Software Requirements Specification (SRS)
Group 7

MuseAI

Lyrics and Chord Generation System

Software Engineering Project

METCS 673

BOSTON UNIVERSITY

Tridev Rapeti
Shreni Singh
Lingjie Yuan
Aayush Raghuvanshi

September 2024

1 Introduction

1.1 Purpose

The primary purpose of the Software Requirements Specification (SRS) document is to provide a clear description of the requirements of MuseAI. It serves as a formal consensus between stakeholders (such as clients, developers, and project managers) on what MuseAI is expected to do and ensures the integrity of the product.

1.2 Scope

The scope of this document is:

- **High-level Requirements**
  - **Functional Requirements**: include all services and observable behaviors that the MuseAI need to provide.
  - **Non-Functional Requirements**: include characteristics of services such as quality, performance, constraints, security, and reliability.
- **High-level Interface(GUI or API) Designs**
  - Refer to the general overview and layout of user interface (GUI) and user experience design.

2 High-level Requirements

2.1 Functional Requirements

2.1.1 Mood Analysis

- **Description**: In the realm of music appreciation, it's hard, especially for amateurs to accurately capture the emotions of songs, especially when dealing with complex feelings. MuseAI will develop the function that enable people to catch intended emotions more effectively and deepen people's understanding of music.
- **Implementation**
  - **Input**: The text of the song's lyrics, either uploaded as a text file or entered directly in a input box. And an audio recording of the song in commonly used formats, which is optional for users.
  - **Methods**: For lyrics, tokenize and clean the lyrics to remove punctuation and irrelevant characters firstly. And use BERT to analyze word embedding to extract emotional cues from the text. For audio, normalize the audio files for consistent analysis at first, and employ Librosa to extract temp, pitch and intensity to quality emotions.
  - **Output**: A list of emotions detected in the input contents, such as happiness, sadness, anger, nostalgia, etc. And numeric scores or percentages for each identified emotion.

2.1.2 Lyrics Generation

- **Description**: Many music creators struggle with turning abstract ideas or themes into concrete lyrics, especially when they have specific concepts in mind but lack the inspiration or time to craft lyrics that fit. MuseAI can help creators quickly translate their ideas into meaningful lyrics without sacrificing artistic quality.
- **Implementation**
  - **Input**: A list of phases inputted or selected by the user via the input window.
  - **Methods**: Pretrain the Generative Transformer model with musiXmatch dataset to ensure creative and diverse lyric-formatted output. And fine-tune the input layer of this model to adapt the inputted phases.
  - **Output**: A set of lyrics that incorporate the provided keywords and formatted in a common song structure that is easy to use in music composition.

2.1.3 Chord Generation

- **Description**: Due to the limitation of existing chords, musicians are always agonizing over the choice of chords. Chords selection must avoid repetitions, but should conform to the genre of the song. MuseAI can break this dilemma by recommending chords according to the classification model.
- **Implementation**
  - **Input**: A list of words, phrases, or full lyrics provided by the user that convey the theme of the song.
  - **Methods**: Use a chord progression model(not decided yet) trained on musiXmatch datasets that map different emotional tones, themes, and genres to appropriate chord progressions. For example, a sad mood might map to minor key progressions, while a happy tone might map to major chords. And fine-tune the input layer to fit inputs.
  - **Output**: Chords aligns with the provided keywords and represented in standard musical notation.

2.1.4 File Upload and Download

- **Description**: To ensure ease of use and simplicity in MuseAI, the product should have the features for uploading both text and audio data, as well as exporting text and audio segments. People will be able to upload their works and get the feedback, all while enjoying a user-friendly interface, without any professional knowledge.
- **Implementation**
  - **Upload**: Design an input space to type in any thought, and set a button beside it to drag files into this program.
  - **Download**: Copy the results from the web page directly, or press the download button to get a file formatted result.

2.1.5 Customization Option

- **Description**: MuseAI encourages users to personalize and modify the generated outputs according to their preferences and creative vision. By adjusting the lyrics or chord progressions to better suit their style, users can take full control of the artistic process, leading to a more fulfilling and customized experience.
- **Implementation**: Create a user-friendly text or chord real-time editor where users can directly modify generated lyrics or chords with HTML5.

2.2 Non-Functional Requirements

2.2.1 Performance

- **Description**: MuseAI should provide fast response times for real-time editing, data processing, and generation of lyrics or chords. The system should handle multiple users simultaneously without noticeable delays.
- **Implementation**
  - Use caching for frequently used data (e.g., chord progression patterns, genre-specific rules).
  - Optimize server performance with load balancing and asynchronous processing to handle multiple requests.
  - Minimize latency by utilizing WebSockets or Socket.IO for real-time updates.
  - Pretrain models for lyrics generating and music classification.

2.2.2 Reliability

- **Description**: The system should be available with minimal downtime and be able to recover quickly from any failures or errors.
- **Implementation**
  - Use database replication to ensure data is consistently backed up and available in case of primary database failure.

2.2.3 Usability

- **Description**: The system should be intuitive, easy to navigate, and accessible to all users, regardless of their technical expertise. The UI should be simple to use, with clear instructions for each feature.
- **Implementation**
  - Conduct advanced user testing to ensure the interface is intuitive and easy to navigate.
  - Implement inline editing, and real-time feedback to make interactions smoother.
  - Provide tool tips and help instruction for users who need guidance.

2.2.4 Scalability

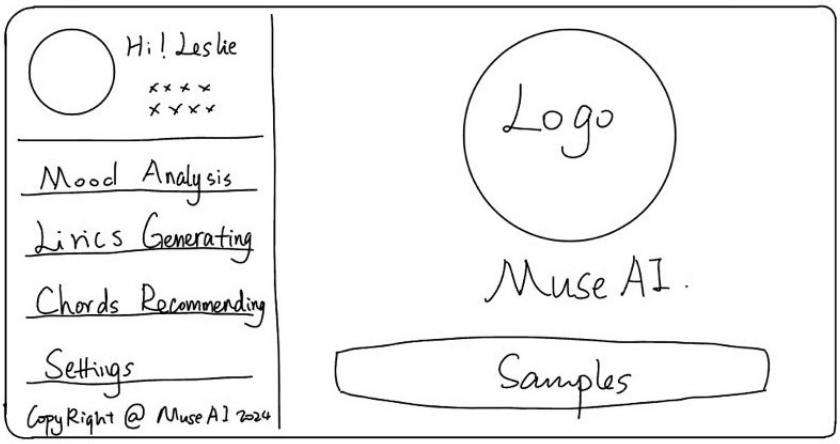
- **Description**: The system should be able to scale to accommodate increasing numbers of users or larger datasets without degrading performance.
- **Implementation**
  - Implement cloud-based solutions, i.e. AWS, for flexible resource allocation.
  - Use containerization, i.e. Docker, to ensure smooth deployment and scaling of services.

2.2.5 Security

- **Description**: The system must ensure the privacy and security of user data, preventing unauthorized access, data leaks, or breaches.
- **Implementation**
  - Grant only the minimum necessary permissions to users, applications, and services to limit unauthorized access.
  - Implement MFA for database administrators to strengthen access security and ensure only authorized personnel can access the database.
  - Schedule regular automatic backups to prevent data loss.

3 High-level Interface(GUI or API) Designs

3.1 Main Interface



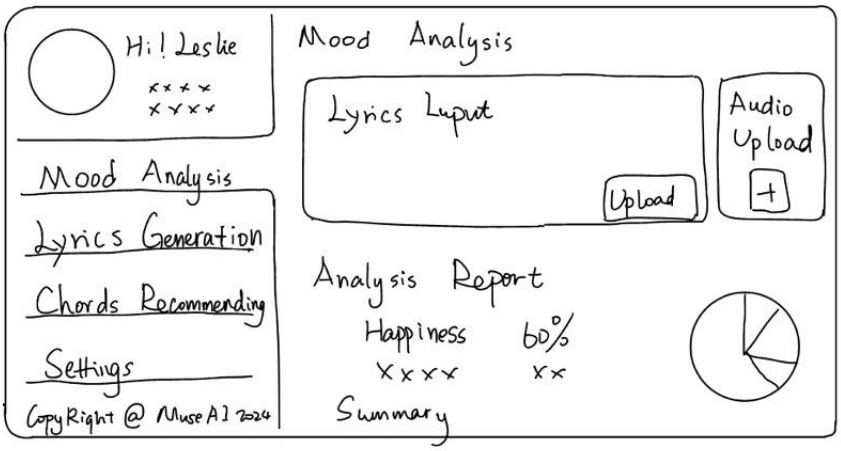
- **User Profile**
  - Show the user profile, including name, portrait, and additional details.
- **Menu Bar**
  - Display the main functions of MuseAI, and press each entry to access the different interfaces.

- Copyright at the bottom of the menu bar.

- **Logo (Main Section)**
  - Include the logo and name of MuseAI, which should be in the middle of interface.

- **Samples and Instruction**
  - Show some samples and instructions of MuseAI.

3.2 Mood Analysis Interface

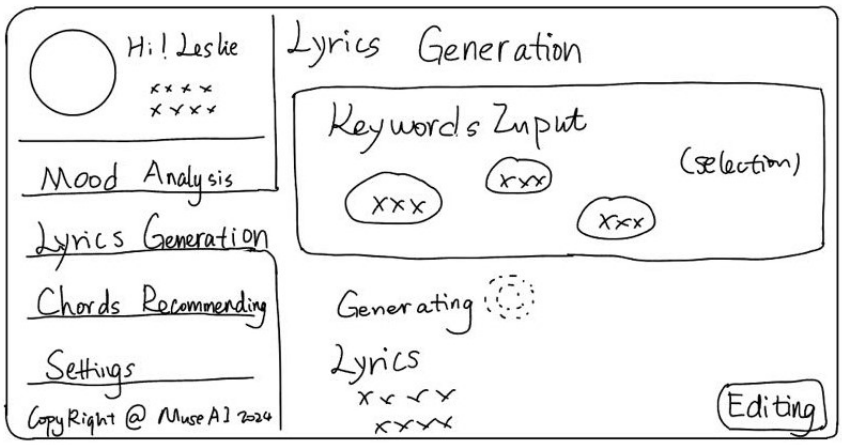


- **User Profile and Menu Bar**
  - Same as the one in the main interface.
  - Highlight the "Mood Analysis" item, and enter the "Mood Analysis" page.

- **Upload Window**
  - Include the text upload part and audio upload part.
  - An input window to type in the lyrics, or use the button to upload a lyrics file.
  - An "add" button to upload an audio file.

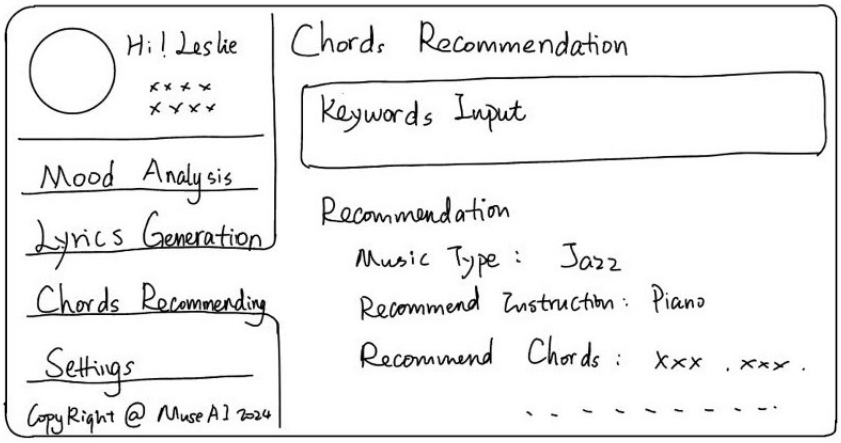
- **Analysis Report**
  - Show the analysis of the music, including percentage of each emotion, diagrams, and summary.

3.3 Lyrics Generation Interface



- **User Profile and Menu Bar**
  - Same as the one in the main interface.
  - Highlight the "Lyric Generation" item, and enter the "Lyric Generation" page.
- **Keywords Window**
  - An input window to type in the keywords of the lyrics users want to generate.
  - Some recommended words in the window for user to select.
- **Generating Loading Signal**
  - Appear when the lyrics is generating.
- **Generated Lyrics**
  - Display the generated lyrics, and press the "Editing" button to edit the lyrics online.

3.4 Chord Recommendation Interface

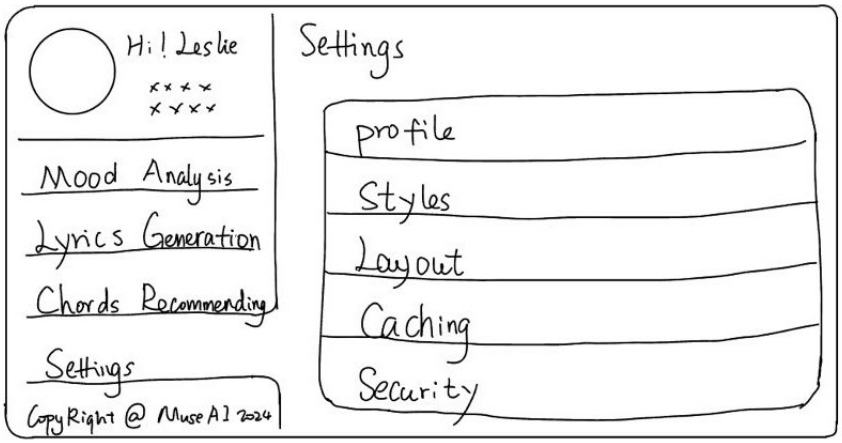


- **User Profile and Menu Bar**
  - Same as the one in the main interface.

- Highlight the "Chord Recommendation" item, and enter the "Chord Recommendation" page.

- **Keywords Window**
  - An input window to type in the keywords or lyrics.
- **Recommendation Report**
  - Show the chords recommendation.

3.5 Settings Interface



- **User Profile and Menu Bar**
  - Same as the one in the main interface.
  - Highlight the "Settings" item, and enter the "Settings" page.
- **Setting Entries**
  - Show different setting that user can set, including profile, styles,layout, caching management, and security management.