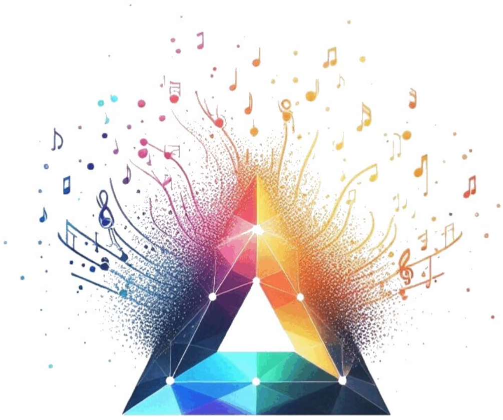
Software Project Management Plan (SPMP)
Group 7



MuseAI

Lyrics and Chord Generation System

Software Engineering Project

METCS 673

BOSTON
UNIVERSITY

Tridev Rapeti
Shreni Singh
Lingjie Yuan
Aayush Raghuvanshi

September 2024

1 Purpose

The purpose of the Software Project Management Plan (SPMP) for an AI-based software project that generates lyrics based on mood or emotion-driven chords or music is to provide a comprehensive framework for managing and executing the project. This plan ensures that all aspects of the project are carefully considered and planned for, facilitating a clear understanding of the project's objectives, scope, constraints, and deliverables.

2 Scope

The scope of this project could be understood from the following key elements:

2.1 Objective

Develop an AI-based software, MuseAI, capable of generating mood or emotion based chord progression and then lyrics that align with the mood or emotion of the generated musical chords or compositions.

2.2 Functional Requirements

- **Mood Analysis:** Implement algorithms to analyze probable musical chords suitable for a given mood or emotion.
- **Chord Generation:** Develop AI models to generate chord progression that match the identified mood or emotion.
- **Lyrics Generation:** Develop AI models to generate lyrics that match the identified mood or emotion and the music.
- **User Interface:** Create an intuitive user interface allowing users to select moods, receive generated lyrics, and .
- **Customization Options:** Provide options for users to customize the generated lyrics, such as adjusting themes, complexity, and language.

2.3 Non-Functional Requirements

- **Performance:** Ensure the software can process inputs and generate lyrics in a timely manner.
- **Scalability:** Design the system to handle multiple users and large datasets efficiently.
- **Usability:** Focus on creating a user-friendly interface that is accessible to users with varying levels of technical expertise.
- **Security:** Implement measures to protect user data and ensure the privacy of generated content.

2.4 Delivarables

- **Pseudo Prototype:** Develop a working pseudo prototype demonstrating update till the mid term.
- **Final Prototype:** Deliver the fully functional MuseAI software prototype with all specified features.
- **Documentation:** Provide comprehensive documentation, including Proposal document, SCMP, SPMP, and dependencies setup guidelines.
- **Testing Reports:** Conduct thorough testing and provide reports on software performance, usability, and security.

2.5 Exclusions

- **Live Performance Integration:** The project will not include features for live performance or real-time lyric generation during live music sessions.
- **Advanced Music Composition:** The project will focus on lyric generation and chord progression and will not include advanced music composition or arrangement features.

2.6 Constraints

- **Budget:** As this is a class project, budget should be zero preferably by using all the free resources available online for students.
- **Timeline:** Complete the project within the specified timeframe, ensuring all milestones are met and deliver a psuedo prototype for mid term presentation.
- **Technology:** Utilize existing simple web backend frameworks and tools to develop the software, avoiding complex frameworks or technologies as the project is based on AI and should focus more on the AI software with uncompromising UI/UX.

2.7 Assumptions

- **Data Availability:** Assume access to a sufficient dataset of music and lyrics for training the AI models.
- **User Base:** Assume a diverse user base with varying levels of musical expertise and preferences.
- **Technology Stack:** Assume the availability of necessary hardware and software resources for development and deployment.

3 Responsibilities and Scheduling

3.1 SCRUM TImeline

We would like to adopt Agile Software Development Technique in order to stay updated and plan for project status reports in between which are to be submitted. Our Scrum time table, which corresponds with the course's project status review schedule, highlights important benchmarks and review sessions. The project is developed between September 16 and December 6, with frequent Scrum meetings and task reviews to ensure efficient project workflow and on-time project delivery.

- **Sprint 1 (09/16 - 09/24)**:
  - **Kickoff Scrum Meeting (09/17)**: Initial sprint planning. Discuss task delegation for the project proposal submission due this week.
  - Tasks: Group project proposal (final) and initial setup for the project.
- **Sprint 2 (09/25 - 10/01)**:
  - **Scrum Meeting (09/24)**: Review of progress, discuss feedback from the proposal, and plan tasks for Project Status Report 1.
  - Tasks: Complete Project Status Report 1.
- **Sprint 3 (10/02 - 10/08)**:
  - **Scrum Meeting (10/01)**: Plan for Project Status Report 2, finalize presentation slides for mid-semester project presentations.
  - Tasks: Complete Project Status Report 2 and finalize presentation slides.
- **Sprint 4 (10/09 - 10/22)**:
  - **Mid-Sprint Scrum Meeting (10/08)**: Mid-semester presentations review, adjustments based on presentation feedback.
  - **Scrum Meeting (10/15)**: Prepare for Project Status Report 3.
  - Tasks: Complete Project Status Report 3 and refine the project based on feedback.
- **Sprint 5 (10/23 - 10/29)**:
  - **Scrum Meeting (10/22)**: Sprint planning for Project Status Report 4 and assign new development tasks.
  - Tasks: Complete Project Status Report 4.
- **Sprint 6 (10/30 - 11/12)**:
  - **Scrum Meeting (10/29)**: Planning for Project Status Report 5 and testing of key features.
  - **Scrum Review (11/05)**: Evaluate testing progress and prepare final improvements.
  - Tasks: Complete Project Status Report 5.

- **Sprint 7 (11/13 - 11/26)**:
  - **Scrum Meeting (11/12)**: Final sprint planning for Project Status Report 6. Address pending issues.
  - Tasks: Complete Project Status Report 6 and final improvements.
- **Sprint 8 (11/27 - 12/03)**:
  - **Scrum Meeting (11/26)**: Sprint planning for final project presentations and report submission.
  - **Final Scrum Review (12/03)**: Review all project deliverables, finalize presentation slides.
  - Tasks: Final project presentations and report submission.

3.2 Project Responsibilities

We have a small but dynamic software team who are on-point in delivering efficient work performance. The team, roles and their responsibilities are listed below:

- **Project & Team Managers and Scrum Masters**: Tridev Rapeti, Shreni Singh
- **Technical writers**: Aayush Raghuvanshi, Shreni Singh, Lingjie Yuan, Tridev Rapeti
- **Software Developers**: Lingjie Yuan, Shreni Singh, Tridev Rapeti, Aayush Raghuvanshi
- **Design and UI/UX team**: Tridev Rapeti, Aayush Raghuvanshi, Shreni Singh
- **Qualitative Engineers**: Lingjie Yuan, Aayush Raghuvanshi

4 Project Management Tools

- Git as file manager, collaborative space and file operations such as sharing, editing and merging changes.
- AWS JetBeans for free Cloud storage based on student discount
- Hosting the software locally to support it without additional budget

5 Probable problems and precautions

- **Algorithm Feasibility Issue:** The AI model might produce irrelevant lyrics and chords or the system might experience delays in generating lyrics and chords, leading to poor user experience.
  - Train different models with diverse datasets to improve accuracy and efficiency and incorporate user feedback for fine-tuning.

  - Consider using dimension reduction or approximate methods to manage computational complexity.

- **Effectiveness Issue:** There may be a risk of not meeting goals within the set timeframe due to scope expansion, unforeseen challenges, or force majeure.
  - Focus on delivering a minimum viable product first, ensuring core functionality is completed on time, and then gradually adding additional features.
  - Monitor project progress regularly to identify potential delays and take corrective actions.
  - Adjust the project plan and resource allocation based on actual progress to ensure timely completion.

- **Data Security Issue:** Data may be tampered with or corrupted, affecting the reliability of generated content.
  - Schedule regular backups of all critical data to ensure that up-to-date copies are available in case of data loss or corruption.
  - Implement robust key management practices to protect encryption keys and ensure they are securely stored and used.

- **Copyright issues:** The music generated, the code used, or the media used could be challenged for a copyright claim
  - Try to avoid using copyrighted content in the first place and if wanted to use it, give the author or the owner their credits.
  - AI-generated music can be protected from copyright laws by limiting human intervention to a limit as AI-generated content with no sufficient human intervention doesn't come under the Copyright Act.