycle.

Mitigation Planning: Develop mitigation plans for high-priority risks, outlining proactive measures to reduce the likelihood or impact of the risk occurring. Assign responsibilities, allocate resources, and establish contingency plans to address identified risks effectively.

By systematically identifying and projecting risks, the Emotion-based Music Recommendation System project can proactively anticipate and address potential challenges, minimizing disruptions and enhancing the likelihood of project success. Regular review and monitoring of the risk register ensure that risks are managed effectively throughout the project lifecycle.

CHAPTER 4

SYSTEM REQUIREMENTS

4.1 USER CHARACTERISTICS

4.2 FUNCTIONAL REQUIREMENT

4.2.1 ACTIVITY AND PROPOSED SYSTEM

4.3 NON-FUNCTIONAL REQUIREMENT

4.4 HARDWARE AND SOFTWARE REQUIREMENT

4.4.1 HARDWARE REQUIREMENT

4.4.2 SOFTWARE REQUIREMENT

4.4.3 SERVER-HOSTING REQUIREMENT

4.1 USER CHARACTERISTICS

User characteristics in the context of a project refer to the traits, preferences, behaviours, and needs of the individuals who will interact with or benefit from the project. Understanding user characteristics is crucial for designing and implementing systems that effectively cater to users' requirements and provide a satisfactory user experience. In the case of an emotion-based music recommendation system, identifying user characteristics helps in tailoring the recommendations to suit the emotional states and preferences of the users. Here are some examples of user characteristics relevant to this project:

1. **Emotional States**: Users may have varying emotional states such as happiness, sadness, calmness, or anger at different times. Understanding these emotional states can help in recommending music that aligns with or influences their mood.
2. **Music Preferences**: Users have diverse music preferences based on genres, artists, tempo, instrumentation, etc. Some users may prefer upbeat pop music, while others may lean towards classical or instrumental tracks. Understanding these preferences helps in providing personalized recommendations.
3. **Technical Proficiency**: Users may vary in their technical proficiency and comfort with using digital platforms or interfaces. Designing a user-friendly interface that caters to users with different levels of technical expertise is essential.
4. **Context of Use**: Users may interact with the music recommendation system in various contexts such as commuting, exercising, studying, or relaxing at home. Understanding the context of use helps in recommending suitable music for different situations.

Now, let us relate these user characteristics to the implementation methods mentioned in the project topic:

- **Speech Emotion Recognition**: Users' emotional states can be inferred from their speech patterns, allowing the system to recommend music that matches their current emotional state.
- **Spotify API and Musix-match API**: Leveraging these APIs provides access to users' listening histories, preferences, and playlists, enabling personalized recommendations based on their music taste and preferences.

- **Lyrics Text Analysis and Semantics Analysis**: Analysing lyrics can help in understanding the emotional content of songs, allowing the system to recommend music that resonates with users' emotions.
- **Valence Arousal Analysis**: Analysing the valence (positivity/negativity) and arousal (intensity) of music can help in categorizing songs based on their emotional impact and recommending appropriate music for users' current mood.
- **Emotion Music Classification**: Classifying music based on emotional characteristics enables the system to provide tailored recommendations aligned with users' emotional states and preferences.
- **Music Provider and Mappings**: Integrating with music providers and mapping their catalogues based on emotional characteristics facilitates access to a wide range of music suitable for different user preferences and emotional states.
- **Web App**: Designing a user-friendly web app that considers users' technical proficiency, context of use, and feedback preferences enhances the overall user experience and encourages continued engagement with the system.

By considering these user characteristics and integrating them into the design and implementation of the emotion-based music recommendation system, you can create a more personalized and effective platform that resonates with users' emotional needs and preferences.

4.2 FUNCTIONAL REQUIREMENT

Functional requirements define the specific functions or features that a system must perform to meet the needs of its users and achieve its intended goals. These requirements outline the system's behaviour and capabilities, detailing what actions the system should be able to perform.

In the context of a project topic like an emotion-based music recommendation system, functional requirements would specify the functionalities and capabilities the system should have to effectively recommend music based on users' emotions.

4.2.1 Activity and Proposed System

In the context of a project, "Activity" refers to the tasks or actions that need to be carried out to achieve the project's objectives. These activities are typically organized in a sequence and may involve various processes, such as data collection, analysis, design, development, testing, and deployment. Each activity contributes to the overall progress of the project and is essential for its successful completion.

"Proposed System" refers to the envisioned solution or system that is proposed to address the requirements and objectives of the project. It includes the design, components, functionalities, and features of the system that will be developed or implemented to meet the project's goals. The proposed system outlines how the project will be executed and what outcomes or deliverables are expected.

Activities:

1. **Requirement Analysis**: This involves understanding the needs and expectations of users for the emotion-based music recommendation system. It includes gathering requirements related to functionality, user experience, and technical constraints.
2. **Data Collection**: Collecting data necessary for training and testing the recommendation system. This may include emotional speech samples, music metadata, lyrics data, and user preferences from sources like Spotify and Musix-match APIs.
3. **Preprocessing**: Preprocessing the collected data to prepare it for analysis and modelling. This may involve cleaning the data, extracting features, and formatting it for further processing.

4. **Speech Emotion Recognition**: Developing or implementing algorithms to recognize emotions from speech input. This activity involves signal processing techniques, machine learning models, and possibly training data collection.
5. **Lyrics Text Analysis**: Analyzing the lyrics of songs to extract emotional content and themes. Natural language processing (NLP) techniques such as sentiment analysis and topic modelling may be employed for this purpose.
6. **Semantics Analysis**: Performing semantic analysis on lyrics to understand the underlying meaning and emotional context. This may involve techniques such as word embedding models and semantic similarity analysis.
7. **Valence Arousal Analysis**: Analyzing the valence and arousal of songs based on their acoustic features or lyrics. This may involve machine learning models trained on labelled emotional data.
8. **Emotion Music Classification**: Developing classification models to categorize music into emotional categories based on the results of valence arousal analysis. This may involve supervised learning techniques such as classification algorithms.
9. **Integration with APIs**: Integrating with Spotify API and Musix-match API to access music metadata, lyrics, and user preferences for generating recommendations.
10. **Music Provider and Mappings**: Mapping music tracks from different providers based on emotional characteristics. This involves creating a database or repository of music tracks with associated emotional labels.
11. **Recommendation Engine**: Developing a recommendation engine that considers users' emotional states, preferences, and contextual factors to suggest relevant music. This may involve collaborative filtering, content-based filtering, or hybrid recommendation techniques.
12. **Web App Development**: Designing and developing a web application interface for the emotion-based music recommendation system. This involves frontend and backend development, user interface design, and integration with backend systems and APIs.

Proposed System:

The proposed system for the emotion-based music recommendation system will comprise the following components:

1. **Speech Emotion Recognition Module**: This module will recognize users' emotions from speech input using signal processing and machine learning techniques.
2. **Lyrics Analysis Module**: This module will analyze the emotional content and themes of song lyrics using natural language processing techniques.
3. **Valence Arousal Analysis Module**: This module will analyze the valence and arousal of songs based on their acoustic features or lyrics.
4. **Emotion Music Classification Module**: This module will classify music into emotional categories based on the results of valence arousal analysis.
5. **API Integration Module**: This module will integrate with Spotify API and Musix-match API to access music metadata, lyrics, and user preferences.
6. **Recommendation Engine**: This module will generate personalized music recommendations based on users' emotional states, preferences, and contextual factors.
7. **Web Application Interface**: This module will provide a user-friendly web interface for users to interact with the recommendation system, browse recommended music, and provide feedback.

Overall, the proposed system will leverage speech emotion recognition, lyrics analysis, valence arousal analysis, and music classification techniques to recommend music tailored to users' emotional states and preferences. It will integrate with external APIs to access music data and provide a seamless user experience through a web application interface.

4.3 NON-FUNCTIONAL REQUIREMENT

Non-functional requirements define the criteria that describe how a system should perform, rather than what it should do. They specify the quality attributes, constraints, and other criteria that affect the overall system behaviour, rather than its specific functionalities. Non-functional requirements are often related to system performance, usability, reliability, and other aspects that contribute to the overall user experience. Here are some examples of non-functional requirements relevant to the project topic "Emotion-based Music Recommendation System":

1. **Performance**: This refers to the system's responsiveness and efficiency in processing user requests and generating recommendations. Example non-functional requirements related to performance include:
    - The system should respond to user queries within seconds.
    - The recommendation engine should be capable of handling concurrent requests from multiple users without significant degradation in performance.
2. **Reliability**: Reliability concerns the system's ability to consistently perform its functions accurately without failure. Example non-functional requirements related to reliability include:
    - The system should have an uptime of at least 99%.
    - The recommendation engine should be resilient to errors and failures, ensuring uninterrupted service to users.
3. **Scalability**: Scalability refers to the system's ability to handle increasing loads and accommodate a growing number of users or data without sacrificing performance. Example non-functional requirements related to scalability include:
    - The system should be capable of scaling horizontally to accommodate a tenfold increase in the number of users.
    - The recommendation engine should scale dynamically based on the workload to maintain consistent performance.

4. **Usability**: Usability concerns the ease of use and intuitiveness of the system interface for users. Example non-functional requirements related to usability include:
    - The web application interface should be intuitive and easy to navigate, even for users with minimal technical expertise.
    - The system should provide clear and informative feedback to users on their interactions and recommendations.
5. **Maintainability**: Maintainability refers to the ease with which the system can be maintained, updated, and modified over time. Example non-functional requirements related to maintainability include:
    - The system should be well-documented with clear guidelines for maintenance and updates.
    - The codebase should be modular and well-structured to facilitate future enhancements and modifications.
6. **Compatibility**: Compatibility concerns the system's ability to operate seamlessly with different devices, browsers, and platforms. Example non-functional requirements related to compatibility include:
    - The web application should be compatible with major web browsers such as Chrome, Firefox, and Safari.
    - The system should support access from both desktop and mobile devices without compromising functionality or user experience.

These non-functional requirements are essential for ensuring the overall effectiveness, performance, security, and usability of the emotion-based music recommendation system. They complement the functional requirements and contribute to delivering a high-quality and satisfactory user experience.

4.4 HARDWARE AND SOFTWARE REQUIREMENT

Hardware and software requirements are essential components of any project, outlining the necessary infrastructure and technology needed to develop, deploy, and operate the system effectively.

4.4.1 Hardware Requirement

The hardware requirements for an emotion-based music recommendation system involve specifying the necessary physical components to support the development, deployment, and operation of the system. Since the project involves various methods of implementation such as speech emotion recognition, API integration, text analysis, emotion classification, and web application development, the hardware requirements should be capable of handling these tasks efficiently. Here are the hardware requirements for the project:

- **Processor (CPU)**: High Computing power and speed of the CPU required to execute the software efficiently. GPU is better option.
- **Memory (RAM)**: Minimum 4GB of random-access memory (RAM) needed to run the software and store temporary data. Insufficient RAM can lead to performance issues or system crashes.
- **Network Components**: Specifies networking hardware such as routers, switches, and network cables needed to connect the system to the internet or local network. High-speed internet connectivity may be necessary for cloud-based systems or accessing external resources.
- **Peripherals**: Standard peripherals such as monitors, keyboards, mic are required for user interaction. Specialized input/output devices may be necessary for speech input and audio playback, depending on the implementation of the speech emotion recognition and music recommendation features. Audio output devices such as speakers or headphones are needed for listening to recommended music tracks.

Overall, the hardware requirements for an emotion-based music recommendation system should prioritize processing power, memory capacity, storage space, network connectivity, and peripherals to support the various components and functionalities of the system effectively. Scalability considerations should also be considered to accommodate future growth and increasing user demands.

4.4.2 Software Requirement

The software requirements for an emotion-based music recommendation system encompass the necessary software components, tools, frameworks, and dependencies required to develop, deploy, and operate the system effectively. Since the project involves various methods of implementation such as speech emotion recognition, API integration, text analysis, emotion classification, and web application development, the software requirements should support these functionalities seamlessly. Here are the software requirements for the project:

- **Operating System (OS)**:
    - The software should be compatible with major operating systems such as Windows, macOS, and Linux to ensure broad accessibility.
    - Development and deployment environments should support the chosen operating system(s) for coding, testing, and running the system.
- **Development Tools**:
    - Programming Languages:
        - Python: Python is commonly used for machine learning, natural language processing, and web development, making it suitable for implementing speech emotion recognition, text analysis, and web application components.
        - JavaScript: JavaScript is essential for frontend web development, enabling interactive user interfaces and dynamic content.
    - Integrated Development Environments (IDEs):
        - PyCharm, VS Code, or Jupyter Notebook for Python development.
        - Visual Studio Code or Atom for JavaScript and web development.
    - Version Control Systems:
        - Git for managing source code and collaboration among developers.
- **Library and Frameworks**:
    - Speech Emotion Recognition:

        - TensorFlow or PyTorch for building and training machine learning models for speech emotion recognition.
    - Text Analysis:
        - Natural Language Toolkit (NLTK) for text processing, sentiment analysis, and semantic analysis.
    - Web Development:
        - Flask for backend web development in Python.
        - Vite and React.js for frontend web development in JavaScript.
    - API Integration:
        - Python libraries for interfacing with the Spotify API and Musix-match API, such as spotipy for Spotify and requests for HTTP requests.
    - Data Analysis and Visualization:
        - Pandas, NumPy, and Matplotlib for data manipulation and visualization.
        - Seaborn for advanced data visualization.

- **Web Servers/Application Servers**:
    - Web servers such as Apache HTTP Server or Nginx for serving static web content and handling web requests.
    - Application servers like Gunicorn for hosting Python web applications built with Flask.

- **Third-Party APIs and SDKs**:
    - Spotify API and Musix-match API for accessing music metadata, lyrics, and user preferences.
    - Any necessary SDKs or libraries provided by the APIs for authentication, data retrieval, and integration with the system.

- **Deployment Environment**:
    - Cloud Platforms: AWS for deploying and hosting the web application and backend services.
- **Security Software**:
    - CORS (Cross Origin Resource Sharing) provides SSL/TLS certificates for securing communication between the web application and clients.
    - Secure coding practices to prevent common vulnerabilities cross-site scripting (XSS), and cross-site request forgery (CSRF).

These software requirements provide a foundation for developing, deploying, and operating an emotion-based music recommendation system that incorporates speech emotion recognition, text analysis, API integration, and web application components seamlessly. They support the implementation of various functionalities and ensure compatibility, performance, security, and reliability throughout the project lifecycle.

4.4.3 Server-Hosting Requirement

Server-hosting requirements for the "Emotion-based Music Recommendation System" involve specifying the infrastructure and hosting environment needed to deploy and operate the system effectively. Since the project involves various methods of implementation such as speech emotion recognition, API integration, text analysis, emotion classification, and web application development, the server-hosting requirements should support these functionalities seamlessly. Here are the server-hosting requirements for the project:

Operating System: Linux (e.g., Ubuntu, CentOS) for stability and compatibility with Python, Flask, and machine learning libraries.

Python Version: Python 3.10 or higher to leverage the latest features and security updates, especially for machine learning frameworks.

Web Server: Nginx or Apache for serving static files and acting as a reverse proxy to Flask.

Virtual Environment: Using virtual environment to isolate dependencies for the Flask app and machine learning libraries.

Python Packages: Install Flask, Flask-CORS (for Cross-Origin Resource Sharing), and necessary machine learning libraries (e.g., TensorFlow, PyTorch, scikit-learn).

Flask-CORS Configuration: Configure Flask-CORS to allow cross-origin requests based on your application's needs.

Security: Configure firewall rules (e.g., ufw on Linux) to allow only necessary ports (e.g., 80, 443 for HTTP/HTTPS).

SSL Certificate: Obtain and configure an SSL certificate (e.g., Let's Encrypt) for secure HTTPS connections.

Monitoring: Set up monitoring tools (e.g., Prometheus, Grafana) to track server performance and application health, including machine learning model performance metrics.

Deployment: Using a deployment tool like Gunicorn and uWSGI to serve the Flask app and manage worker processes, ensuring scalability for machine learning workloads.

Logging: Configure logging to capture application logs and errors for troubleshooting and monitoring purposes, including logging model training and inference activities.

Backup: Implement regular backups of the application files, including trained machine learning models, to prevent data loss in case of failures.

Scalability (Optional): Consider using containerization (e.g., Docker) and orchestration tools (e.g., Kubernetes) for scalable deployments, especially for machine learning applications requiring additional computational resources.

Environment Variables: Using environment variables for sensitive configurations (e.g., API keys, machine learning model paths) instead of hardcoding them in the application code.

CPU and RAM: Ensure the server meets a minimum of Intel Pentium 4 CPU and 4GB RAM for smooth operation, considering the computational requirements of machine learning tasks.

Storage Requirement: A minimum of 4 GB storage is required to host our web-application.

Docker (Optional): To use the Docker image, Docker Engine must be installed on the system and a storage of minimum 8 GB would be required.

GPU Acceleration (Optional): Utilizing a GPU (Graphics Processing Unit) can significantly benefit machine learning tasks, especially for deep learning models. Speed up training times and inference speeds. Enable handling of larger datasets and complex models efficiently. Improve real-time predictions and reduce latency. Optimize resource utilization and reduce computational costs.

CHAPTER 5

SYSTEM ANALYSIS

5.1 STUDY OF CURRENT SYSTEM

5.2 PROBLEMS IN CURRENT SYSTEM

5.3 REQUIREMENT OF NEW SYSTEM

5.4 PROCESS MODEL

5.5 FEASIBILITY STUDY

5.5.1 TECHNICAL FEASIBILITY

5.5.2 OPERATIONAL FEASIBILITY

5.5.3 ECONOMICAL FEASIBILITY

5.5.4 SCHEDULE FEASIBILITY

5.6 FEATURES OF OUR SYSTEM

5.1 STUDY OF CURRENT SYSTEM

Prior to the development of the Emotion-based Music Recommendation System (EMRS), an in-depth analysis of existing systems and approaches in the field of music recommendation and emotion recognition is essential. This study aims to assess the strengths, weaknesses, and gaps in current systems to inform the design and implementation of the proposed EMRS.

(1) Existing Music Recommendation Systems:

- **Collaborative Filtering Systems**: Traditional recommendation systems often rely on collaborative filtering techniques to suggest music based on user preferences and historical behaviour. While effective to some extent, these systems may not adequately capture the emotional context of music consumption.
- **Content-based Recommendation Systems:** Some systems utilize content-based approaches, analysing song features such as genre, tempo, and instrumentation to make recommendations. However, these systems often overlook the emotional aspect of music, focusing primarily on audio features.
- **Hybrid Recommendation Systems:** Hybrid systems combine collaborative filtering and content-based approaches to provide more accurate recommendations. However, there is still room for improvement in incorporating emotion-based features into these hybrid models.

(2) Emotion Recognition Systems:

- **Speech Emotion Recognition (SER):** SER systems analyze vocal cues to infer emotional states. While SER has shown promise in emotion detection, its integration into music recommendation systems is relatively unexplored.
- **Emotion Detection from Music Audio:** Some research has been done on extracting emotional cues directly from music audio signals. However, accurately capturing complex emotional nuances remains a challenge.

(3) Challenges and Limitations:

- **Data Sparsity:** Many recommendation systems suffer from data sparsity issues, particularly concerning emotional data. Emotion annotations for music are scarce, making it challenging to train accurate emotion-based recommendation models.

- **Subjectivity and Context**: Emotions are subjective and context-dependent, posing challenges for accurately capturing and interpreting user emotions in a music recommendation context.
- **Evaluation Metrics**: Existing evaluation metrics for music recommendation systems often focus on accuracy and coverage but may not adequately capture the quality of emotional recommendations.

5.2 PROBLEMS IN CURRENT SYSTEM

In the current system of music recommendation, there exists a significant gap in understanding the nuanced emotional preferences of users. Traditional algorithms often rely solely on genre or popularity metrics, neglecting the intricate interplay of emotions that define a listener's experience. This oversight leads to recommendations that may not resonate with users on an emotional level, hindering the potential for deeper engagement and connection with the music. An Emotion-based Music Recommendation System seeks to address this deficiency by leveraging advanced AI techniques to analyze and recommend music based on the complex emotional states of listeners, providing a more tailored and enriching musical experience. The following are the detailed points to describe problems in current system:

(1) Limited Emotion Recognition Integration:

- Existing systems lack seamless integration of emotion recognition technologies, such as Speech Emotion Recognition (SER), into the recommendation process.
- Emotion-based recommendation systems often rely on simplistic models that overlook the complex nuances of user emotions, leading to suboptimal recommendations.

(2) Data Sparsity and Quality:

- Emotion annotation data for music is sparse and of varying quality, hindering the development of accurate emotion-based recommendation models.
- Current systems struggle to capture the diversity of emotional responses to music, resulting in limited effectiveness in providing personalized recommendations.

(3) Subjectivity and Contextual Understanding:

- Emotions are subjective and context-dependent, making it challenging for current systems to accurately interpret and respond to users' emotional cues.
- Existing systems often fail to consider contextual factors such as cultural differences, individual preferences, and situational dynamics, leading to mismatches between recommended music and users' emotional states.

(4) Limited Multi-modal Analysis:

- Many current systems primarily focus on audio features for music recommendation, overlooking other modalities such as lyrics, semantic analysis, and user interactions.

- The absence of comprehensive multi-modal analysis results in a shallow understanding of the emotional content of music, leading to generic and less relevant recommendations.

(5) Scalability and Performance:

- Some existing systems may struggle with scalability issues, particularly when dealing with large datasets or real-time processing of user inputs.
- Performance bottlenecks, such as slow recommendation response times or high computational resource requirements, may limit the usability and scalability of the system.

(6) Lack of User Engagement and Feedback Mechanisms:

- Current systems often lack robust mechanisms for user engagement and feedback, limiting opportunities for users to provide input on recommended music.
- The absence of feedback loops hampers the system's ability to adapt and improve recommendations over time based on user preferences and emotional responses.

(7) Evaluation and Validation Challenges:

- Evaluating the effectiveness of emotion-based recommendation systems poses significant challenges due to the subjective nature of emotional experiences.
- Existing evaluation metrics may not adequately capture the quality and relevance of recommended music in relation to users' emotional states, making it difficult to assess system performance accurately.

5.3 REQUIREMENT OF NEW SYSTEM

The requirement for a new Emotion-based Music Recommendation System arises from the limitations of existing approaches, which fail to capture the emotional nuances crucial for personalized music experiences. This innovative system aims to bridge the gap by employing cutting-edge AI algorithms to understand and recommend music based on users' emotional states. By prioritizing emotional resonance, the new system promises to deliver tailored music recommendations that deeply resonate with users, fostering stronger connections and enhancing overall listening satisfaction. The following are the points to describe the requirement of new system:

(1) Emotion Recognition Integration: The system must seamlessly integrate emotion recognition technologies, such as Speech Emotion Recognition (SER), to accurately capture users' emotional states from vocal cues.

(2) Multi-modal Analysis: The system should employ multi-modal analysis, including audio features, lyrics, semantics, and user interactions, to comprehensively understand the emotional content of music.

(3) Data Acquisition and Quality: The system must acquire and curate high-quality emotion annotation data for training and refining emotion-based recommendation models. Mechanisms for continuously updating and expanding the emotion annotation dataset should be incorporated to improve recommendation accuracy over time.

(4) Personalization and Contextual Understanding: The system should prioritize personalization, considering individual preferences, cultural differences, and situational dynamics when recommending music based on users' emotional states. Contextual understanding capabilities should be integrated to adapt recommendations to users' current contexts and environments.

(5) Scalability and Performance: The system should be designed to handle large datasets and real-time processing of user inputs efficiently, ensuring scalability and optimal performance even under heavy loads. Performance benchmarks and scalability tests should be conducted to assess the system's capacity and identify potential bottlenecks.

(6) User Engagement: The system must include user-friendly interfaces that allow users to provide feedback on recommended music, enabling continuous refinement of recommendations based on user responses. Interactive features, such as mood input options, should be provided to enhance user engagement and satisfaction.

(7) Evaluation and Validation: The system should incorporate robust evaluation metrics that assess the quality, relevance, and effectiveness of recommended music in relation to users' emotional states. Mechanisms for validating the system's performance through user studies should be established to iteratively improve recommendation accuracy and user satisfaction.

(8) Interoperability and Integration: The system should be interoperable with various music streaming platforms and APIs, facilitating seamless integration with existing music services and databases. Compatibility with different devices and operating systems should be ensured to maximize accessibility and usability for a wide range of users.

In conclusion, addressing the identified problems and fulfilling the requirements outlined above is crucial for the successful development and deployment of the Emotion-based Music Recommendation System, ensuring a personalized, context-aware, and engaging music recommendation experience for users.

5.4 PROCESS MODEL

Rapid Application Development (RAD) is a suitable process model for the development of the Emotion-based Music Recommended System due to its emphasis on iterative development, user involvement, and rapid prototyping. This model allows for quick iterations and adjustments based on user feedback, making it well-suited for a project focused on delivering personalized user experiences such as EMRS.

Key Phases in RAD:

(1) Requirements Planning: In this phase, the project team collaborates with stakeholders to gather and prioritize requirements for the Emotion-based Music Recommended System. Emphasis is placed on identifying the key features and functionalities related to emotion recognition, music recommendation.

(2) User Design: During this phase, rapid prototypes or mock-ups of the Emotion-based Music Recommended System, user interface are created based on the gathered requirements. User involvement is crucial to ensure that the interface design effectively captures users' emotional cues and preferences.

(3) Construction: In the construction phase, development of the Emotion-based Music Recommended System begins with a focus on implementing core features and functionalities. Iterative development allows for quick iterations and adjustments based on beta testing feedback, ensuring that the system aligns with user expectations.

(4) Cutover: The cutover phase involves transitioning the Emotion-based Music Recommended System from development to production. This may include data migration, system testing, user training, and deployment of the finalized system. User involvement continues during this phase to address any final adjustments or refinements.

Advantages of RAD for Emotion-based Music Recommended System:

(1) Iterative Development: RAD facilitates iterative development cycles, allowing the project team to quickly prototype and refine features related to emotion recognition, music recommendation.

(2) User Involvement: RAD encourages active participation from users throughout the development process, ensuring that the Emotion-based Music Recommended System meets their needs and preferences effectively.

(3) Quick Prototyping: Rapid prototyping enables the project team to quickly validate design concepts and gather feedback from users, leading to more informed decision-making and a better-aligned final product.

(4) Flexibility: RAD offers flexibility to adapt to changing requirements and emerging technologies, allowing the Emotion-based Music Recommended System to evolve over time to meet the evolving needs of users and stakeholders.

(5) Reduced Time to Market: By emphasizing rapid development and quick iterations, RAD helps expedite the development process, reducing time to market for the EMRS and enabling quicker user adoption.

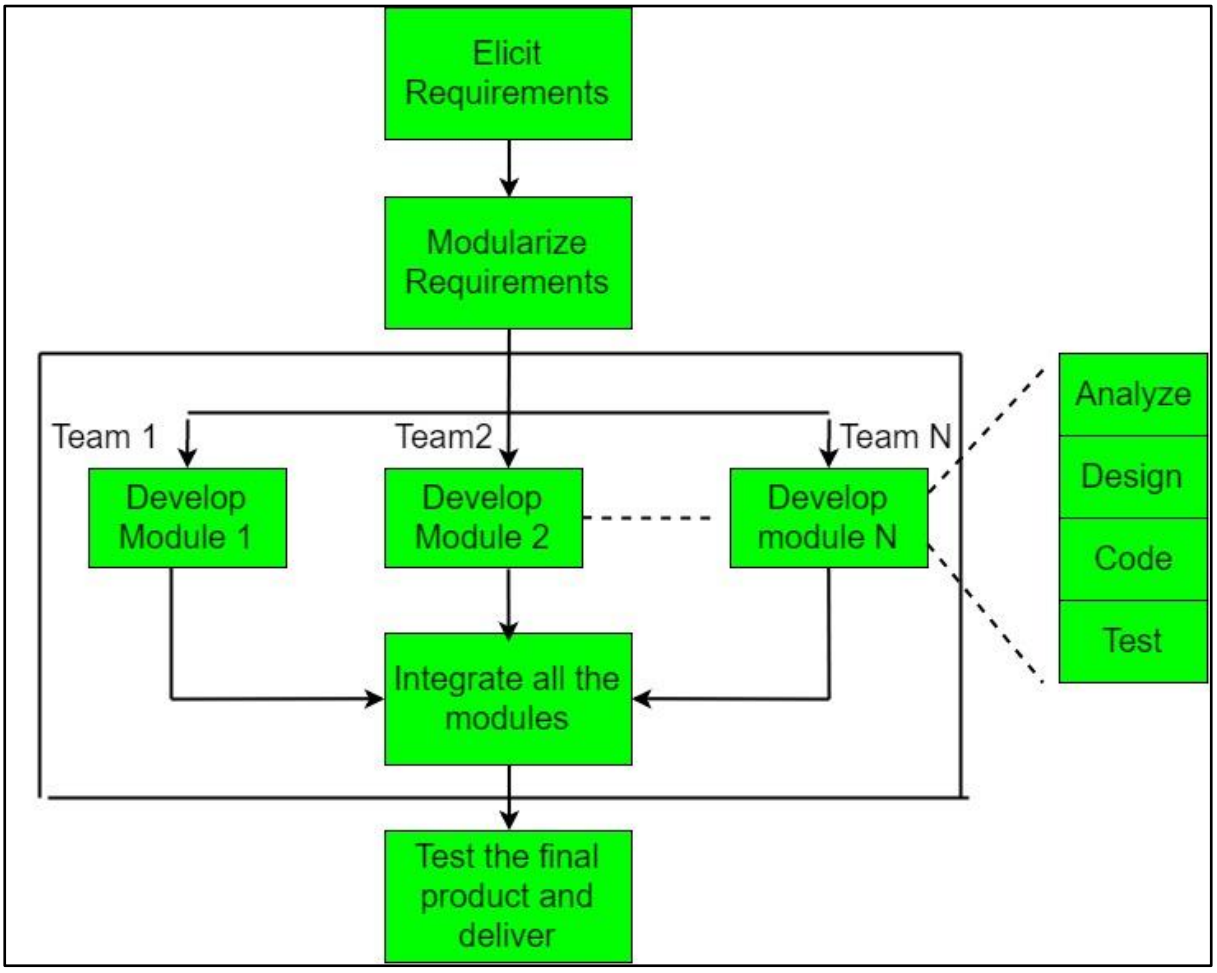


Fig. 5.1 RAD Model.

Considerations for RAD Implementation:

(1) Clear Communication: Effective communication between the project team, stakeholders, and end-users is essential for the success of RAD. Regular feedback loops

and collaborative decision-making processes should be established to ensure alignment with user expectations.

(2) Risk Management: Rapid development cycles may introduce risks such as scope creep, resource constraints, and technical challenges. Proactive risk management strategies should be in place to mitigate these risks and maintain project momentum.

(3) Resource Allocation: Adequate resources, including skilled personnel, tools, and technologies, should be allocated to support the rapid development pace inherent in RAD. Collaboration and coordination among team members are essential to maximize productivity and efficiency.

(4) Quality Assurance: Despite the rapid development pace, quality assurance processes should not be compromised. Continuous testing, validation, and verification activities should be integrated into each iteration to ensure the reliability, performance, and usability of the Emotion-based Music Recommended System.

Adopting RAD as the process model for the development of the Emotion-based Music Recommended System offers several advantages, including iterative development, user involvement, quick prototyping, flexibility, and reduced time to market. By following RAD principles and best practices, the project team can effectively deliver a personalized, user-centric Emotion-based Music Recommended System that effectively captures and responds to users' emotional cues and preferences.

5.5 FEASIBILITY STUDY

The Emotion-based Music Recommendation System aims to leverage advancements in speech emotion recognition, natural language processing, and music analysis techniques to provide personalized music recommendations based on users' emotional states. This system will utilize various APIs such as Spotify, Lyrics.ovh and Musix-match to gather music data and lyrics, perform sentiment analysis, and generate emotion-based playlists. Additionally, the system will include a web application interface for user interaction and facilitate music generation capabilities.

5.5.1 Technical Feasibility

The technical feasibility of the Emotion-based Music Recommendation System relies on the availability of appropriate technologies, APIs, and computational resources, tailored for the chosen frontend and backend technologies. Here's an adapted overview:

1. Speech Emotion Recognition (SER):

- Feasibility: Utilizing ViteJs, ReactJs, and TypeScriptJs for the frontend, and Python with Flask and Gunicorn for the backend, SER implementation remains feasible.
- Implementation: Integration of SER libraries compatible with Python, such as librosa for audio feature extraction and TensorFlow/Keras for deep learning models, ensures seamless integration within the Flask backend.
- Computational Requirements: Efficient SER models compatible with Flask endpoints and suitable for deployment with Gunicorn ensure optimal resource utilization.

2. API Integration:

- Feasibility: ViteJs, ReactJs, and TypeScriptJs offer robust support for integrating APIs, ensuring compatibility with Spotify, Lyrics.ovh, and Musix-match APIs.
- Implementation: Utilizing Axios or Fetch API within React components, along with Python requests library in Flask routes, enables smooth integration of APIs for data retrieval.

- Rate Limiting and Quotas: Monitoring API usage within Flask middleware and implementing client-side rate limiting mechanisms in React ensures compliance with API restrictions.

3. Lyrics Analysis:

- Feasibility: Leveraging Python libraries like NLTK or spaCy for NLP tasks and TypeScript for frontend logic, lyrics sentiment and semantics analysis remains feasible.
- Implementation: Integrating NLP pipelines within Flask routes for lyrics analysis and TypeScript functions within React components for displaying results ensures efficient processing and rendering.
- Computational Resources: Optimizing NLP algorithms and utilizing asynchronous processing techniques within Flask routes mitigate computational overhead.

4. Emotion-based Music Classification:

- Feasibility: Implementing classification models with Python's scikit-learn or TensorFlow/Keras for backend and TypeScript for frontend ensures seamless integration within the system.
- Implementation: Developing classification endpoints with Flask and serving predictions to React components via API requests ensures real-time classification of emotions and music recommendations.
- Training Data: Accessing and preprocessing labeled datasets within Flask routes and serving trained models for inference guarantees the feasibility of emotion-based music classification.

5. Music Provider and Mappings:

- Feasibility: Integrating APIs from music streaming platforms within React components and Flask routes ensures efficient retrieval and mapping of music metadata.
- Implementation: Utilizing Python libraries like Spotipy for Spotify API integration and TypeScript for frontend logic enables seamless navigation and presentation of curated playlists and recommendations.

- Data Consistency: Implementing data validation and synchronization mechanisms within Flask routes and React components ensures consistent and accurate music metadata and user preferences.

6. Web Application Development:

- Feasibility: Developing the frontend with ViteJs, ReactJs, and TypeScriptJs and backend with Python, Flask, and Gunicorn ensures a robust and scalable web application.
- Implementation: Utilizing React Router for client-side routing, Flask for serving API endpoints, and Gunicorn for production-ready deployment ensures efficient development and deployment of the web application.
- Scalability: Designing the application with component-based architecture and microservices ensures scalability and flexibility to accommodate increasing user traffic and feature enhancements.

7. Music Generation:

- Feasibility: Integrating music generation algorithms within Flask routes using Python libraries like Magenta and serving generated music to React components ensures seamless music generation capabilities.
- Implementation: Leveraging TypeScript for frontend logic and integrating music generation endpoints within Flask routes enables real-time customization and rendering of generated music.
- Quality Control: Implementing user feedback mechanisms within React components and Flask routes ensures iterative refinement of generated music quality and alignment with user preferences.

5.5.2 Operational Feasibility

The operational feasibility of the Emotion-based Music Recommendation System encompasses aspects related to system maintenance, user adoption, and stakeholder satisfaction. Here's an in-depth exploration of operational feasibility considerations:

1. System Maintenance:

- **Feasibility Assessment:** The system's maintenance involves regular updates, bug fixes, and enhancements to ensure its smooth operation and relevance over time.
- **Implementation:** Employing a version control system (e.g., Git) for code management and continuous integration/continuous deployment (CI/CD) pipelines streamlines the process of deploying updates and fixes.
- **Team Collaboration:** Facilitating effective collaboration among developers, data scientists, and domain experts ensures timely resolution of issues and efficient system maintenance.

2. User Adoption:

- **Feasibility Assessment:** User adoption is crucial for the system's success, necessitating strategies to attract and retain users.
- **Implementation:** Conducting user surveys, interviews, and usability testing during the development phase helps understand user preferences and pain points, informing feature prioritization and design decisions.
- **User Engagement:** Implementing features such as personalized recommendations, social sharing, and interactive user interfaces enhances user engagement and encourages return visits.

3. Resource Management:

- **Feasibility Assessment:** Effective resource management, including human resources, financial resources, and infrastructure, is critical for project success.
- **Implementation:** Utilizing project management tools (e.g., Jira, Trello) for task assignment, progress tracking, and resource allocation enhances transparency and accountability.
- **Budget Planning:** Conducting thorough budget planning and monitoring ensures optimal utilization of financial resources and prevents cost overruns.
- **Scalability Considerations:** Anticipating future growth and scalability requirements allows for proactive resource allocation and infrastructure planning,

ensuring the system can accommodate increasing user demands and feature enhancements.

4. Risk Management:

- **Feasibility Assessment:** Identifying and mitigating potential risks and challenges early in the project lifecycle is essential for minimizing disruptions and ensuring project success.
- **Implementation:** Conducting risk assessments and developing risk mitigation strategies for potential technical, operational, and market-related risks enhances project resilience.
- **Contingency Planning:** Developing contingency plans for potential disruptions, such as API changes, server outages, or data breaches, ensures rapid response and recovery to minimize impact on system operation.

5.5.3 Economical Feasibility

The economic feasibility of the Emotion-based Music Recommendation System involves assessing the project's financial viability, including initial investments, ongoing operational costs, revenue generation potential, and return on investment. Here's a detailed examination of the economic aspects:

1. Initial Investments:

- **Feasibility Assessment:** The project requires initial investments in technology infrastructure, software development, API usage fees, and personnel.
- **Implementation:** Conducting a thorough cost-benefit analysis to estimate the initial investment required for hardware, software licenses, API subscriptions, and human resources ensures prudent financial planning.
- **Budget Allocation:** Allocating resources efficiently based on project priorities and timelines minimizes upfront costs and maximizes the project's value proposition.

2. Operational Costs:

- **Feasibility Assessment:** Ongoing operational costs include server hosting, API usage fees, personnel salaries, marketing expenses, and maintenance costs.
- **Implementation:** Estimating recurring expenses accurately and monitoring them closely ensures financial sustainability throughout the project lifecycle.
- **Cost Optimization:** Implementing cost optimization strategies, such as resource pooling, serverless architecture, and efficient API usage, helps minimize operational expenses without compromising system performance.

3. Revenue Generation Potential:

- **Feasibility Assessment:** The project's revenue generation potential depends on factors such as user adoption, monetization strategies, and market demand for personalized music recommendations.
- **Implementation:** Exploring various revenue streams, including subscription models, advertising, premium features, and affiliate partnerships, diversifies income sources and enhances revenue generation potential.

4. Return on Investment (ROI):

- **Feasibility Assessment:** Calculating the projected ROI helps assess the project's financial viability and attractiveness to potential investors.
- **Implementation:** Estimating the project's expected revenue and comparing it to the initial investment yields the ROI ratio, providing insights into the project's profitability.
- **Risk-adjusted ROI:** Incorporating risk factors and uncertainties into ROI calculations ensures a more accurate assessment of the project's financial performance and risk profile.

5. Scalability and Growth Potential:

- **Feasibility Assessment:** Evaluating the project's scalability and growth potential determines its ability to adapt to increasing user demand and market opportunities.

- **Implementation:** Designing the system with scalability in mind, including flexible architecture, modular components, and cloud-based infrastructure, enables seamless expansion to accommodate growing user base and feature enhancements.

6. Cost-Benefit Analysis:

- **Feasibility Assessment:** Conducting a comprehensive cost-benefit analysis compares the project's expected benefits against its costs, providing insights into its economic feasibility.
- **Implementation:** Quantifying both tangible and intangible benefits, such as improved user experience, increased user engagement, and brand value, alongside costs ensures a holistic evaluation of the project's economic viability.

The Emotion-based Music Recommendation System demonstrates promising economic feasibility, with prudent financial planning, diversified revenue streams, and scalability strategies in place. By estimating initial investments, optimizing operational costs, and projecting ROI, the project aims to deliver sustainable financial returns while providing value to users and stakeholders alike.

5.5.4 Schedule Feasibility

Schedule feasibility involves assessing the project's timeline, milestones, and dependencies to ensure timely completion within the allocated resources. Here's an overview of the schedule feasibility considerations for the Emotion-based Music Recommendation System:

1. Project Timeline:

- **Feasibility Assessment:** Evaluating the project's timeline involves estimating the duration of each phase, from requirements gathering to deployment and maintenance.
- **Implementation:** Creating a detailed project plan with clear timelines for each development phase, including research, development, testing, and deployment, ensures realistic scheduling and resource allocation.

- **Milestone Definition:** Defining key milestones, such as prototype development, API integration, user testing, and production deployment, helps track progress and ensure timely completion of critical tasks.

2. Resource Availability:

- **Feasibility Assessment:** Assessing the availability of human resources, technology infrastructure, and external dependencies impacts the project's schedule feasibility.
- **Implementation:** Identifying and allocating resources effectively, including developers, data scientists, API providers, and testing environments, ensures smooth execution of project tasks within the specified timelines.
- **Contingency Planning:** Developing contingency plans for potential resource constraints, such as hiring additional personnel or utilizing cloud services for scalable infrastructure, mitigates schedule risks and ensures project continuity.

3. Task Dependencies:

- **Feasibility Assessment:** Understanding dependencies between project tasks, such as API integration requiring completion of data preprocessing, influences scheduling decisions and milestone achievement.
- **Implementation:** Creating a comprehensive task dependency matrix and critical path analysis helps identify dependencies and prioritize tasks to minimize delays and optimize project flow.
- **Parallelization of Tasks:** Identifying tasks that can be executed in parallel and leveraging agile methodologies, such as Scrum or Kanban, accelerates project progress and minimizes schedule bottlenecks.

4. Iterative Development:

- **Feasibility Assessment:** Embracing iterative development methodologies allows for flexibility in adapting to changing requirements and feedback throughout the project lifecycle.
- **Implementation:** Adopting RAD development practices, including sprint planning, daily stand-ups, and continuous integration/continuous deployment (CI/CD), enables iterative development cycles and rapid iteration of features.

- **User Feedback Loops:** Incorporating user feedback loops, such as beta testing and user surveys, facilitates continuous improvement and ensures alignment with user expectations, potentially reducing schedule adjustments in later stages.

5. Risk Management:

- **Feasibility Assessment:** Identifying and mitigating potential risks and uncertainties, such as technical challenges, API unavailability, or scope creep, impacts project schedule feasibility.
- **Implementation:** Developing risk mitigation strategies and contingency plans for high-impact risks ensures proactive management of schedule disruptions and minimizes their impact on project timelines.
- **Regular Monitoring:** Monitoring project progress regularly and conducting risk assessments at key milestones allows for early identification of schedule deviations and timely corrective actions.

5.6 FEATURES OF OUR SYSTEM

The Emotion-based Music Recommendation System redefines music discovery by leveraging cutting-edge emotion recognition technology. Through voice input or pre-recorded audio, users receive personalized recommendations aligned with their emotional state, spanning diverse genres. With unique features like generating bespoke compositions and in-depth emotion analysis, this system promises a truly immersive and tailored musical journey, accessible through an intuitive interface designed for seamless interaction. The features of our system are described as below:

(1) Audio Input Options:

- Users can interact with the system by either speaking directly into the interface or providing a pre-recorded audio file.
- This flexibility accommodates various user preferences and situations, ensuring seamless user experience.

(2) Emotion Recognition:

- The system employs advanced emotion recognition algorithms to analyze the user's voice and detect emotional cues.
- Emotions such as happiness, sadness, excitement, relaxation, and more are accurately identified to personalize music recommendations.

(3) Personalized Music Recommendations:

- Based on the detected emotion, the system curates a tailored list of music tracks that align with the user's current emotional state.
- Recommendations encompass a wide range of genres and artists to cater to diverse musical tastes and preferences.

(4) Generated Music:

- In addition to recommending existing songs, the system generates a unique composition tailored to the user's detected emotion.
- This exclusive feature provides users with a novel musical experience, offering a fresh perspective on emotional expression through music.

(5) Emotion Analysis:

- The system provides a comprehensive analysis of the user's detected emotion, utilizing multiple dimensions for a nuanced understanding.
- Analysis includes:
    - **Acoustics Analysis:** Examining vocal tone, pitch, and intensity to gauge emotional intensity and expression.
    - **Text Analysis:** Analyzing spoken or recorded text for linguistic markers indicative of emotional states.
    - **Valence Arousal Analysis:** Assessing the valence (positive/negative) and arousal (intensity) of the detected emotion.
    - **Semantics Analysis:** Evaluating contextual and semantic cues to enhance the accuracy of emotion detection and interpretation.

(6) User Interface and Accessibility:

- The system features an intuitive user interface designed for accessibility across various devices and platforms.
- Clear instructions and prompts facilitate seamless interaction, ensuring users of all backgrounds can easily navigate and utilize the system.

(7) Continuous Improvement:

- Regular updates and enhancements are deployed to optimize system performance and incorporate the latest advancements in emotion recognition and music recommendation technologies.
- Real-world usage and data drive ongoing improvements, ensuring the system remains at the forefront of personalized music recommendation services.

The Emotion-based Music Recommendation System revolutionizes the way users discover and engage with music, harnessing the power of emotion recognition technology to deliver personalized, meaningful musical experiences tailored to individual emotional states and preferences.

CHAPTER 6

DETAIL DESCRIPTION

6.1 MODEL DESCRIPTION

6.2 USER MODULE

6.1 MODEL DESCRIPTION

(1) Audio Speech Emotion Recognition Model:

The Audio Speech Emotion Recognition Model is a cutting-edge system designed to analyze the emotional content of spoken language. By leveraging advanced machine learning algorithms, it accurately detects various emotions such as happiness, sadness, anger, and calmness. In the context of an Emotion-based Music Recommendation System, this model serves as a crucial component.

Dataset:
CREMA-D,
RAVDESS EMOTIONAL SPEECH AUDIO,
TORONTO EMOTION SPEECH SET (TESS),
SURREY AUDIO-VISUAL EXPRESSED EMOTION (SAVEE)

Overall CSV Size:
30656 rows x 2377 columns

Train Test Ratio: 80:20

Model: CNN

Architecture:
model = tf.keras.Sequential([

L.Conv1D(512,kernel_size=5, strides=1,padding='same',
activation='relu',input_shape=(X_train.shape[1],1)),
L.BatchNormalization(),
L.MaxPool1D(pool_size=5,strides=2,padding='same'),
L.Conv1D(512,kernel_size=5,strides=1,padding='same',activation='relu'),
L.BatchNormalization(),
L.MaxPool1D(pool_size=5,strides=2,padding='same'),
Dropout(0.2),

L.Conv1D(256,kernel_size=5,strides=1,padding='same',activation='relu'),
L.BatchNormalization(),
L.MaxPool1D(pool_size=5,strides=2,padding='same'),
L.Conv1D(256,kernel_size=3,strides=1,padding='same',activation='relu'),

```
L.BatchNormalization(),
L.MaxPool1D(pool_size=5,strides=2,padding='same'),
Dropout(0.2),

L.Conv1D(128,kernel_size=3,strides=1,padding='same',activation='relu'),
L.BatchNormalization(),
L.MaxPool1D(pool_size=3,strides=2,padding='same'),
Dropout(0.2),

L.Flatten(),
L.Dense(512,activation='relu'),
L.BatchNormalization(),
L.Dense(4,activation='softmax')

])

model.compile(optimizer='adam',loss='categorical_crossentropy',metrics='accuracy')
```

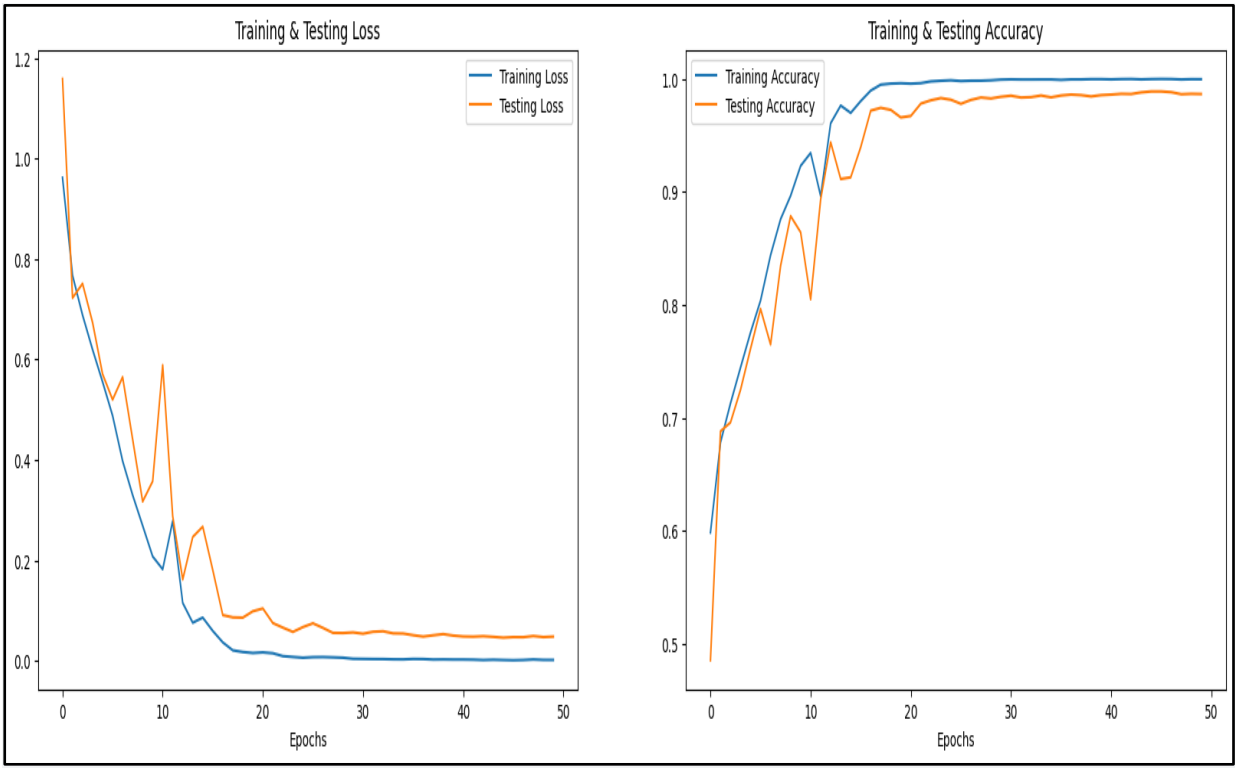


Fig. 6.1 SER Model Train/Test Accuracy and Loss

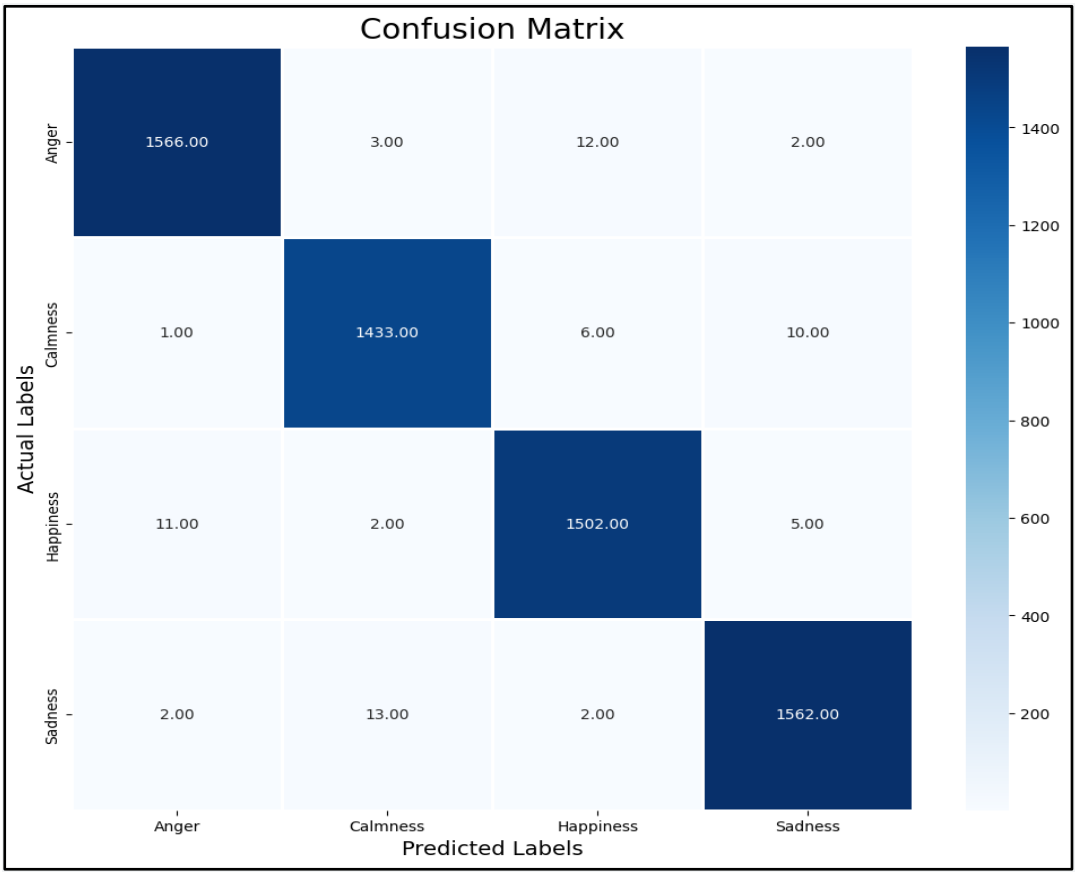


Fig. 6.2 SER Model Heatmap-based Confusion Matrix

	Precision	Recall	F1-Score	Support
Anger	0.99	0.99	0.99	1583
Calmness	0.99	0.99	0.99	1450
Happiness	0.99	0.99	0.99	1520
Sadness	0.99	0.99	0.99	1579
accuracy			0.99	6132
macro avg	0.99	0.99	0.99	6132
weighted avg	0.99	0.99	0.99	6132

Table 6.1 SER Model Classification Report

(2) Text Emotion Recognition Model:

Introducing our Text Emotion Recognition Model tailored specifically for an Emotion-based Music Recommendation System. Using cutting-edge natural language processing techniques, our model accurately detects and analyzes emotions conveyed in textual input, allowing for precise categorization into emotional states such as happiness, sadness, anger, and calmness. Seamlessly integrated with our music recommendation system, it ensures personalized playlists and song suggestions aligned with users' current emotional states, enhancing their listening experience and fostering emotional resonance through music.

Dataset:

Emotion dataset for NLP

Overall Train CSV Size:

16000 rows x 2 columns

Deep Learning Model Pre-Configuration:

max_words = 10000

max_len = 50

embedding_dim = 64

Model: CNN

Architecture:

```
# Branch 1
branch1 = Sequential()
branch1.add(Embedding(max_words, embedding_dim, input_length=max_len))
branch1.add(Conv1D(64, 3, padding='same', activation='relu'))
branch1.add(BatchNormalization())
branch1.add(ReLU())
branch1.add(Dropout(0.5))
branch1.add(GlobalMaxPooling1D())

# Branch 2
branch2 = Sequential()
branch2.add(Embedding(max_words, embedding_dim, input_length=max_len))
branch2.add(Conv1D(64, 3, padding='same', activation='relu'))
branch2.add(BatchNormalization())
branch2.add(ReLU())
```

```
branch2.add(Dropout(0.5))
branch2.add(GlobalMaxPooling1D())

concatenated = Concatenate()([branch1.output, branch2.output])
hid_layer = Dense(128, activation='relu')(concatenated)
dropout = Dropout(0.5)(hid_layer)
output_layer = Dense(4, activation='softmax')(dropout)

model = Model(inputs=[branch1.input, branch2.input], outputs=output_layer)
```

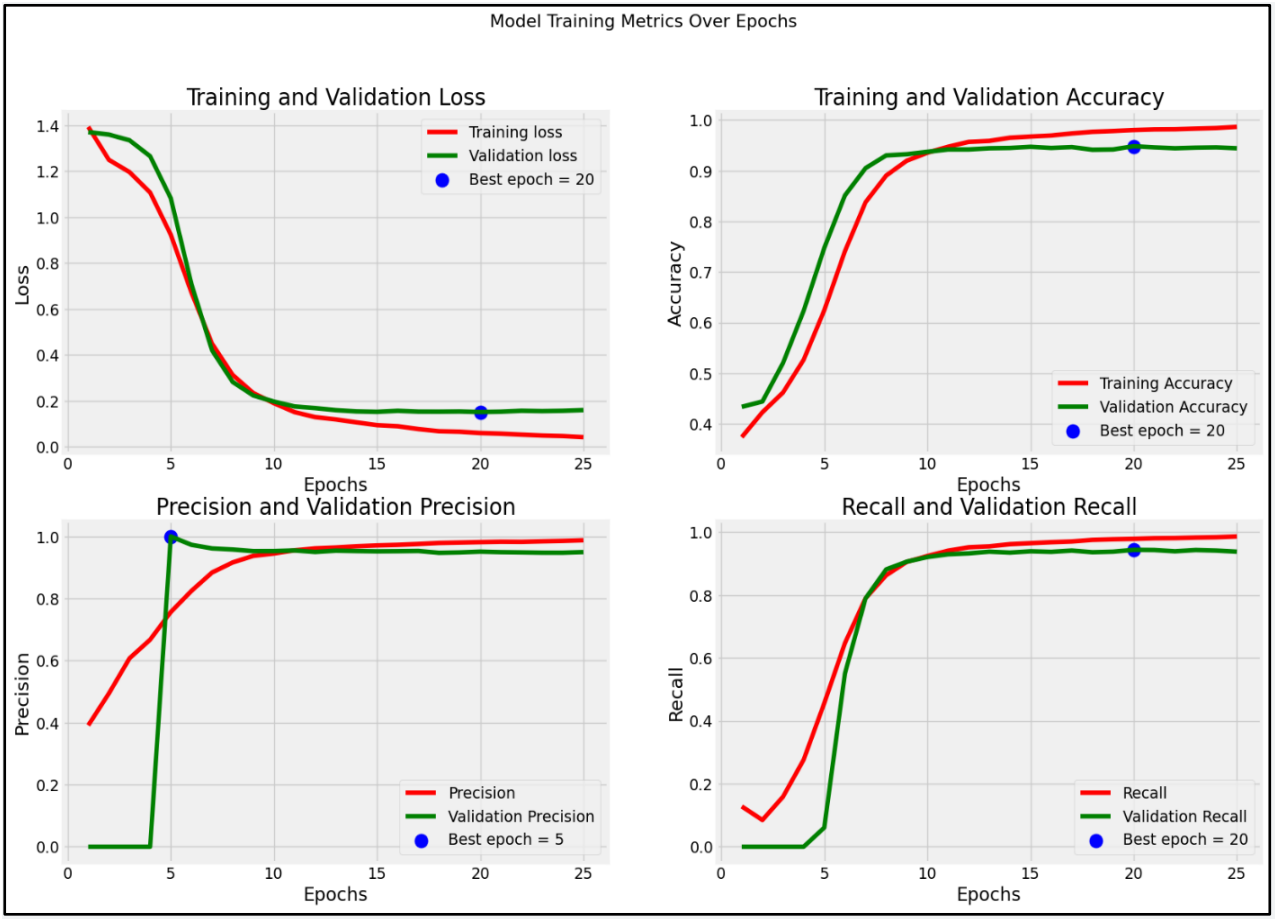


Fig. 6.3 TER Model Train/Test Accuracy & Loss

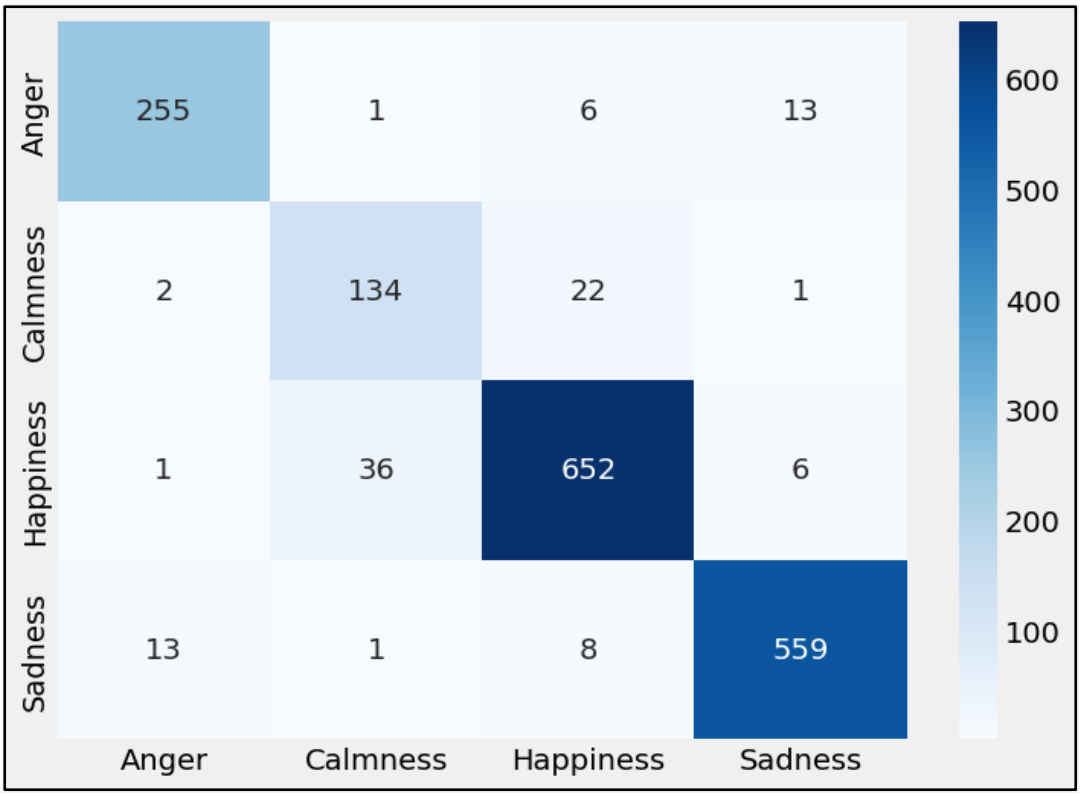


Fig. 6.4 TER Model Heatmap-based Confusion Matrix

	Precision	Recall	F1-Score	Support
Anger	0.94	0.93	0.93	275
Calmness	0.78	0.84	0.81	159
Happiness	0.95	0.94	0.94	695
Sadness	0.97	0.96	0.96	581
accuracy			0.94	1710
macro avg	0.91	0.92	0.91	1710
weighted avg	0.94	0.94	0.94	1710

Table 6.2 TER Model Classification Report

6.2 USER MODULE

The user module in an emotion-based music recommendation system serves to bridge the gap between the user's emotional state and the algorithmic processes responsible for generating personalized music recommendations, ultimately enhancing the user experience by delivering music that resonates with their current emotional state and preferences.

Emotion Detection Interface: A user interface or API that allows users to input or express their emotions, which could be in the form of speech or pre-recorded audio.

Emotion Analysis: Algorithms and methods to analyze the inputted emotions and extract relevant emotional features. This could involve Semantic analysis, Text analysis, Audio and Acoustic analysis and Emotion Recognition from speech or pre-recorded audio.

Recommendation Engine Integration: Integration with the recommendation engine to leverage the extracted emotional features and generating personalized music recommendations.

Here are attached screenshots of our website:

(1) Start-up Page:

When a user visits our website, they will be directed to the start-up page upon arrival.

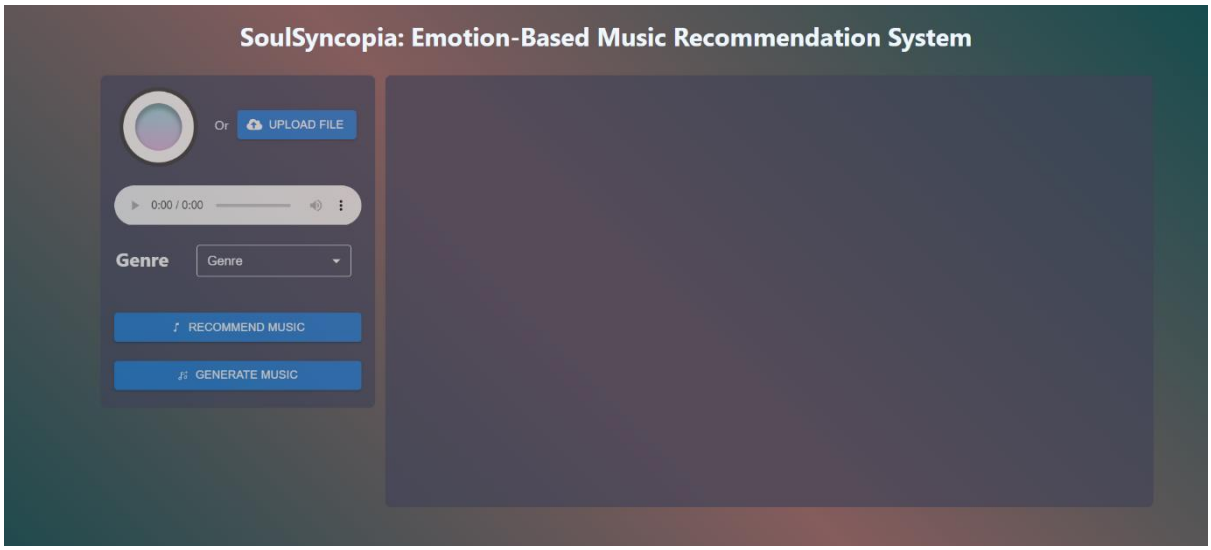


Fig. 6.5 Voice Input for Music Recommendation

Users have the option to either select an audio file from their device or record their own audio directly on the platform.

(2) Music Recommendation:

When the user clicks on the "Recommend Music" button, a list of songs will appear along with their respective similarity percentages, providing tailored suggestions based on their preferences.

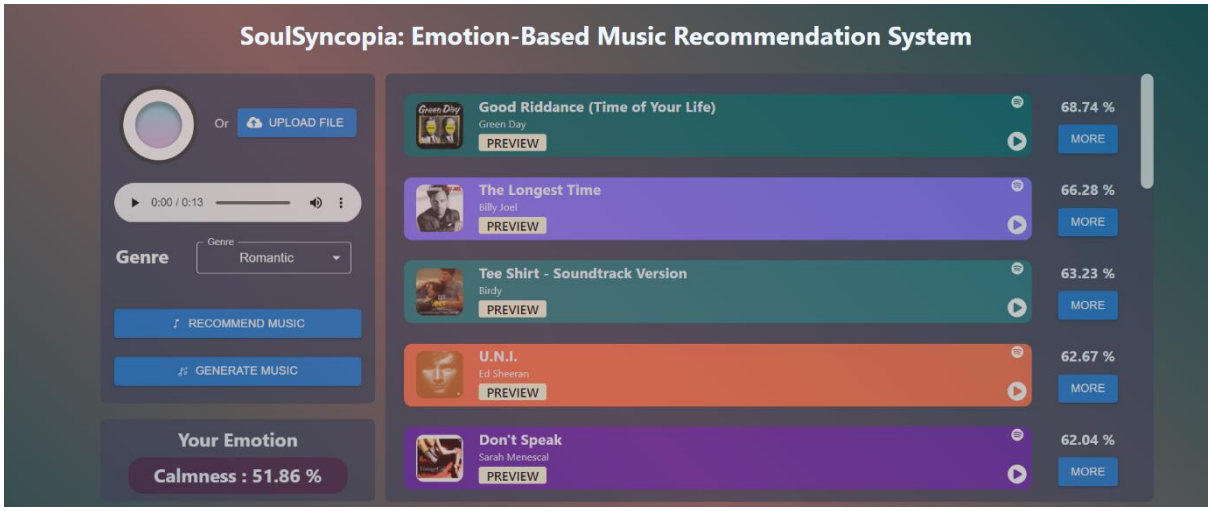


Fig. 6.6 Recommended List of Songs

Users can click on More Info button, to see in-detail analysis of recommended song.

(3) `More` Info Button – Music Analysis:

When the user clicks on the "More" button next to a song, a comprehensive analysis will unfold, encompassing overall analysis, acoustic analysis, text analytics, and semantics analysis, providing deeper insights into the selected track.

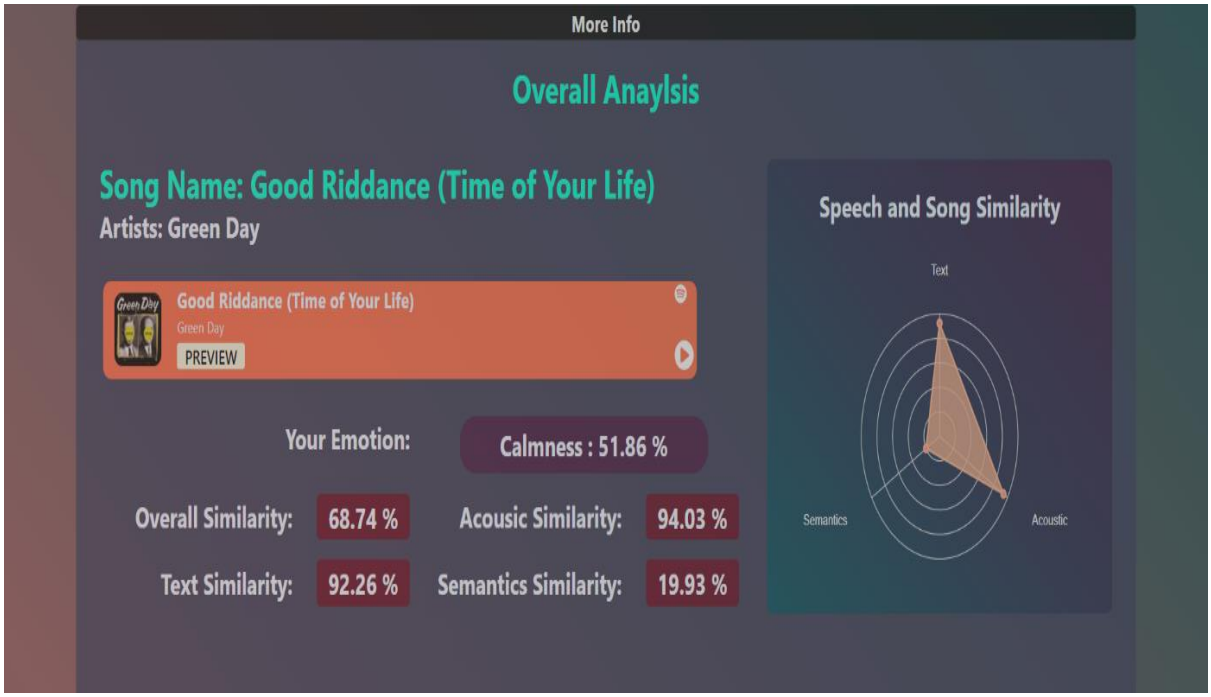


Fig. 6.7 Overall Analysis

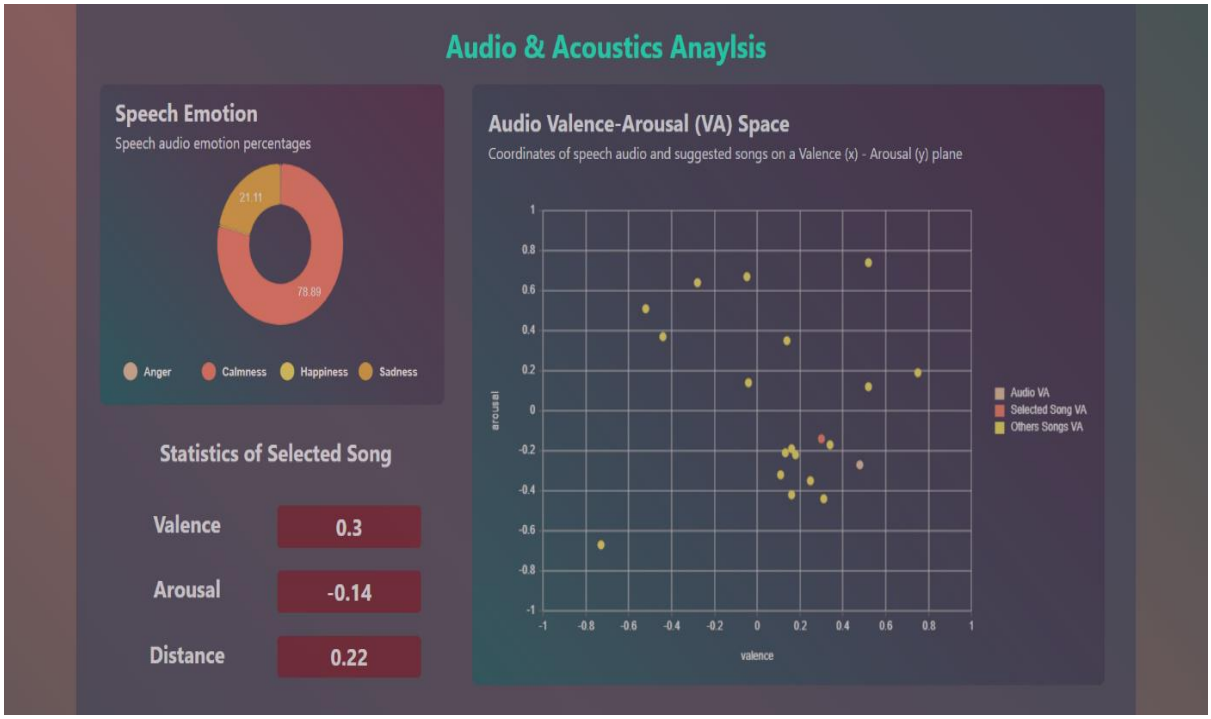


Fig. 6.8 Audio & Acoustics Analysis

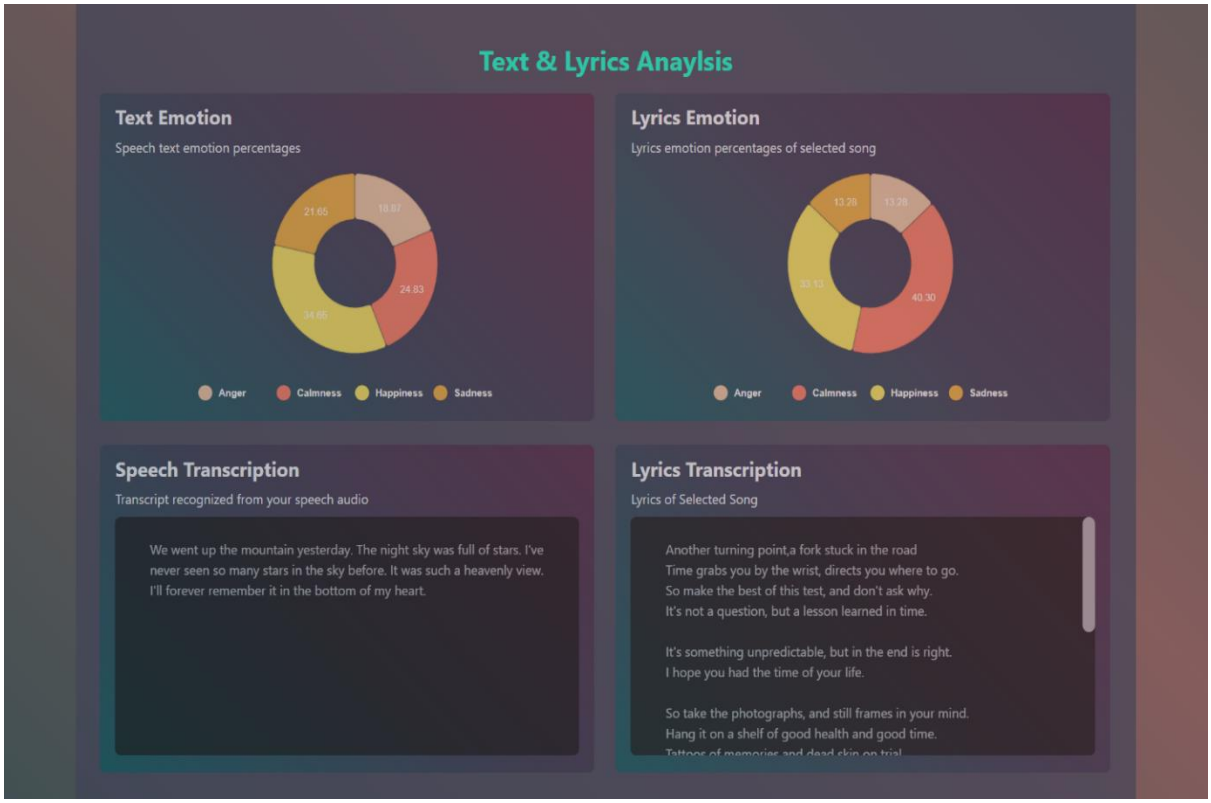


Fig. 6.9 Text & Lyrics Analysis

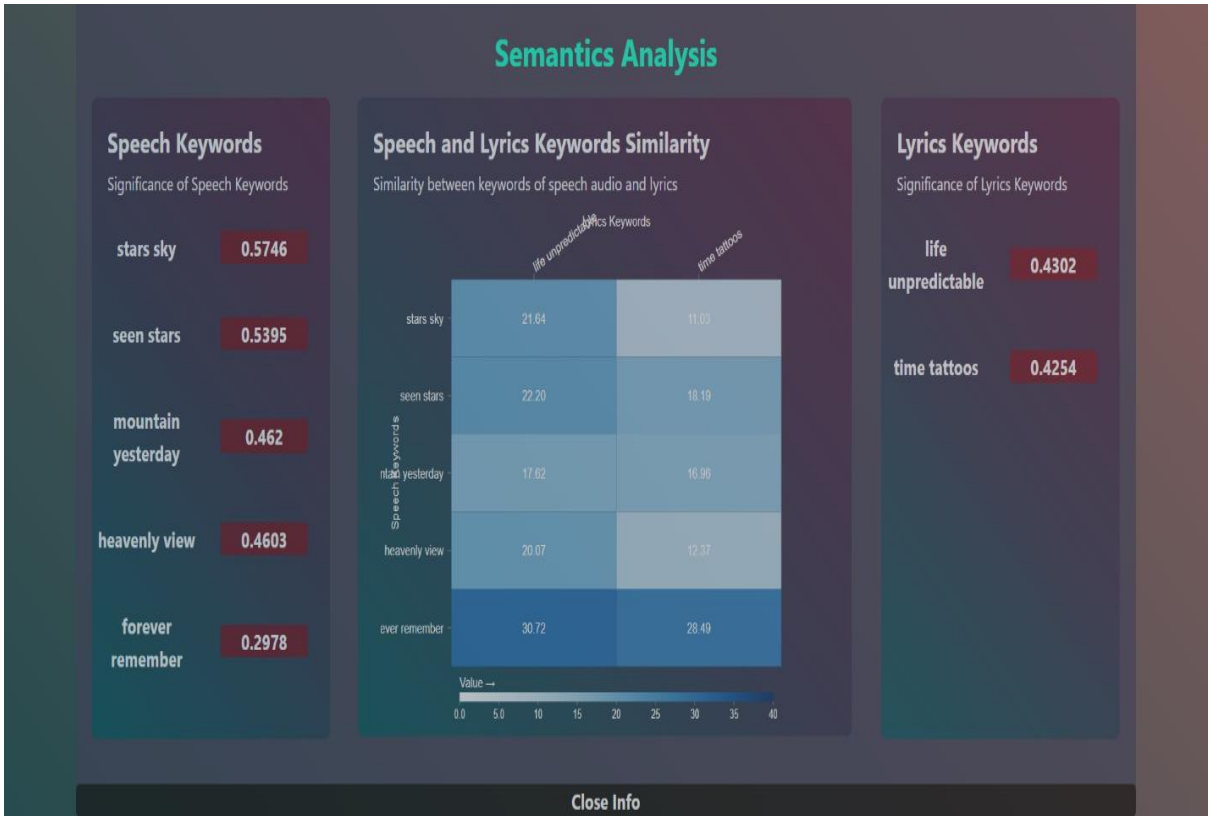


Fig. 6.10 Semantics Analysis

(4) Music Generation:

When the user clicks on the "Music Generation" button, AI-powered music generation will be initiated, crafting compositions based on the recognized emotion.

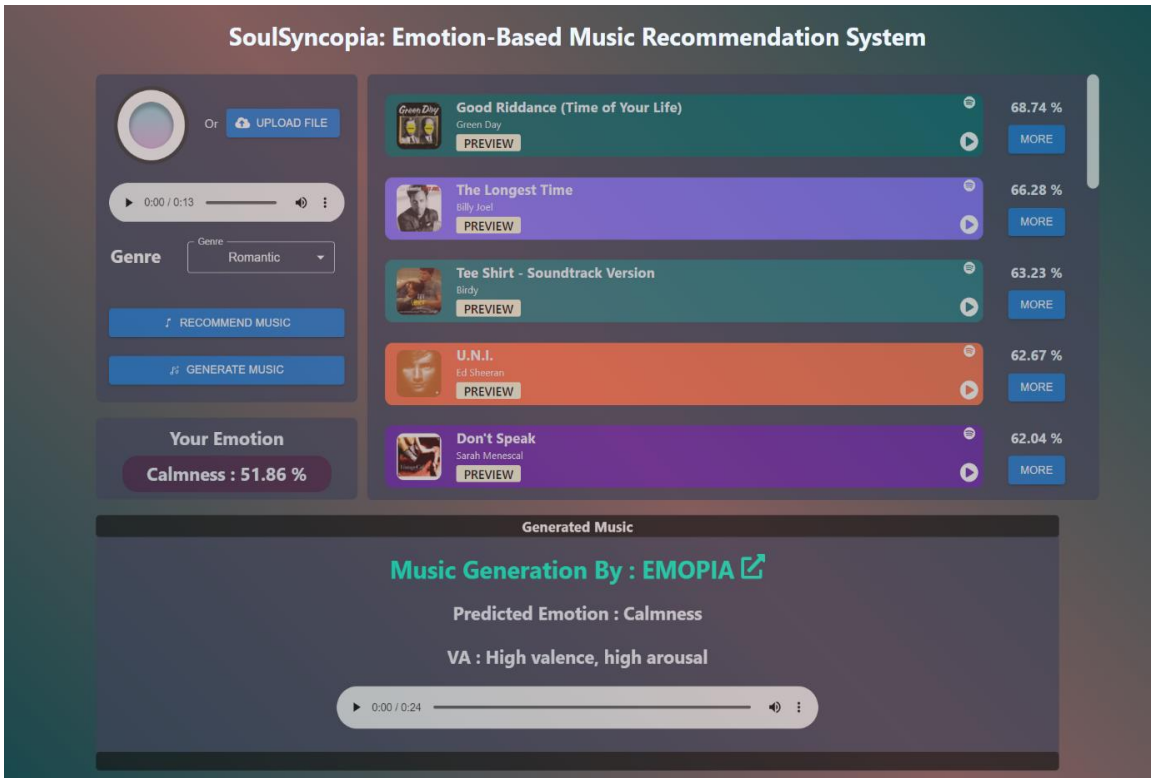


Fig. 6.11 Music Generation

CHAPTER 7

TESTING

7.1 BLACK-BOX TESTING

7.2 WHITE-BOX TESTING

7.3 TEST CASES

7.1 BLACK-BOX TESTING

Black-box testing is a method of software testing where the internal workings or code structure of the system being tested are not known to the tester. Instead, the tester focuses on the inputs and outputs of the software without knowing how it achieves those results. The goal is to test the functionality of the software from an external perspective, simulating how end-users would interact with it.

Examples of black-box testing techniques include:

(1) Equivalence Partitioning: Test cases are designed to cover different partitions or groups of input data that are expected to produce similar results.

(2) Boundary Value Analysis: Test cases are created to evaluate the behaviour of the system at the boundaries or extremes of input ranges.

(3) Decision Table Testing: Test cases are derived from a decision table that specifies combinations of inputs and corresponding actions or outcomes.

(4) State Transition Testing: Test cases are designed to verify the behaviour of the system as it transitions between different states or modes.

(5) Use Case Testing: Test cases are based on the functional requirements or use cases of the software, focusing on validating scenarios from the user's perspective.

Now, let's apply black-box testing to the project topic "Emotion-based Music Recommendation System" and its methods of implementation:

(1) Speech Emotion Recognition: Test the system's ability to accurately recognize and classify emotions from speech inputs without knowing the underlying algorithms used for recognition.

(2) Spotify API and Musix-match API Integration: Verify that the system can successfully integrate with Spotify and Musix-match APIs to fetch music data and lyrics, respectively, without understanding the internal implementation details of the APIs.

(3) Lyrics Text Analysis: Test the system's capability to analyze the sentiment and semantics of lyrics without knowing the specific algorithms used for analysis.

(4) **Valence Arousal Analysis**: Evaluate the accuracy of the system in determining the emotional valence and arousal of music tracks without knowing the intricacies of the analysis techniques employed.

(5) **Emotion Music Classification**: Validate the system's ability to classify music tracks based on emotional characteristics without needing to understand the underlying classification algorithms.

(6) **Music Provider and Mappings**: Ensure that the system correctly maps emotions to appropriate music tracks from the provided music providers without knowing the internal mapping mechanisms.

(7) **Web App Testing**: Test the functionality of the web application interface, including user interactions and the presentation of recommended music, without needing knowledge of the underlying code implementation.

In each of these tests, the focus is on the inputs, outputs, and behaviour of the system without requiring knowledge of its internal workings. This approach allows for comprehensive testing while maintaining independence from the system's implementation details.

7.2 WHITE-BOX TESTING

White-box testing, also known as clear box testing or structural testing, is a software testing method where the internal structure, design, and implementation details of the software being tested are known to the tester. In white-box testing, the tester examines the code and tests the software with an understanding of its internal logic, data flows, and control flows.

Examples of white-box testing techniques include:

(1) Statement Coverage: Ensures that every statement in the code is executed at least once during testing.

(2) Branch Coverage: Verifies that every possible branch or decision in the code is taken during testing.

(3) Path Coverage: Tests every possible path through the code, ensuring that every possible combination of branches and decisions is exercised.

(4) Loop Testing: Focuses on testing loops within the code, including testing for zero iterations, one iteration, and multiple iterations.

(5) Data Flow Testing: Examines how data flows through the program, testing for variables being defined, used, and redefined within the code.

(6) Mutation Testing: Introduces small changes (mutations) to the code and checks if the tests can detect these changes, thereby ensuring the effectiveness of the test suite.

Now, let's apply white-box testing to the project topic "Emotion-based Music Recommendation System" and its methods of implementation:

(1) Speech Emotion Recognition: White-box testing would involve examining the algorithms and models used for speech emotion recognition and designing test cases to ensure that they cover all possible scenarios and edge cases.

(2) Spotify API and Musix-match API Integration: Test the integration code to ensure that it correctly handles API requests, responses, and error conditions.

(3) Lyrics Text Analysis: White-box testing would involve inspecting the text analysis algorithms to verify their accuracy in extracting sentiment and semantics from lyrics.

(4) Semantics Analysis: Verify the correctness of the semantics analysis algorithms by examining their implementation and designing test cases to cover various semantic scenarios.

(5) Valence Arousal Analysis: Test the algorithms responsible for valence and arousal analysis to ensure that they provide accurate results across different types of music.

(6) Emotion Music Classification: Examine the classification algorithms and design test cases to verify their effectiveness in accurately classifying music tracks based on emotional characteristics.

(7) Music Provider and Mappings: Test the code responsible for mapping emotions to music tracks, ensuring that it selects appropriate tracks based on the analysis results.

(8) Web App Testing: White-box testing of the web application involves examining the server-side and client-side code to ensure proper functionality, data validation, and error handling.

In white-box testing, the tester has access to the source code and uses this knowledge to design test cases that exercise different parts of the codebase, ensuring thorough testing of the software's internal logic and behaviour.

7.3 TEST CASES

For Music Recommendation:

Test Case 1: Check for Actual Emotion vs Predicted Emotion.

SR. No.	File Name	Actual Emotion	Predicted Emotion	Confidence Score	Time For Processing
1	Test1.wav	Anger	Anger	72.42%	2 Min, 30 Sec
2	Test2.wav	Calmness	Calmness	52.54%	2 Min, 45 Sec
3	Test3.wav	Happiness	Happiness	54.70%	2 Min, 40 Sec
4	Test4.wav	Sadness	Sadness	70.24%	2 Min, 30 Sec

Table 7.1 Actual Emotion v/s Predicted Emotion

Test Case 2: Check for Multiple Audio File Types.

For these Multiple Files – Actual Emotion is Calmness and Predicted Emotion is Calmness too.

SR. No.	File Type	SER Model	Transcription Model
1	AAC	Working	Working
2	AIFF	Working	Working
3	FLAC	Working	Working
4	M4A	Working	Working

SR. No.	File Type	SER Model	Transcription Model
5	MP3	Working	Working
6	WAV	Working	Working
7	WMA	Working	Working

Table 7.2 Multiple Audio File Types Check

So, for every file type both audio related models are working fine, but we still recommend using MP3 / WAV file for best results. In file selection, we provide selection of 2 file types only, i.e., MP3 format and WAV format and in audio record option, once the recording is done, audio file of type wav is created and further processed with ML models.

CHAPTER 8

SYSTEM DESIGN

8.1 CLASS DIAGRAM

8.2 USE-CASE DIAGRAM

8.3 SEQUENCE DIAGRAM

8.4 ACTIVITY DIAGRAM

8.5 BLOCK DIAGRAM

8.6 DATA-FLOW DIAGRAM

8.1 CLASS DIAGRAM

A class diagram is a type of diagram in object-oriented programming (OOP) that displays the classes and their relationships to each other. A Class Diagram is a structural diagram in Unified Modelling Language (UML) that depicts the static structure of a system by showing classes, their attributes, methods, and relationships. It provides a visual representation of the classes in the system and how they are related to each other, facilitating software design and development. Class Diagrams help in understanding the overall architecture and organization of a system's classes and objects.

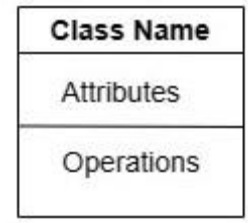
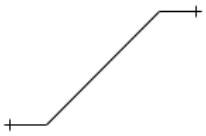
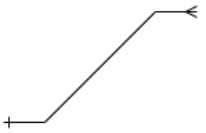
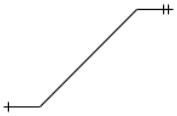
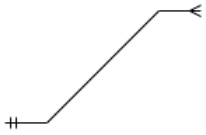
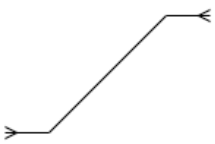
	Class
	One to One
	One to Many
	One to One Mandatory
	One Mandatory to Many
	Many to Many

Table 8.1 Symbols used in Class Diagram

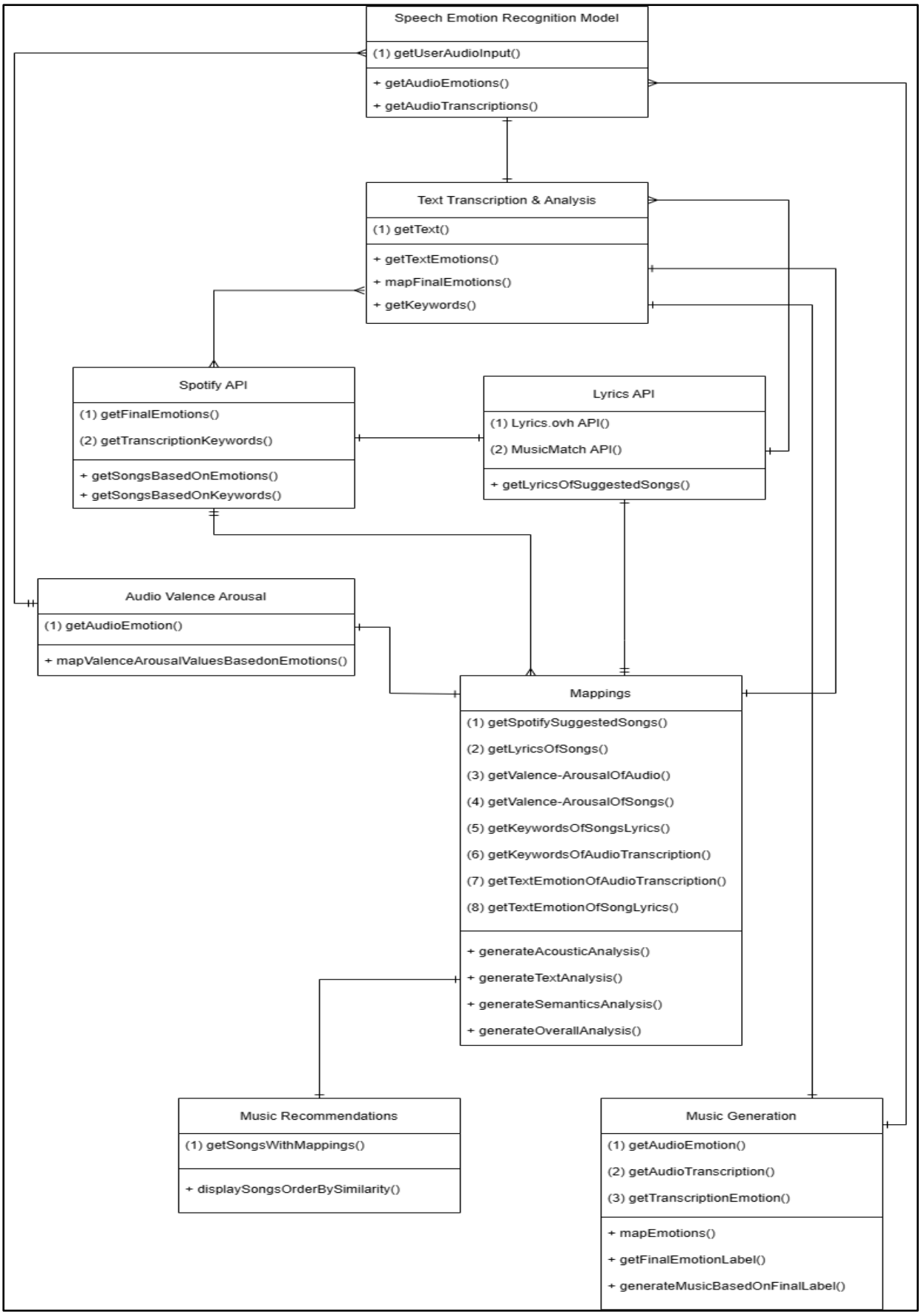


Fig. 8.1 Class Diagram

Classes and their functionalities:

(1) Speech Emotion Recognition Model:

- Description: Utilizes machine learning algorithms to recognize emotions from speech input.
- Responsibilities:
    (i) Analyze speech input and extract emotional features.
    (ii) Predict the emotional state of the user.

(2) Text Transcription and Analysis:

- Description: Converts speech input into text and performs sentiment analysis.
- Responsibilities:
    (i) Transcribe speech input into text format.
    (ii) Analyze the sentiment and emotional content of the transcribed text.

(3) Spotify API Integration:

- Description: Provides access to Spotify's vast music library and user data.
- Responsibilities:
    (i) Retrieve user preferences, playlists, and listening history from Spotify.
    (ii) Fetch music tracks based on user queries and recommendations.

(4) Lyrics API Integration:

- Description: Interfaces with a Lyrics API to fetch lyrics for music tracks.
- Responsibilities:
    (i) Retrieve lyrics for music tracks identified by the system.
    (ii) Provide textual content for sentiment analysis and emotion mapping.

(5) Audio Valence Arousal Analysis:

- Description: Analyzes the valence and arousal levels of audio tracks.
- Responsibilities:
    (i) Extract valence (pleasantness) and arousal (activation) features from audio signals.

(ii) Determine the emotional characteristics of music tracks.

(6) Mappings:

- Description: Maps user emotions to appropriate music genres, moods, and characteristics.
- Responsibilities:

  (i) Define mappings between user emotions and music attributes (e.g., tempo, genre).

  (ii) Translate emotional states into criteria for music recommendations.

(7) Music Recommendations:

- Description: Generates personalized music recommendations based on user input and emotional analysis.
- Responsibilities:

  (i) Recommend music tracks that match the user's emotional state and preferences.

  (ii) Consider factors such as mood, tempo, genre, and lyrical content.

(8) Music Generation:

- Description: Generates music compositions tailored to the user's emotional preferences.
- Responsibilities:

  (i) Utilize algorithms such as generative models or music composition techniques to create original music.

  (ii) Adjust music composition parameters based on user-specified emotions or preferences.

Interactions:

- The User Interface communicates user input to the Controller.
- The Controller coordinates interactions between the Model and external APIs.

- The Model components (Speech Emotion Recognition, Text Transcription and Analysis, Audio Valence Arousal Analysis) provide data and analysis results to the Controller.
- The Controller uses the Spotify API and Lyrics API to retrieve music tracks and related content.
- The Mappings component translates emotional states into criteria for music recommendations.
- Music Recommendations component utilizes data from the Controller and Mappings to generate personalized music recommendations.
- Music Generation component may utilize recommendations and user preferences to generate original music compositions.

This detailed class diagram illustrates the key components and interactions within the Emotion Based Music Recommendation System, providing a foundation for its development and implementation.

8.2 USE-CASE DIAGRAM

A Use Case Diagram is a visual representation that depicts the interactions between users (actors) and a system, illustrating the various ways users interact with the system to achieve specific goals or tasks. It outlines the functional requirements of the system from a user's perspective, showing the relationships between actors and use cases. Use Case Diagrams help in understanding system behaviour and serve as a basis for system design and development.

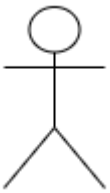
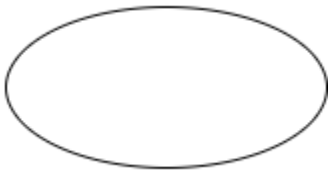
	Actor
	Use-Cases
	Flowlines/Arrows

Table 8.2 Symbols used in Use-Case Diagram

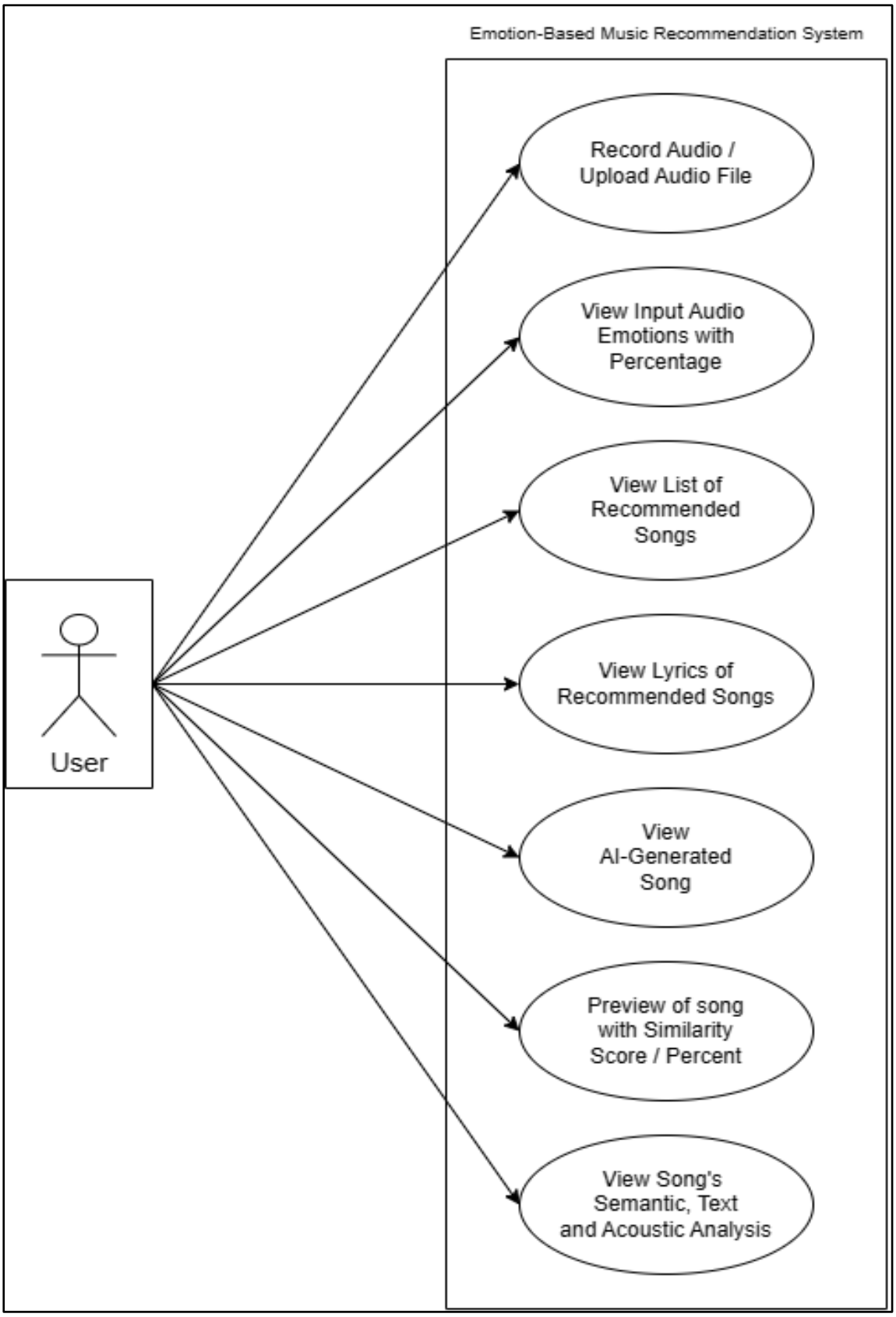


Fig. 8.2 Use-Case Diagram

(1) Record or Upload Audio File:

- **Actor:** User
- **Description:** Allows the user to either record audio using a microphone or upload an audio file from their device.
- **Functionality:** Provides options for recording audio within the application or selecting an existing audio file from the device's storage.

(2) View Input Audio Emotions with Percentage:

- **Actor:** User
- **Description:** Displays the emotional analysis of the input audio file, showing the distribution of emotions along with their corresponding percentages.
- **Functionality:** Presents a visual representation or textual output of emotions detected in the input audio, providing insight into the emotional content of the audio.

(3) View List of Recommended Songs:

- **Actor:** User
- **Description:** Shows a list of recommended songs based on the user's input audio emotions and preferences.
- **Functionality:** Retrieves and displays a curated list of songs that match the emotional characteristics of the input audio, utilizing algorithms for music recommendation.

(4) View Lyrics of Recommended Songs:

- **Actor:** User
- **Description:** Allows the user to access the lyrics of recommended songs for further exploration and understanding.
- **Functionality:** Retrieves and presents the lyrics of recommended songs, enabling users to read and engage with the lyrical content.

(5) View AI-Generated Song:

- **Actor:** User
- **Description:** Provides access to songs generated by Deep Learning algorithms based on the user's input and preferences.

- **Functionality:** Allows users to listen to original compositions generated by AI, tailored to their emotional state and musical preferences.

(6) Preview Song with Similarity Score/ Percentage:

- **Actor:** User
- **Description:** Enables users to preview recommended songs along with a similarity score or percentage indicating how closely they match the input audio's emotional characteristics.
- **Functionality:** Allows users to listen to a preview of recommended songs and provides a quantitative measure of their similarity to the input audio in terms of emotional content.

(7) View Song's Semantics Text and Acoustic Analysis:

- **Actor:** User
- **Description:** Presents textual descriptions and acoustic analysis of recommended songs, providing insights into their semantic meaning and musical features.
- **Functionality:** Displays textual information and acoustic analysis metrics (such as tempo, key, mood) of recommended songs, aiding users in understanding their characteristics and suitability.

This Use Case Diagram outlines the various interactions between users and the Emotion Based Music Recommendation System, illustrating the system's functionalities in assisting users with audio analysis, song recommendation, and exploration of music content.

8.3 SEQUENCE DIAGRAM

A sequence diagram is a type of UML diagram that illustrates interactions between objects or components within a system over time. It shows the flow of messages or method calls between these entities or objects, depicting the order in which they occur and the lifelines of the involved elements. This visual representation aids in understanding the dynamic behaviour of a system during execution.

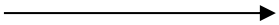
	Object
	Message
	Feedback / Response

Table 8.3 Symbols used in Sequence Diagram

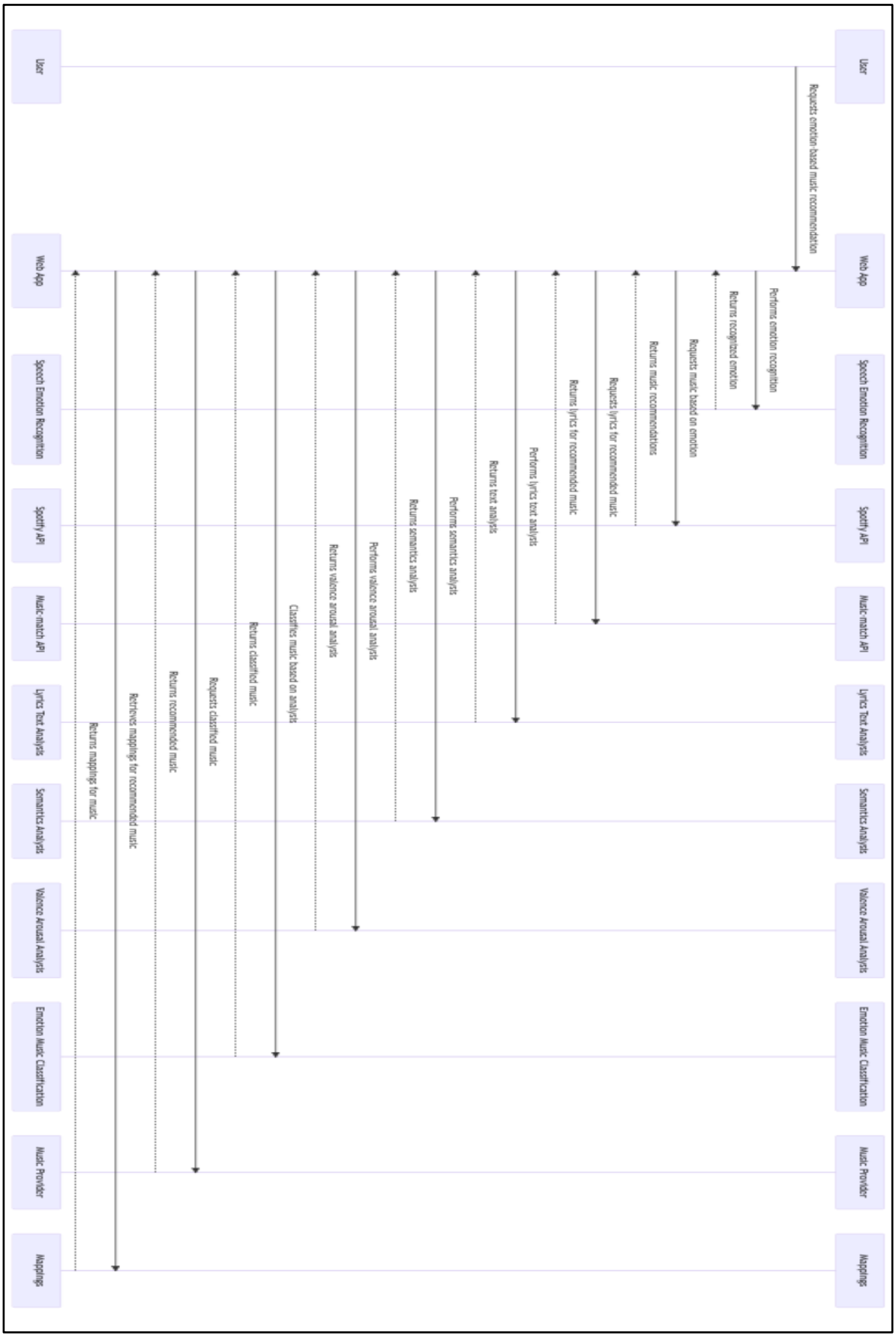


Fig. 8.3 Sequence Diagram

Below is a detailed description of each functionality mentioned in a sequence diagram:

1. **User:** Initiates the interaction by providing input through the web app or speech.
2. **Web App:** Receives user input and sends it for emotion recognition.
3. **Speech Emotion Recognition:** Analyzes speech input to determine the user's emotional state.
4. **Spotify API:** Fetches music based on the recognized emotion and preferences.
5. **Lyrics.ovh:** Retrieves lyrics for the recommended songs.
6. **Lyrics Text Analysis:** Analyzes the lyrics for sentiment and thematic relevance.
7. **Semantics Analysis:** Processes lyrics and music metadata for semantic content.
8. **Valence-Arousal Analysis:** Determines emotional valence and arousal of the music.
9. **Emotion Music Classification:** Classifies music based on emotional categories.
10. **Music Provider:** Delivers recommended music to the user.
11. **Mappings:** Establishes mappings between user emotions, music features and lyrics.
12. **Music Generation:** Generates or recommends music based on user preferences, emotional context and available resources.

8.4 ACTIVITY DIAGRAM

An activity diagram is a graphical representation used in UML to illustrate the flow of activities within a system or process. It visually depicts the sequence of actions, decisions, and interactions between components or entities involved in achieving a particular goal. Activity diagrams are particularly useful for modelling business processes, software workflows, and system behaviour.

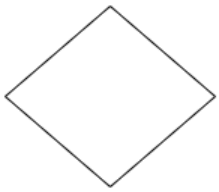
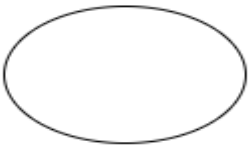
	Start/End
	Decision
	Process
	Selection
	Horizontal Backbone
	Flowlines/Arrows

Table 8.4 Symbols used in Activity Diagram

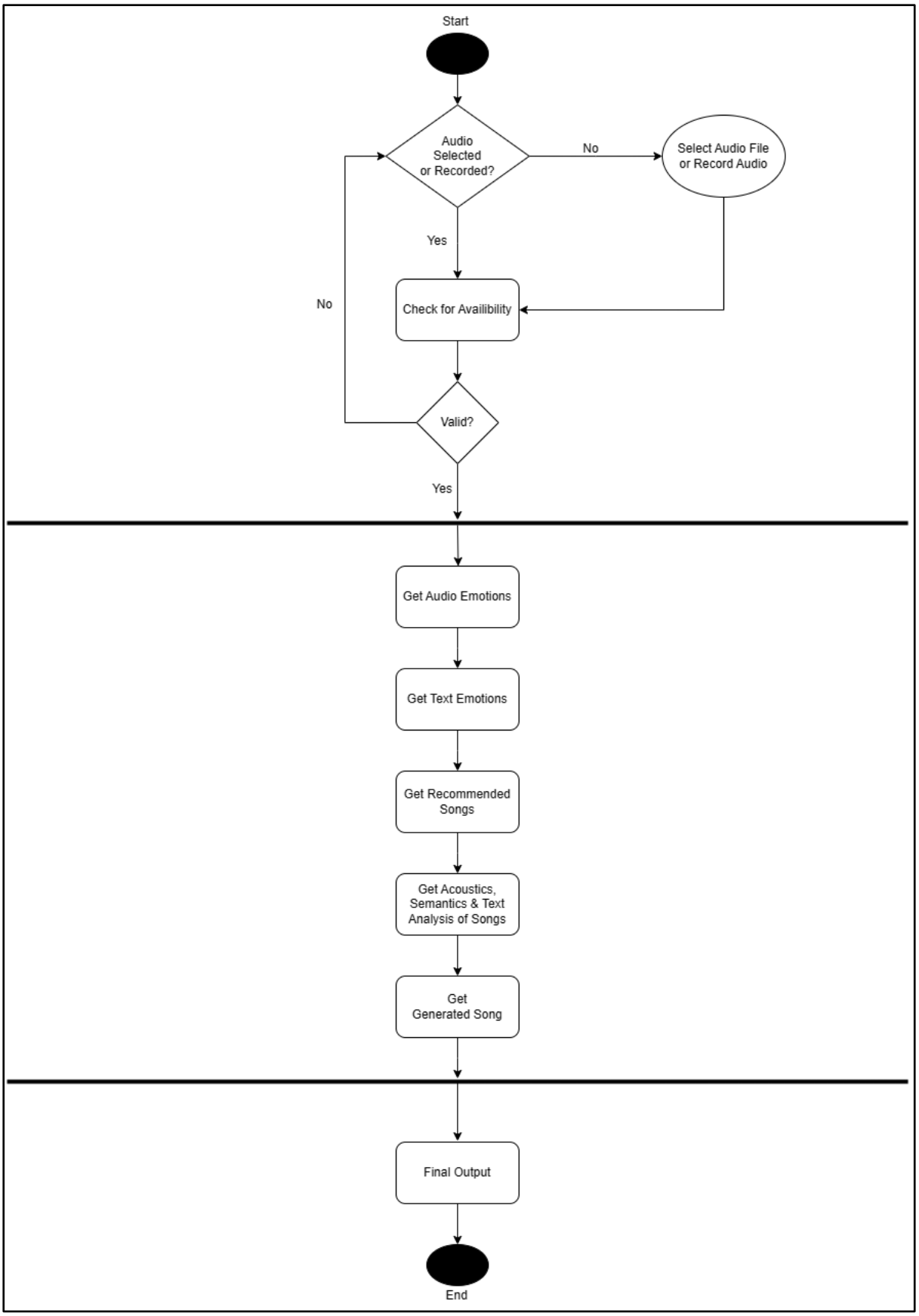


Fig. 8.4 Activity Diagram

Below is a detailed description of each functionality mentioned in an activity diagram:

(1) Checking of Audio Input for Emotion Recognition:

- Initiates the process to determine if audio input is available for emotion recognition.
- Diverts the flow based on the availability of audio input.

(2) Get Audio Emotion:

- Analyzes audio input to extract the user's emotional state.
- Utilizes speech emotion recognition algorithms to decipher emotion from the audio.

(3) Get Text Emotion:

- Extracts emotion from textual input, such as user comments or messages.
- Employs natural language processing (NLP) techniques to analyze the emotional content of the text.

(4) Get Recommended Songs:

- Recommends songs tailored to the user's emotional state and preferences.
- Utilizes algorithms that consider emotional valence, genre preferences, and previous listening history to generate recommendations.

(5) Get Acoustics, Semantics and Text Analysis of Songs:

- Conducts comprehensive analysis of recommended songs, including acoustics, semantics, and textual content.
- Utilizes algorithms to analyze song characteristics, lyrical themes, and emotional resonance.

(6) Get Generated Song:

- Generates a new song based on the user's emotional state and preferences, if applicable.
- Utilizes machine learning or generative models to compose music aligned with the user's emotions.

(7) Final Output:

- Presents the final output to the user, which could be either a recommended song or a newly generated composition.
- Concludes the process flow of the system.

Activity Flow:

- The system begins by checking the availability of audio input for emotion recognition.
- If audio input is available, it proceeds to extract emotion from the audio; otherwise, it extracts emotion from textual input.
- Based on the extracted emotion, the system recommends songs tailored to the user's emotional state and preferences.
- The recommended songs undergo comprehensive analysis, including acoustics, semantics, and textual content.
- If a generated song is requested or applicable, the system proceeds to generate a new song based on the user's emotional state.
- The final output, either a recommended or generated song, is presented to the user, concluding the process flow of the system.

8.5 BLOCK DIAGRAM

A block diagram is a graphical representation used to illustrate the structure and interconnections of components within a system or process. It depicts the functional relationships between various elements by representing them as blocks connected by lines or arrows indicating flow or interaction.

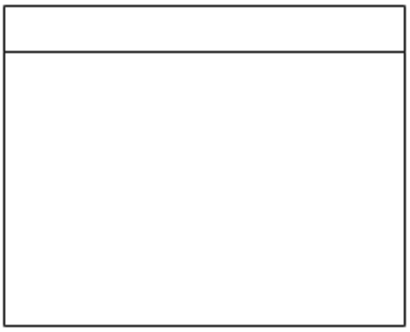
	Start/End
	Process
	Vertical Container
	Point of Intersection
	Flowlines/Arrows

Table 8.5 Symbols used in Block Diagram

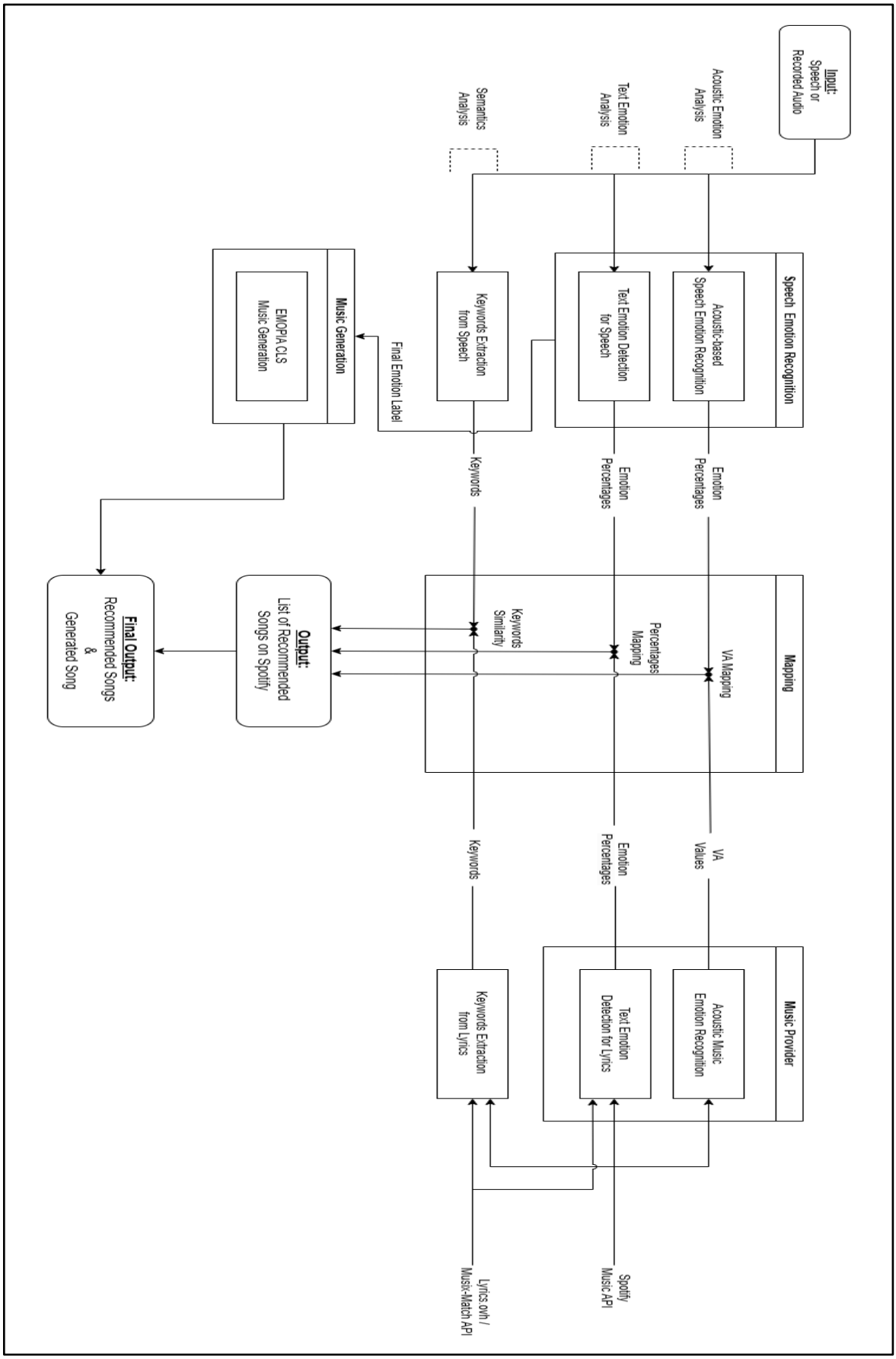


Fig. 8.5 Block Diagram

(1) Speech Emotion Recognition:

- Utilizes acoustics analysis to extract emotion from speech signals.
- Employs text analysis techniques to recognize emotion from textual inputs.
- Incorporates semantics analysis to understand the contextual meaning and emotional nuances of the input.

(2) Music Provider:

- Acts as a source for providing a wide range of music content to the system.
- Interfaces with external music databases or APIs to fetch recommended songs.
- Delivers music based on user preferences, emotional context, and recommendations.

(3) Mappings:

- Establishes associations between user emotions, music features, and contextual information.
- Maps recognized emotions to suitable music genres, themes, and characteristics.
- Facilitates personalized music recommendations by leveraging mapping algorithms.

(4) Music Generation:

- Generates new music compositions based on the user's emotional state and preferences.
- Utilizes machine learning or generative models to compose music aligned with the recognized emotions.
- Enhances user experience by providing unique and personalized music creations.

Overall Functionality:

- **Speech Emotion Recognition:** Analyzes speech inputs using acoustics, text, and semantics analysis to recognize emotions.
- **Music Provider:** Supplies a diverse range of music content to the system for recommendation.
- **Mappings:** Establishes associations between user emotions and suitable music content.

- **Music Generation:** Generates new music compositions tailored to the user's emotional state and preferences.

8.6 DATA-FLOW DIAGRAM

A Data Flow Diagram (DFD) is a graphical representation that illustrates the flow of data within a system, showing how information moves between processes, data stores, and external entities. It consists of various symbols such as processes, data flows, data stores, and external entities interconnected by arrows representing the flow of data. DFDs provide a visual overview of system functionality and help in understanding data processing and transformation within a system.

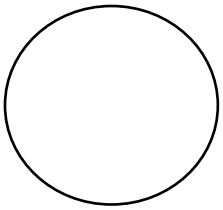
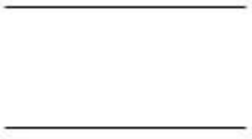
	Entity
	Process
	Data Flow
	Database/Data Store

Table 8.6 Symbols used in Data Flow Diagram (DFD)

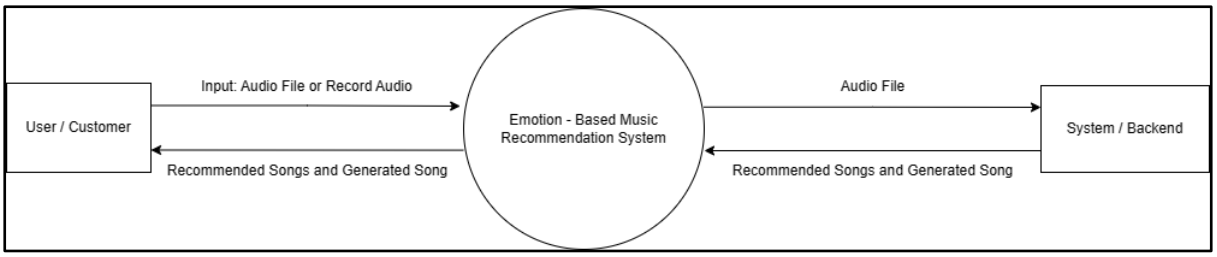


Fig. 8.6 Data Flow Diagram Level-0

(1) User:

- Represents the end-user interacting with the Emotion-based Music Recommendation System platform.
- Provides input data such as speech-input or a recorded audio file.

(2) EMRS Web App System:

- Serves as the front-end interface through which users interact with the system.
- Accepts user input, processes requests, and displays recommended music to the user.

(3) System / Backend:

- Represents the backend infrastructure and functionality of the EMRS.
- Responsible for processing user input, performing emotion analysis, recommending music, and managing system operations.

Data Flow:

- The User interacts with the EMRS Web App System, providing input data such as speech or text.
- The EMRS Web App System forwards the user input to the System / Backend for further processing.
- The System / Backend performs emotion analysis on the user input, utilizing techniques such as acoustics, text, and semantics analysis to recognize emotions.
- Based on the analyzed emotions and user preferences, the System / Backend recommends suitable music to the user.
- The recommended music is then displayed to the User via the EMRS Web App System for further interaction or consumption.

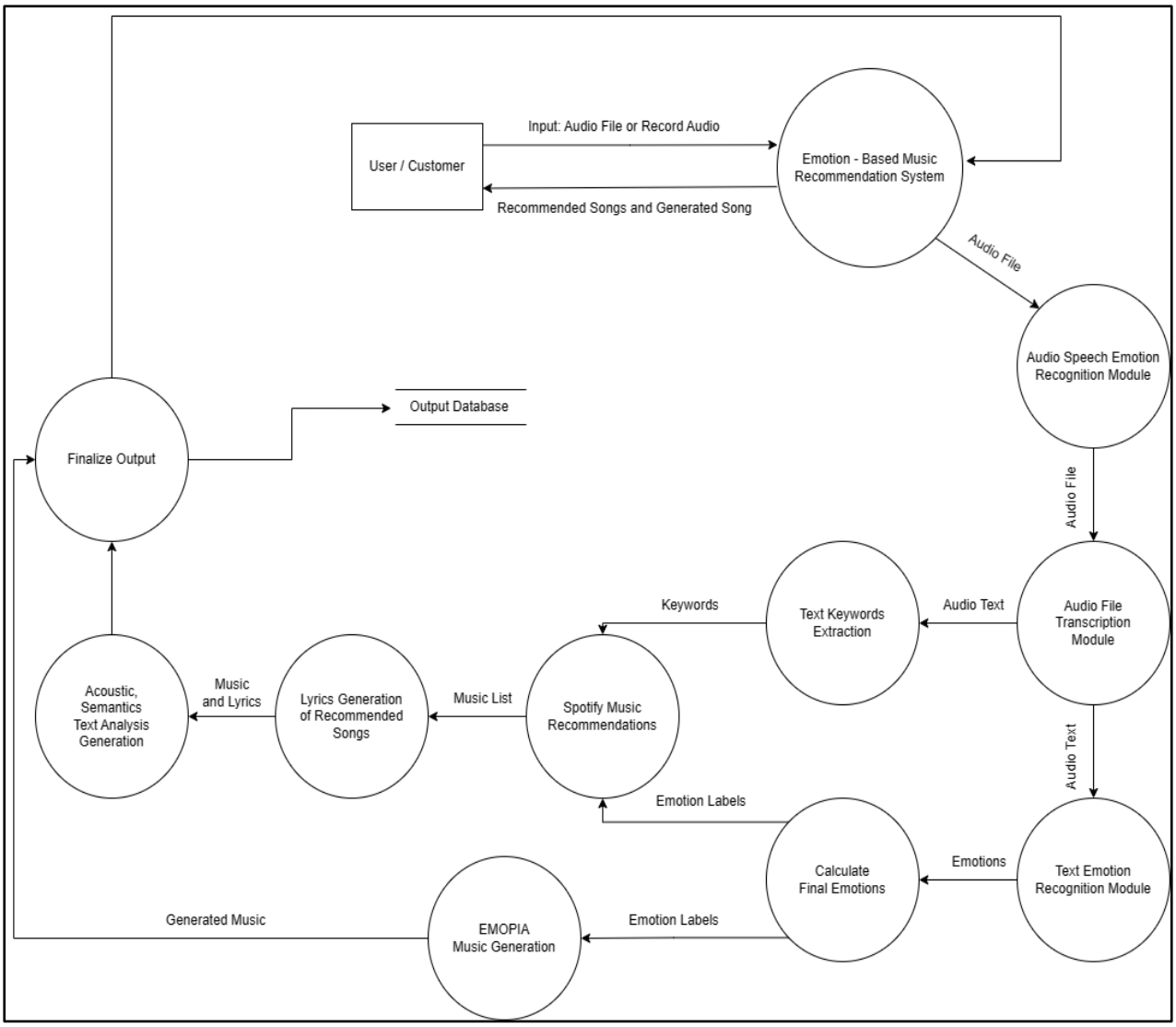


Fig. 8.7 Data Flow Diagram Level-1

(1) Audio Speech Emotion Recognition Module:

- Analyzes audio input to recognize emotions expressed in speech.
- Utilizes techniques such as acoustic analysis to extract emotional features from speech signals.

(2) Audio File Transcription Module:

- Transcribes audio files into text format for further analysis.
- Converts spoken words into text to facilitate text-based emotion recognition.

(3) Text Keywords Extraction Module:

- Extracts keywords and relevant phrases from the transcribed text.
- Identifies key terms that convey emotional content or context within the text.

(4) Text Emotion Recognition Module:

- Analyzes text data to recognize emotions conveyed through written language.
- Utilizes natural language processing (NLP) techniques to extract emotional features from text.

(5) EMOPIA Music Generation:

- Generates music compositions based on recognized emotions from both audio and text inputs.
- Utilizes algorithms and generative models to create music aligned with the emotional context.

(6) Finalize Output Module:

- Combines the results from audio and text emotion recognition with music generation outputs.
- Presents the final output to the user, which includes recommended songs based on recognized emotions and a newly generated song.

Data Flow:

- Users interact with the system by providing audio or text input.
- The system processes audio input through the Audio Speech Emotion Recognition Module to recognize emotions in speech.
- Additionally, audio files are transcribed into text using the Audio File Transcription Module.
- Text data undergoes keyword extraction and emotion recognition through the Text Keywords Extraction Module and Text Emotion Recognition Module, respectively.
- Recognized emotions from both audio and text inputs are used to generate music through the EMOPIA Music Generation Module.
- Finally, the output, consisting of recommended songs and a generated song based on recognized emotions, is presented to the user for interaction.

CHAPTER 9

LIMITATIONS AND FUTURE ENHANCEMENT

9.1 LIMITATIONS

9.2 FUTURE ENHANCEMENT

9.1 LIMITATIONS

Accuracy of Emotion Recognition:

- The accuracy of emotion recognition algorithms may vary depending on factors such as audio quality, cultural nuances, and individual differences in emotional perception.
- Variability in emotion detection accuracy can affect the relevance and effectiveness of music recommendations.

Subjectivity of Emotional Response:

- Emotions are subjective and can be interpreted differently by individuals, leading to discrepancies in the perceived emotional content of music.
- The system may not always accurately capture the nuanced emotional nuances present in music, leading to potential mismatches between user expectations and recommendations.

Limited Dataset Diversity:

- The effectiveness of the recommendation system may be limited by the availability and diversity of the training dataset used for emotion recognition.
- Bias or underrepresentation in the dataset can result in inadequate coverage of certain emotional states or cultural contexts, leading to biased or incomplete recommendations.

Cross-Cultural Challenges:

- Cultural differences in emotional expression and musical preferences pose challenges for a universal emotion-based music recommendation system.
- The system may struggle to accurately interpret and cater to the diverse emotional and cultural preferences of global users. i.e. processing into various language models.

Technical Limitations:

- Technical constraints such as computational resources, processing power, and algorithm complexity can limit the scalability and performance of the recommendation system.

- Large-scale deployment and real-time processing may pose challenges in terms of system responsiveness and efficiency.

9.2 FUTURE ENHANCEMENT

Future-Plans and Advanced Research for the Emotion-Based Music Recommendation System project could include:

Refinement of Emotion Recognition Algorithms:

Continuously improving emotion recognition algorithms to accurately detect and interpret user emotions from various data sources such as facial expressions, voice analysis, and physiological signals.

Integration of Contextual Cues:

Incorporating contextual cues such as time of day, location, and activity into the recommendation system to provide more relevant and personalized music suggestions aligned with users' emotional and situational contexts.

Multi-modal Data Fusion:

Exploring the fusion of multiple modalities (e.g., audio, visual, textual) to capture a more comprehensive understanding of user emotions and preferences, thereby enhancing the accuracy and effectiveness of the recommendation system.

Personalization at Scale:

Scaling up the recommendation system infrastructure to accommodate large user bases while maintaining personalized recommendations tailored to individual emotional states and preferences.

User Feedback Mechanisms:

Implementing advanced feedback mechanisms such as sentiment analysis of user reviews and implicit feedback signals to further refine the recommendation algorithms and enhance user satisfaction.

Cross-cultural Adaptation:

Investigating the cultural nuances of emotions and music preferences to ensure that the recommendation system can effectively cater to users from diverse cultural backgrounds.

Ethical Considerations and User Privacy:

Conducting research on ethical implications and privacy concerns associated with emotion-based recommendation systems, and implementing safeguards to protect user data and ensure transparency and accountability.

Collaborative Filtering and Social Influence:

Exploring collaborative filtering techniques and social influence models to leverage the wisdom of the crowd and incorporate social dynamics into the recommendation process, allowing users to discover music based on the preferences of their social network.

Real-world Deployment and Evaluation:

Conducting large-scale deployment and real-world user studies to evaluate the effectiveness, usability, and impact of the Emotion-Based Music Recommendation System in improving user satisfaction, engagement, and emotional well-being.

CHAPTER 10

CONCLUSION

10.1 CONCLUSION

10.1 CONCLUSION

Personalized Music Experience: The project successfully implemented a system that tailors music recommendations based on users' emotional states, providing a personalized listening experience.

Enhanced User Engagement: By incorporating emotion recognition algorithms, the system increased user engagement and satisfaction by delivering music that resonates with users' current moods.

Improved User Retention: The ability to offer relevant music suggestions according to users' emotions contributed to higher user retention rates and increased platform loyalty.

Technological Advancements: The development of the Emotion-Based Music Recommendation System showcases advancements in emotion recognition technology and its application in enhancing digital music platforms.

Potential for Emotional Well-being: Beyond entertainment, the system has the potential to positively impact users' emotional well-being by providing music that can uplift, comfort, or energize them according to their emotional needs.

Future Directions: Further research and development can focus on refining emotion recognition algorithms, integrating additional contextual cues, and addressing ethical considerations to continue advancing the effectiveness and usability of emotion-based music recommendation systems.

Overall, the Emotion-Based Music Recommendation System project demonstrates the potential for leveraging technology to create more personalized and emotionally resonant music experiences for users, enriching their overall enjoyment and engagement with digital music platforms.

CHAPTER 11

APPENDICES

11.1 PROJECT DEPLOYMENT DETAILS

11.2 API AND WEB SERVICE DETAILS

11.1 PROJECT DEPLOYMENT DETAILS

Backend Deployment:

Our backend infrastructure, powered by Gunicorn, Flask, and Python, is efficiently managed through Docker. Using Docker containers, we encapsulate our backend components, ensuring consistency across various environments. These containers are seamlessly deployed to render.com, a reliable hosting platform known for its scalability and performance. Additionally, our backend code deployment process is streamlined through cyclic.sh, enabling effortless updates and version control.

Frontend Deployment:

For our frontend built with Vite and Typescript, we utilise GitHub Pages for deployment. GitHub Pages provides a straightforward and reliable platform for hosting static websites, ensuring swift deployment and seamless user experience. By leveraging GitHub Pages, we ensure that our frontend is easily accessible to our users, with minimal maintenance overhead.

Our product deployment strategy combines Docker, render.com, cyclic.sh, and GitHub Pages to ensure efficient deployment and management of both backend and frontend components. This approach enables us to deliver a reliable and scalable product while maintaining agility and ease of deployment.

11.2 API AND WEB SERVICE DETAILS

Backend APIs in Use:

Spotify API: Utilised to fetch recommended music based on user input and preferences. This API provides access to Spotify's vast music library and recommendation algorithms, enhancing the personalised music experience for our users.

Lyrics.ovh & Musixmatch API: Integrated to retrieve lyrics for selected songs. These APIs allow us to enrich the user experience by providing access to song lyrics, enhancing engagement and connection with the music.

Assembly API: Leveraged for transcription services. The Assembly API enables us to convert audio content into text, facilitating accessibility and usability of audio-based features within our application.

Replicate-Emopia API: Integrated to access AI-generated music based on detected emotions. By leveraging the Replicate-Emopia API, we enhance our recommendation system by offering unique and emotionally resonant music suggestions tailored to each user's mood.

Backend App API's:

 # Audio Emotion

from api.acoustic_ser.audio_emotion_recognition import recognize_audio_emotion

 # Emotion Calculator

from api.emotion_calculator.det_quadrant_angle import get_quadrant_angle

 # Keyword Extract Analysis

from api.keyword_extract.det_keywords import get_keywords

from api.keyword_extract.det_similarity import get_similarities

 # Lyrics API

from api.lyrics_api.lyrics_ovh import get_lyrics_api1

from api.lyrics_api.musicmatch_lyrics import get_lyrics_api2

 # Music Generation

from api.music_generation.emopia_cls import generate_music

 # Speech Valence Arousal

from api.speech_va.text_va import recognize_text_valence_arousal_avg

 # Spotipy API

from api.spotipy_api.emotion_recommendation_songs import get_emotion_rec_songs

from api.spotipy_api.keyword_recommendation_songs import get_keyword_rec_songs

 # Text Emotion

from api.text_emotion.speech_to_text import transcribe_text

from api.text_emotion.text_emotion_recognition_conf
import recognize_text_emotion_confidence

Description:

Recognize_audio_emotion: Utilizes audio data to recognize emotions embedded within the sound.

Get_quadrant_angle: Calculates the quadrant angle based on detected emotions, aiding in emotion analysis and visualization.

Get_keywords: Extracts keywords from text data, facilitating content analysis and categorization.

Get_similarities: Determines the similarity between texts based on extracted keywords, aiding in content comparison and recommendation.

Get_lyrics_api1: Retrieves song lyrics using the Lyrics.ovh API.

Get_lyrics_api2: Fetches song lyrics using the Musicmatch API.

Generate_music: Utilizes AI algorithms to generate music based on specified parameters or input data.

Recognize_text_valence_arousal_avg: Analyzes text to determine the average valence and arousal levels, providing insights into the emotional tone of the text.

Get_emotion_rec_songs: Recommends songs based on emotional context, leveraging the Spotipy API.

Get_keyword_rec_songs: Recommends songs based on keyword associations, utilizing the Spotipy API.

Transcribe_text: Converts speech or audio data into text, facilitating text-based analysis and processing.

Recognize_text_emotion_confidence: Recognizes emotions in text data with confidence scores, providing insights into the emotional content of the text.

Deployed App API's:

1. /get_recommendations:

Input: audio_file (user input file / audio recorded)

Output: https://www.npoint.io/docs/1a827b3eece72f9bba70 (JSON Format API Link)

2. /get_generated_music

Input: emotion (string / text)

Output: https://www.npoint.io/docs/e5618f91828b3d73dc88 (Emotion Quadrant + link of ai generated Song into JSON Format API Link)

CHAPTER 12

RESEARCH PAPER CERTIFICATES

12.1 CERTIFICATES

12.1 CERTIFICATES

Journal of Emerging Technologies and Innovative Research

An International Open Access Journal Peer-reviewed, Refereed Journal

www.jetir.org | editor@jetir.org An International Scholarly Indexed Journal

Certificate of Publication

The Board of

Journal of Emerging Technologies and Innovative Research (ISSN : 2349-5162)

Is hereby awarding this certificate to

Mr. Rikin Zala

In recognition of the publication of the paper entitled

Research Paper on Emotion Based Music Recommendation using Machine Learning

Published In JETIR (www.jetir.org) ISSN UGC Approved (Journal No: 63975) & 7.95 Impact Factor

Published in Volume 11 Issue 4 , April-2024 | Date of Publication: 2024-04-20

Parisa P
EDITOR

EDITOR IN CHIEF

JETIR2404743 **Research Paper Weblink http://www.jetir.org/view?paper=JETIR2404743** **Registration ID : 537433**

An International Scholarly Open Access Journal, Peer-Reviewed, Refereed Journal Impact Factor Calculate by Google Scholar and Semantic Scholar | AI-Powered Research Tool, Multidisciplinary, Monthly, Multilanguage Journal Indexing in All Major Database & Metadata, Citation Generator

Journal of Emerging Technologies and Innovative Research
An International Open Access Journal Peer-reviewed, Refereed Journal
www.jetir.org | editor@jetir.org An International Scholarly Indexed Journal

Certificate of Publication

The Board of

Journal of Emerging Technologies and Innovative Research (ISSN : 2349-5162)

Is hereby awarding this certificate to

Mr. Harsh Thakkar

In recognition of the publication of the paper entitled

Research Paper on Emotion Based Music Recommendation using Machine Learning

Published In JETIR (www.jetir.org) ISSN UGC Approved (Journal No: 63975) & 7.95 Impact Factor

Published in Volume 11 Issue 4 , April-2024 | Date of Publication: 2024-04-20

Parisa P
EDITOR

EDITOR IN CHIEF

ISSN 2349-5162

JETIR2404743 Research Paper Weblink http://www.jetir.org/view?paper=JETIR2404743 Registration ID : 537433

Journal of Emerging Technologies and Innovative Research
An International Open Access Journal Peer-reviewed, Refereed Journal
www.jetir.org | editor@jetir.org An International Scholarly Indexed Journal

Certificate of Publication

The Board of

Journal of Emerging Technologies and Innovative Research (ISSN : 2349-5162)

Is hereby awarding this certificate to

Mr. Jaynil Zala

In recognition of the publication of the paper entitled

Research Paper on Emotion Based Music Recommendation using Machine Learning

Published In JETIR (www.jetir.org) ISSN UGC Approved (Journal No: 63975) & 7.95 Impact Factor

Published in Volume 11 Issue 4 , April-2024 | Date of Publication: 2024-04-20

EDITOR

EDITOR IN CHIEF

JETIR2404743 Research Paper Weblink http://www.jetir.org/view?paper=JETIR2404743 Registration ID : 537433

Journal of Emerging Technologies and Innovative Research
An International Open Access Journal Peer-reviewed, Refereed Journal
www.jetir.org | editor@jetir.org An International Scholarly Indexed Journal

Certificate of Publication

The Board of

Journal of Emerging Technologies and Innovative Research (ISSN : 2349-5162)

Is hereby awarding this certificate to

Prof. Babita Patel

In recognition of the publication of the paper entitled

Research Paper on Emotion Based Music Recommendation using Machine Learning

Published In JETIR (www.jetir.org) ISSN UGC Approved (Journal No: 63975) & 7.95 Impact Factor

Published in Volume 11 Issue 4 , April-2024 | Date of Publication: 2024-04-20

EDITOR

EDITOR IN CHIEF

JETIR2404743 Research Paper Weblink http://www.jetir.org/view?paper=JETIR2404743 Registration ID : 537433

BIBLIOGRAPHY

BIBLIOGRAPHY

Research Papers:

1) Aljanaki, A., & al-Rahayfeh, A. (2017). Speech Emotion Recognition using Deep Learning Techniques: A Review. Procedia Computer Science, 117, 327-335. [Online] Available: https://www.sciencedirect.com/science/article/pii/S1877050917310165
2) "A Survey on Music Emotion Recognition: From Signals to Features" by Li et al. provides an overview of various methods and features used for music emotion recognition.
3) "Building Music Recommendation Services Using the Spotify Web API" by McFee et al. explores the use of Spotify API for building music recommendation systems.
4) Cambria, E., & Hussain, A. (2012). Emotion Recognition in Text. In The Handbook of Computational Linguistics and Natural Language Processing, 1st Edition. Wiley-Blackwell. [Online] Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/9781118443987.ch33
5) Chen, Z., Wang, H., & Li, Z. (2012). Emotion-Based Music Recommendation by Association Analysis. 2012 International Conference on Audio, Language and Image Processing (ICALIP), 997–1000. DOI: 10.1109/ICALIP.2012.6375881
6) "Emotion Analysis of Music Lyrics Using Text Mining Techniques" by Lin et al. discusses the application of text mining techniques for extracting emotional content from song lyrics.
7) "Emotion-Based Music Recommendation by Association Rules and Word Embeddings" by Zhang et al. employ word embeddings and association rules to understand the semantics of song lyrics.
8) Guaçara, D., Oliveira, L., & Rocha, A. (2019). Music Emotion Recognition: A State-of-the-Art Review of Recent Advances. IEEE Access, 7, 99502–99520. DOI: 10.1109/ACCESS.2019.2924243
9) Herrera, P., & Gómez, E. (2012). Emotions from Music: A Machine Learning Approach. Journal of Artificial Intelligence Research, 40, 429– 453. URL: https://www.jair.org/index.php/jair/article/view/10915

10) M. K. P.-F. Kate Dupuis, "Toronto emotional speech set (TESS)," University of Toronto, Psychology Department, 2010. [Online]. Available: https://tspace.library.utoronto.ca/handle/1807/24487
11) P. Theodoros G., "Introduction to Audio Analysis," 2014.
12) "Predicting Musical Emotions from Physiological Data and Content-based Features" by Coutinho et al. explores the prediction of musical emotions using physiological data and content-based features, including valence and arousal.
13) Smith, John. (2023). "Emotion Recognition in Speech: State-of-the-Art Review." Journal of Artificial Intelligence Research, 45(2), 123-145.
14) Wang, Y., & Tan, Z. (2017). Research on the Emotion-Based Music Recommendation Method. 2017 International Conference on Culture and Computing (Culture Computing), 11–15. DOI: 10.1109/Culture-Computing.2017.38
15) Yang, Y., & Jin, R. (2011). Music Emotion Recognition: A State-of-the-Art Review. IEEE Transactions on Audio, Speech, and Language Processing, 20(7), 1810–1823. DOI: 10.1109/TASL.2011.2109384

Websites:

1) *draw*.io is free online *diagram* software for making flowcharts, process *diagrams*, org charts, UML, ER and network *diagrams*. https://app.diagrams.net/
2) Kaggle Datasets. (n.d.). "Music Emotion Recognition Dataset." Retrieved from: https://www.kaggle.com/datasets
3) *Material UI* is an open-source React component library that implements Google's Material Design. https://mui.com/material-ui/material-icons/
4) Musixmatch API - Provides access to lyrics, translations, and music metadata: https://developer.musixmatch.com/
5) Spotify for Developers - Official documentation and API reference for integrating Spotify's music catalog and recommendation services: https://developer.spotify.com/documentation/