Problem Statement and Goals
Software Engineering

Team 8 – Rhythm Rangers

Ansel Chen
Muhammad Jawad
Mohamad-Hassan Bahsoun
Matthew Baleanu
Ahmed Al-Hayali

September 24, 2024

Table 1: Revision History

Date	Developer(s)	Change
2024-09-24	All members	Complete Revision 0

1 Problem Statement

1.1 Problem

1.1.1 Problem Definition

Music is an effortless art to consume, but with a laborious and inaccessible creation process. It demands extensive proficiency in playing instruments, the use of complex music production software, e.g., FL Studio or Cakewalk, the use of mixing tools and techniques, e.g., level faders and equalization, among other barriers to entry. GenreGuru strives to democratize experimentation in music production, making it accessible to beginners and more streamlined for professionals by automatically generating songs or snippets inspired by user-inputted songs or snippets. In support of the generative system, a music analysis tool can be used to summarize songs using tabular features, akin to Spotify's "danceability", for example. Finally, a catalogue of songs and their corresponding features can be accessed to recommend users songs that share similar features to ones they input.

1.1.2 Problem Importance

This project holds significant value for a wide range of users. Music recommendation systems have millions of daily users, and improving the quality of recommendations can greatly enhance user satisfaction. Additionally, by providing tools that make music creation more accessible, the project opens creative opportunities for individuals who might otherwise be excluded due to the technical barriers involved in music production. The platform is poised to impact both the music industry and hobbyists, facilitating innovation and creativity in music generation and exploration.

1.2 Inputs and Outputs

The project can be characterized by three systems: *music generation*, *music analysis*, and *music recommendation*. For completeness, a song is a full track, i.e., a completed audio file published by a musician, and a snippet is a fragment of a track, i.e., an incomplete audio file. Such audio files can be in a *lossy* compressed format, e.g., `.mp3`, a *lossless* compressed format, e.g., `.FLAC`, or an uncompressed format, e.g., `.WAV`. While the file formats of audio files are known, the source of a song or song snippet can be anything. A concrete example is a 30-second snippet from Spotify's API being an MP3 file delivered through the Spotify content delivery network, `scdn`, e.g., Jack Harlow's "First Class".

1.2.1 Inputs

- *Music generation:* reference song(s) and/or song snippet(s).
- *Music analysis:* reference song(s) and/or song snippet(s).
- *Music recommendation:* reference song(s).

1.2.2 Outputs

- *Music generation:* generated song or song snippet, likely `.MP3` files.
- *Music analysis:*
  - A collection $\in \mathbb{R}^k$ of features, e.g., pitch, timbre, or "danceability".
  - Visualizations of song characteristics, e.g., spectrograms and spectral density estimates.
- *Music recommendation:* collection of references to songs, e.g., string names of songs or URLs to music providers. Potentially sortable by likeness to the input reference songs. Please note the distinction between *references to songs* and *reference songs*.

1.3 Stakeholders

Project stakeholders are concerned with experimentation in music.

- *Music producers:* professionals looking to generate new ideas, augment their existing works, and experiment with familiar and new genres.
- *Hobbyist musicians:* individuals interested in exploring and tinkering with familiar sounds.
- *Music theorists:* educated users who seek to study, analyze, and experiment with different musical elements, e.g., pitch and rhythm in contemporary blues.
- *Audio engineers:* experts who can use the system to study audio characteristics and rapidly experiment with new sounds to help optimize their workflows.
- *Music educators:* teachers seeking innovative ways to introduce students to music theory.
- *Casual music listeners*: novices who want to discover and generate music for personal enjoyment.

1.4 Environment

We strive to launch an on-premise server operating with a version of Ubuntu server, likely the most recent version, 24.04.1. The server shall respond to and process requests from a web-application front-end, making the service accessible to many different devices, but requiring a network.

2 Goals

The project goals are captured in tabular form below.

Goals	**Importance**
The collective system shall adequately process and respond to requests involving widely-published music genres, e.g., pop, hip-hop, and rock.	Widely-published music genres have the largest corpus of data that can be used to train the featurization and generation mechanisms of the system, i.e., the system must perform favourably in tasks that it is well-trained on.
The generation system shall produce an audio artifact similar to (a) reference song(s) and/or snippet(s) provided by the user.	This is a core feature of the system. Its inclusion should make experimenting with "new" but familiar "sounds" easy.
The analysis system shall generate tabular features that correspond to characteristics of the input song (snippet), akin to those of Spotify, e.g., danceability, instrumentalness, and energy.	This is a core feature of the system. Structured tabular data can be rapidly process, making the task of song recommendation more efficient and song generation more explainable.
The recommendation system shall produce a collection of songs that are similar to a single reference song provided or a collecetion of reference songs provided by the user.	This is a core feature of the system. Its inclusion should facilitate users to explore a music genre or "sound" of interest.
The collective system shall include a clean, intuitive interface to facilitate easy access to the generation, analysis, and recommendation systems. The user interface should require minimal understanding of how music production is conducted.	This is a foundational non-functional goal – a simple and accessible interface appeals to a broad audience, hobbyists and professionals alike.

3 Stretch Goals

Beyond goals that are essential to the project's success, stretch goals are additional opportunities for success if the project is completed early. The stretch goals are captured in tabular form below.

Goals	**Importance**
The system shall adequately process and respond to requests involving *not-as-widely*-published music genres, e.g., jazz, funk, and blues.	Such music genres have a smaller corpus of data that can be used for training, hence the system may not perform as favourably in tasks that it is not very well-trained on, but the inclusion of such genres would allow access to a larger user-group.
The generation system shall allow adjustment of the initial audio artifact produced, altered by editing the tabular features produced by the analysis system.	This is an additional feature of the system. Its inclusion should make the experimentation horizon broader, exposing more music than would be possible by default.
The recommendation system shall allow adjustment of the initial collection of songs produced, altered by editing the tabular features produced by the analysis system.	This is an additional feature of the system. Its inclusion should facilitate users to explore a music genre or "sound" of interest with much more fidelity, again expanding the experimentation horizon.
The system shall generate tabular features that correspond to characteristics of the input song's *cover art*.	Cover art tends to capture, however abstractly, the mood, energy, and intent of a song or album, thus may contain tacit information that can be accessed with image processing.
The system shall generate custom cover art that corresponds to characteristics of the input song's features, e.g., mood, energy, or style.	Cover art abstractly captures the mood and energy of a song, thus producing imagery relevant to a song's character can enhance the user's listening experience.

4 Challenge Level and Extras

The project is of a *general* challenge level.

- It requires domain knowledge about signal (audio) processing, music theory, learning models, generative models, and infrastructure setup.
- Its implementation is non-trivial, incorporating algorithm implementations, training and testing models, assessing their performance, automating the data extraction-transformation-loading workflow and the live-response workflow.
- The system is not particularly novel. Recommender systems are not new, but we are attempting to find and use features to create a better recommender system. The generative component has been done before with images and video, so scaling down to audio and frequency should be attainable, especially as it is a field that was researched quite deeply even before the advent of neural network-based generative techniques.

The project will include extras like user & API Documentation for ease of reference, usability testing for easy startup, and design thinking to build an intuitive user interface.

Appendix — Reflection

The purpose of reflection questions is to give you a chance to assess your own learning and that of your group as a whole, and to find ways to improve in the future. Reflection is an important part of the learning process. Reflection is also an essential component of a successful software development process.

Reflections are most interesting and useful when they're honest, even if the stories they tell are imperfect. You will be marked based on your depth of thought and analysis, and not based on the content of the reflections themselves. Thus, for full marks we encourage you to answer openly and honestly and to avoid simply writing "what you think the evaluator wants to hear."

Please answer the following questions. Some questions can be answered on the team level, but where appropriate, each team member should write their own response:

1. What went well while writing this deliverable?

   One of the things that went particularly well while writing this deliverable was the effective communication and coordination within the team. We maintained regular and open discussions, which helped ensure that everyone was clear on their assigned tasks and responsibilities. We also took the time to review each other's work, offering feedback and suggestions to improve the overall quality of the deliverable. This collaborative effort helped us stay focused and aligned. As a result, we were able to complete a scheduled 2-hour meeting in just 1 hour, demonstrating our ability to work efficiently while maintaining attention to detail.

2. What pain points did you experience during this deliverable, and how did you resolve them?

   The main disagreement, prior to the deliverable, was project selection. To resolve this, we used a project selection matrix, comparing proposed projects using success criteria, discussing results and choosing candidates based on interest level, feasiblity, and difficulty. We used a multi-vote (assign 0, 1, or 2 votes to a project) poll to choose from the final candidate projects, finally selecting GenreGuru. The written documentation allowed neat organization of thoughts on each project and comparison in a fair and honest capacity. We suspected that editing the same LaTeXdocument on GitHub could be an issue, but we resolved it by having each member edit one section of the document, make a pull request, and having one person handling merging all our edits into the final `.tex` file. This helped us stay organized and not cause headaches with merge conflicts.

   During the deliverable, we anticipated challenges, so we approached the task by dividing the workload among our team. Each member volunteered to focus on a specific section while contributing thoughts during collaborative discussions. After completing our individual sections and reviewing each other's work, we convened for a call to share our feedback on what we liked and what needed adjustments. Overall, while the deliverable went

smoothly, we did encounter some pain points. One critical challenge was selecting the right technologies, particularly in determining which frameworks and machine learning libraries to use. To tackle this, each member conducted research to evaluate the strengths of various options, allowing us to narrow down our choices effectively. Another challenge arose from team members' busy schedules, making it difficult to coordinate in-person meetings. To overcome this, we prioritized asynchronous communication throughout the week, ensuring everyone stayed informed. Instead of requiring everyone to attend every meeting, we arranged for a few members to attend and then relay key points to those who couldn't make it. This approach helped keep the entire team aligned and engaged.

3. How did you and your team adjust the scope of your goals to ensure they are suitable for a Capstone project (not overly ambitious but also of appropriate complexity for a senior design project)?

   We decomposed the project into its primary working subsystems, song generation, analaysis, and recommendation, outlining each of their core functionalities and posing those as primary goals. All other functionality is regarded as a stretch goal to allow us to focus on making the core functionality, i.e., the most impactful components of the project, work as best as possible. We ensured the project was of appropriate complexity through the project selection exercise described in the response to reflection question 2.