Development Plan
ScoreGen

Team #7, Tune Goons
Emily Perica
Ian Algenio
Jackson Lippert
Mark Kogan

Table 1: Revision History

Date	Developer(s)	Change
24/09/2024	Jackson, Ian, Emily, Mark	Initial revision
Date2	Name(s)	Description of changes
...	...	...

[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?

There is no confidential information needing protection for our project.

2 IP to Protect

There is no IP to protect for our project.

3 Copyright License

Our team will be adopting the MIT License, which is standard for allowing usage without limitation to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software. This is due to the nature of our project, which is focused on learning and not for commercial use.

4 Team Meeting Plan

[How often will you meet? where? —SS]

[If the meeting is a physical location (not virtual), out of an abundance of caution for safety reasons you shouldn't put the location online —SS]

[How often will you meet with your industry advisor? when? where? —SS]

[Will meetings be virtual? At least some meetings should likely be in-person. —SS]

[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

[Issues on GitHub should be part of your communication plan. —SS]

6 Team Member Roles

[You should identify the types of roles you anticipate, like notetaker, leader, meeting chair, reviewer. Assigning specific people to those roles is not necessary at this stage. In a student team the role of the individuals will likely change throughout the year. —SS]

7 Workflow Plan

- 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

What is the main risk, or risks, for the success of your project? What will you demonstrate during your proof of concept demonstration to convince yourself that you will be able to overcome this risk?

10 Expected Technology

[What programming language or languages do you expect to use? What external libraries? What frameworks? What technologies. Are there major components of the implementation that you expect you will implement, despite the existence of libraries that provide the required functionality. For projects with machine learning, will you use pre-trained models, or be training your own model? —SS]

[The implementation decisions can, and likely will, change over the course of the project. The initial documentation should be written in an abstract way; it should be agnostic of the implementation choices, unless the implementation choices are project constraints. However, recording our initial thoughts on implementation helps understand the challenge level and feasibility of a project. It may also help with early identification of areas where project members will need to augment their training. —SS]

Topics to discuss include the following:

- Specific programming language

- Specific libraries

- Pre-trained models

- Specific linter tool (if appropriate)

- Specific unit testing framework

- Investigation of code coverage measuring tools

- Specific plans for Continuous Integration (CI), or an explanation that CI is not being done

- Specific performance measuring tools (like Valgrind), if appropriate

- Tools you will likely be using?

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

11 Coding Standard

Our source code will adhere to the following widely adopted coding style guides:

- For **C++**, which will likely be our main coding language for the signal processing, we will follow Google's C++ Style Guide due to its widespread adoption.

- For **Python**, we will adhere to the PEP8 Style Guide.

- For **HTML** and **CSS**, we will implement the BEM Methodology.

All functions, classes, and modules must be well-documented using inline comments and docstrings where appropriate. Comments should provide context when necessary, rather than stating the obvious. This ensures clear communication and explanation of code between team members, making our pull requests easy to follow and the codebase more maintainable. Additionally, all pull requests must be reviewed by at least one other team member before being merged into the main branch. Reviews will focus on code quality, adherence to coding standards, and potential performance improvements.

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?

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

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

Appendix — Team Charter

External Goals

Our team's primary external goal for this project is to maximize learning and skill development in software engineering. We are focused on creating a technically impressive and innovative project to earn the highest possible grade. Additionally, we want to ensure our project can be showcased in interviews and added to our professional portfolios and résumés. These goals will help us stand out when pursuing future opportunities in the tech industry.

Attendance

Expectations

Team members are expected to attend all scheduled meetings on time within reason. Leaving early is acceptable if communicated in advance and if that member has completed their contributions for the meeting. Additionally, all team members may leave if they feel the meeting's agenda has been completed. Missing meetings should be a rare occurrence, and any absences must be communicated at least 24 hours in advance when possible.

Acceptable Excuse

Acceptable excuses for missing a meeting or deadline include illness, family emergencies, or unavoidable academic conflicts (exams or critical project deadlines). Unacceptable excuses include forgetfulness, oversleeping, or personal plans that were not communicated ahead of time. All excuses must be communicated as soon as possible to the rest of the group.

In Case of Emergency

If a team member has an emergency and cannot attend a meeting or complete their work, they should notify the team as soon as possible via the agreed-upon communication method. If possible, the team member should provide updates on their progress and share any work that has been completed so far, so others can step in if necessary. In case of an emergency during critical project milestones, the team will redistribute tasks to ensure deadlines are met.

Accountability and Teamwork

Quality

All deliverables should be thoroughly tested, documented, and meet the project's technical and design standards before being submitted to the team. If a team member encounters difficulties, they should raise a GitHub issue for discussion before the meeting to avoid delays and ensure early feedback. Pull requests

raised to change code will be reviewed by at least one other team member, ensuring only quality code is being deployed via CI. Finally, all team members will look over and edit each others' work for documentation, ensuring everything is clear to the reader.

Attitude

We will foster a positive, supportive, and professional team environment where everyone's ideas are heard and respected. During discussions, we will prioritize constructive feedback and aim to solve problems together. In every interaction, we will emphasize respect, inclusion, and clear communication. Any conflicts will be addressed through open discussions. If unresolved, a neutral team member may mediate, or the issue will be escalated to the TA.

Stay on Track

For progress tracking we will use GitHub's issue tracking and Kanban board features to manage tasks, ensuring that all assignments are transparent and visible to the team. Each team member will update the status of their tasks by moving issues across the Kanban board into their respective swim lanes. GitHub commits should reference corresponding issues, ensuring that work is traceable. A meeting chair is appointed to keep meetings on schedule and ensure the meeting agenda is followed.

Team Building

To strengthen our teamwork, members are encouraged to engage with each other informally outside the scope of this project. As a team ritual, we will celebrate milestones by acknowledging successes and accepting every failure gracefully and as a team.

Decision Making

We will aim for consensus in decision-making. If a consensus cannot be reached within a set timeframe, we will hold a majority vote. For major disagreements, we will discuss the pros and cons of each option and if needed, consult a TA, our supervisor, or the professors of the course.