Development Plan
Software Engineering

Team 8 – Rhythm Rangers

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

Table 1: Revision History

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

[Put your introductory blurb here. Often the blurb is a brief roadmap of what is contained in the report. —SS]

[Additional information on the development plan can be found in the lecture slides. —SS]

1 Confidential Information?

[State whether your project has confidential information from industry, or not. If there is confidential information, point to the agreement you have in place. —SS]

[For most teams this section will just state that there is no confidential information to protect. —SS]

2 IP to Protect

[State whether there is IP to protect. If there is, point to the agreement. All students who are working on a project that requires an IP agreement are also required to sign the "Intellectual Property Guide Acknowledgement." —SS]

3 Copyright License

[What copyright license is your team adopting. Point to the license in your repo. —SS]

4 Team Meeting Plan

Team meetings will be scheduled in a relatively ad-hoc fashion. Members have shared their daily schedules throughout the working week with each other, and weekly availability notes are accounted for when scheduling meetings. Availability notes are to be shared before the start of the working week so a week's meeting plan can be drafted and voted on by the end of Monday. Team meetings can occur for:

- task delegation for upcoming deliverables & discussing deliverable progress

  we hope to make these meetings brief and infrequent in the future by using GitHub Projects and asynchronous communication instead;
- work sessions to collaborate and discuss ideas (deliverable-related) synchronously, *preferably in-person*;
- pair programming;
- conducting deliverable reviews;

- conducting deliverable retrospectives, i.e., reflecting on successful and unsucessful practices used in the most recent deliverable after its completion.

[How will the meetings be structured? There should be a chair for all meetings. There should be an agenda for all meetings. —SS]

5 Team Communication Plan

All team communication is done through a discord server. The discord has three text channels:

- General (anything not project-related is posted here)
- Important Updates (anything related to weekly meeting availability goes here)
- Locked In (anything related to project work goes here)

The discord also has 2 voice channels:

- Weekly Meeting (administrative meetings happen here)
- Locked In (collaboration of project development happens here)

6 Team Member Roles

- Ansel Chen: Team liaison, developer
- Muhammad Jawad: Developer
- Mohamad-Hassan Bahsoun: Developer
- Matthew Baleanu: Developer
- Ahmed Al-Hayali: Meeting manager and scheduler, developer

7 Workflow Plan

7.1 General Workflow

- Issues are created, assigned, and attached to the project. Issues will have a template akin to those find on stevemao/github-issue-templates. These issues should pertain to deliverable sections, split into a completion assignee and a reviewer;
- The `main` branch is protected, so team members must work in independent branches. We hope to restrict branch naming to a standardized format, e.g., this or akin to conventional commits (which comes with a linter!);

- Team members' commits must follow the conventional commits standard;
- Whenever necessary, a team member can choose to merge their changes to the `main` branch with a pull request. Pull requests should have a template akin to the simple pull request found here or the more complicated data-centric template from dbt. A pull request should be attached to the GenreGuru Project on GitHub with a *dedicated* reviewer and potentially a review timeline and checklist. Having only one reviewer avoids the issue of diffusion of responsiblity.

7.2 Usability Testing

- How will you be using git, including branches, pull request, etc.?
- How will you be managing issues, including template issues, issue classification, etc.?
- Use of CI/CD

8 Project Decomposition and Scheduling

- How will you be using GitHub projects?
- Include a link to your GitHub project

[How will the project be scheduled? This is the big picture schedule, not details. You will need to reproduce information that is in the course outline for deadlines. —SS]

9 Proof of Concept Demonstration Plan

There are two main risks to the project — the song data collection and song generation.

9.1 Risks regarding song collection

- License acquisition may be necessary for some songs;
  - Acquisition of the song in general may require a license from the artist, label, publisher, or platform, e.g., Spotify;
  - Platform providing songs may have limited API access.
- Songs may be only partially accessible, e.g., song snippets from Spotify;
- Songs may be available but we are prohibited from using them to train a machine learning model.

These risks can be dismissed if the project is to use different, less strict, song providers, or tailor the project to only use non-copyrighted songs.

9.2 Risks regarding song generation

- The generative mechanism will inherently be a machine learning model, which entails issues,
  - The model may hallucinate and produce unexpected outputs, i.e., music of the wrong genre (particularly of concern if training data is unbalanced), or just uncomfortable nonsensical sounds. *This could be a result of too little data to train a complex model (resulting in high variance), or too simplistic of a model (resulting in high bias)*;
- The model will be challenging to formulate, e.g., establishing architecture, objective function, and optimizer;
- The model will be so complex, i.e., will contain many parameters, such that it requires tremendous quantities of data and training time to converge to sensical results.

These risks cannot be entirely dismissed, but can be remedied greatly by considering the work of previous similar works and following their process, i.e., reusing architecture, data, or training mechanism. Nonetheless, this project is doable, as a parallel of it was completed in 2017.

10 Expected Technology

The technologies and tools expected for this project include:

- **Programming Language:** Python, due to its vast libraries in machine learning and audio processing, such as `librosa` and `pydub`.
- **Libraries:**
  - `librosa`: For music and audio analysis.
  - `pydub`: For audio processing and manipulation.
  - `scikit-learn` and `TensorFlow`: For building machine learning models to classify and generate music.
- **Frameworks:**
  - `Flask` or `Django`: For the web-based interface, allowing users to interact with the system.
  - `PyTorch` or `TensorFlow`: For implementing deep learning models to generate and classify music.
- **External APIs:** Spotify API will be used for fetching song previews, features, and other metadata for recommendation purposes.

- **Pre-trained Models:** We may leverage some pre-trained models for audio generation, such as OpenAI's Jukebox or similar publicly available models, while customizing them to fit our needs.
- **Linters:** a CI-integrable Python-specific linter, e.g., `pylint`, `ruff`, or `flake8`. There also are git-specific linters like conventional commit's linter.
- **Formatter(s):** a CI-integrable formatter formatter, e.g., `black`.
- **Type-checker(s):** a CI-integrable type-checker, e.g., `mypy`.
- **Documentation Generation:** a CI-integrable documentation generator, e.g., `sphinx`.
- **Tester(s):** a CI-integrable testing framework, e.g., `PyTest`.
- **Continuous Integration:** the 5 previous "CI-integrable" components are a good baseline, but we plan on using a local server, and deployment onto it with CI would be sweet. As much automation with CI, within reason, is desirable as it is excellent experience to be transferred to a working setting.
- **Environment Management:** environment managers for Python like `pipenv` and `poetry` are excellent for ensuring the program works the same everywhere, installation becomes straightforward, and CI becomes marginally easier because of more standardization.

[git, GitHub and GitHub projects should be part of your technology. —SS]

11 Coding Standard

The coding standards will be a series of PEPs, but importantly PEP 8 for coding style and PEP 257 for docstrings that will form the basis for the API and user reference. For simplicity, a package-like folder structure should be used in conjunction with the provided template.

12 Project Scheduling

There will be no GANTT charts - we will conduct weekly meetings as "standups" and decide what is to be worked on. Please refer to section 1, "Team Meeting Plan" for details.

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. Why is it important to create a development plan prior to starting the project?
   - Creating a development plan is crucial for a few reasons:

   - It gives the project a clear direction and scope since all goals are outlined
   - It sets boundaries on the project to prevent unplanned expansions
   - It outlines anticipated challenges and contingency plans
   - It provides transparent communication of the project expectations to the stakeholders

2. In your opinion, what are the advantages and disadvantages of using CI/CD?

   CI/CD is a wonderful tool that automates mind-numbing tasks of linting, styling, formatting, deployment, and document generation, for example, but it comes at the heavy cost of relatively complicated setup process for said CI/CD to work. Thankfully, there are many GitHub actions templates online, so we often need not to worry about starting from scratch.

3. What disagreements did your group have in this deliverable, if any, and how did you resolve them?

   There were few disagreements, but the frequency of meetings is a hotly-contested topic. We chose to have relatively frequent meetings during the first 3 weeks of the project, will experiment with far fewer in the future and rely on asynchronous communication. Depending on the success of it during the next few weeks, we will continue or adjust it to better suit our needs in completing future deliverables succesfully.

Appendix — Team Charter

[borrows from University of Portland Team Charter —SS]

External Goals

- Make a project that can be but on a resumé
- Learn software engineering industry standards
- Follow the engineering process
  - Draft documentation
  - Conduct research
  - Research Documentation
  - Prototyping
  - Implementation
- Learn signal processing
- Have a project we can work on after graduating
- Impress peers (interviewers, fellow peers at the EXPO)
- Gain familiarity with development tools and frameworks

Attendance

Expectations

[What are your team's expectations regarding meeting attendance (being on time, leaving early, missing meetings, etc.)? —SS]

- Administrator schedules meeting (time, location)
- Administrator outlines meeting agenda
- Administrator creates team meeting issue
- Administrator completes meeting minutes
- Team members are expected to attend most work sessions
- All team members are expected to attend TA deliverable feedback meetings
- All concerned team members should attend necessary deliverable check-in meetings

- Administrator must give 'reasonable' advanced notice to meeting attendees (at least 24 hours in advance unless team members agree to scheduling a time earlier)
- All members are expected to attend and take initiative in the issue provision process when delegating work
- Members to attend a meeting during the scheduled lecture or tutorial times (except for Friday) if that meeting is arranged before 10pm the night prior

Acceptable Excuse

- If the task assigned to a team member is independent of others' contributions, the group can elect the member to miss the work session

In Case of Emergency

- If an emergency arises for a member, they are responsible for delegating their tasks to be completed by others. However, if those tasks are not completed, the person is responsible for repurcussions
- If an emergency arises for a member, they must notify the group on discord as soon as possible, and they must reach out to that week's administrator to catch up on the missed meeting

Accountability and Teamwork

Quality

[What are your team's expectations regarding the quality of team members' preparation for team meetings and the quality of the deliverables that members bring to the team? —SS]

- The meeting administrator must prepare the meeting agenda and attach it to the meeting issue at least 2 hours before the scheduled meeting time
- For a work session, the administrator must prepare a list of tasks to be completed before and during the work session
- For a work session, the atendees are expected to complete the necessary work prior to arrival and update the administrator on tasks to be completed during the work session
- For a check-in, the administrator must prepare a list of tasks to be completed before and during the check-in
- For a check-in, the atendees are expected to complete the necessary work prior to arrival and update the administrator on tasks to be completed during the check-in

- For a delegation meeting, the administrator must prepare a list of issues that must be delegated
- For a delegation meeting, the atendees are expected to take initiative and assume responsibility for all presented issues
- The member who is delegated an issue should look into the deliverable checklist and rubric on avenue, then generate a checklist for the PR reviewer to consult as they review the PR

Attitude

All members are expected an enjoyable work environment.

Stay on Track

- The github project board and meetings will keep the team on track
- Issue completion and status checks during meetings will ensure that members contribute as expected and that the team performs as expected
- Verbal affirmations will boost team morale after completing issues at a high quality
- Recurrent sub-expectation performance from a member will be raised at the following delegation meeting
- If a team member recurrently under-performs and fails to meet consecutive check-ins without justification, their performance will be raised to the TA and professor Smith

Team Building

The team will sometimes have a team outing (e.g team lunch, dinner, gaming session)

Decision Making

Decisions will primarily be made through a discussion, followed by a consensus. If that cannot be achieved, a vote will be cast where the majority rules