Development Plan
Software Engineering

Team 1, BANDwidth
Declan Young
Ben Dubois
Nathan Uy
Aidan Mariglia

Table 1: Revision History

Date	**Developer(s)**	**Change**
Sep 24th, 2024	Aidan Mariglia, Declan Young, Benjamin Dubois, Nathan Uy	Adding initial information
April 02, 2024	Aidan Mariglia, Declan Young, Benjamin Dubois, Nathan Uy	Refining Development Plan

This report contains information related to the development plan of our project. Included in this report is information regarding team operation, scheduling, workflow, and expected technologies

1 Confidential Information

The development of the project does not involve any confidential first-party information from the industry.

2 IP to Protect

The development of the project does not involve any protected IP.

3 Copyright License

The project will be licensed under the MIT open-source license.
License

4 Team Meeting Plan

The team is expected to meet virtually through Microsoft Teams at least once every week. Progress meetings will be scheduled for 30 minutes and collaboration meetings will be scheduled as needed. The team will meet with Dr. Phil virtually once every other week to update on progress and ask for any clarifications.

Meeting agendas will be formed throughout the week leading up to the meeting. Rather than spending time answering non-urgent questions/concerns, these will be collected and addressed at the meeting. Meetings will review current progress, challenges, next steps, task assignments, and any relevant questions. ~~The chair of the meeting will rotate weekly between team members.~~

5 Team Communication Plan

The team will use Microsoft Teams as the primary form of communication to coordinate deliverables and discuss any design decisions that need to be made, or any issues that are encountered during development. Discussions will be held with this platform to ensure that there is transparency between all team members and the supervisor to avoid any confusion or miscommunications with the expectations for the project.

Additionally, GitHub issues will be used by the team members to coordinate what work needs to be completed, the deadlines for these issues and to evenly divide tasks between team members. All issues that are created on GitHub must follow the correct template so that they contain all of the necessary information

for the task to be completed. These fields include a description and acceptance criteria, as well as the assignee(s) (who will be working on the task), the related milestone (containing the details regarding the deadline and related work) and a label (describing the purpose of the issue, such as to fix a bug or add a new feature).

Following this communication plan will ensure that the team and supervisor can work together efficiently and effectively throughout the project. It will do this by providing a clear communication process within the team so that all members have input and are aware of major decisions/problems that are discussed and that no work is duplicated.

6 Team Member Roles

The chair of the meeting will rotate weekly between team members.
Our approach is adaptable, allowing team members to take on different roles as necessary rather than following rigid role assignments.

Role	**Responsibilities**	**Team Member**
Project Lead	Create issues, Review pull requests, Lead meetings, Get progress updates	Declan Young
Reviewer	Review all documentation and pull requests to ensure completeness and correctness, Inform the original author if any issues are found, Run/verify code changed in pull requests	Aidan Mariglia, Declan Young, Benjamin Dubois, Nathan Uy
Developer	Create issues, Implement features, Test features, Close issues	Aidan Mariglia, Declan Young, Benjamin Dubois, Nathan Uy

7 Workflow Plan

Git

Git will be used for version control for all code, documentation and any other related files for this project.

Issues

GitHub issues will be created for tracking various project items, such as meetings, lectures and development tasks. Any issues that are created should be

done using the correct template, so that all issues of a certain type follow the standard format as specified here: Issue Templates, and contain all the necessary details mentioned in the template ~~related to the issue~~, such as a description, deadline and issue size (in the case of a development task). Team members can then assign tickets to themselves when they are available to take on a piece of work, and will keep the ticket updated throughout the duration of the work.

Issues will have the types the following types:

- Lecture
- Peer Review
- Supervisor Meeting
- TA Meeting
- Team Meeting
- Development Task

Branches

If a piece of work requires changes to any of the files in the GitHub repository, the team member should create a branch specifically for these changes that are being made. Branches should follow the naming convention ¡issue_type¿/¡issue number¿-¡issue description¿ so that the purpose of the branch is made very clear.

Pull Requests

Once the work related to the issue has been completed, a pull request can be made on GitHub. Pull requests should include a summary of the changes, how the change was tested, as well as a description of how to run the affected code. A minimum of 1 approval will be required before merging pull requests.

CI/CD

GitHub actions will be used as the CI/CD provider of choice. CI/CD will involve running the autopep8 linter and unit tests upon PR creation and deploying to a staging environment upon PR merge.

8 Project Decomposition and Scheduling

The team will be using the backlog feature within the GitHub Project to track the status and priority of development tasks, documentation, and deliverables. When issues are first created by a member of the team, they will be placed in the 'Backlog' state. When this new issue has been reviewed by another member of the team and approved, the issue will move to the 'Ready' state.

Then, when a member of the team picks up the issue and begins working on it, the issue will be moved to the 'In progress' state. When the issue has been completed by a member of the team, it will be moved to the 'In Review' state to be reviewed by another member before it is complete. Lastly, once the issue has been reviewed, it will move to the 'Done' state.

The team will also be using milestones on the issues that are created to sort the issues according to the project deliverable they correspond to. For the project components outlined in the table below, the team has created a new milestone and when issues are created they will be added to the required milestone. For example, for the Proof of Concept Demonstration, the team will need to bootstrap the web server, so a new issue under the 'Proof of Concept Demonstration' milestone named 'Bootstrap Webserver' is created. This allows tracking the progress of all deliverables easily. Github Project.

Project Component	**Deadline Date**
Requirements Documentation Revision 0	October 23rd
Hazard Analysis 0	October 23rd
V&V Plan Revision 0	November 1st
Proof of Concept Demonstration	November 13th
Design Document Revision 0	January 15th
Final Demonstration Revision 0	February 13th
V&V Report Revision 0	March 7th
Final Demonstration Revision 1	March 24th
EXPO Demonstration	April TBD
Final Documentation Revision 1	April 2nd

9 Proof of Concept Demonstration Plan

The main risks of our project involve the execution of Matlab code in a cloud environment. More specifically our concerns are that matlab licensing which would allow us to autoscale the number of containers executing user input code does not exist/would be prohibitively expensive.

A possible solution to this is to use an open-source alternative, GNU Octave, however, this poses risks as well. GNU Octave seeks to be a superset of the Matlab language, however, that does not mean that it is a replacement with perfect compatibility in every situation.

Another risk would be the potential costs of hosting our project using cloud technologies. To show we can overcome these risks with our POC we will demonstrate executing Matlab code on our cloud backend, using either a licensed version of Matlab or GNU Octave, and recording and extrapolating the cloud hosting costs involved.

10 Expected Technology

- Django will be used as the web framework for its scalability, built-in features such as ORM and authentication, and rapid development capabilities.
- Python will be the programming language of choice for the project's backend. Python is widely used, and as a result, offers many useful frameworks and community resources. It is very effective for creating website backends and communicating with cloud platforms making use of SDKs.
- ~~Boto3 is the AWS SDK for Python. Making use of this SDK will allow for the required dynamic cloud orchestration and scaling.~~
- ~~React is a widely used and well-supported front-end framework that allows for the creation of reusable front-end components. It will allow for the creation of a usable and reliable front end.~~
- ~~AWS is a powerful cloud platform with a large variety of services, it is more than adequate for the use case in mind.~~
- ~~Fast Api will be the backend framework of choice, it is efficient and barebones, allowing for the flexible creation of web servers.~~
- ~~Uvicorn is an application server that uses the ASGI protocol, making it compatible with Fastapi. It handles many of the concerns involved with running a web server such as replication, and worker management. It will be an efficient choice for this application~~
- ~~NGINX is an extremely popular and efficient web server. It will sit in front of Uvicorn and act as a reverse proxy.~~
- Celery will be used for task queue processing, enabling asynchronous execution and efficient workload distribution.
- Docker is a product that supports building and executing containers, simplifying development and deployment. It is also a good choice for executing our user input code in the backend, due to the ease of scaling docker containers, and the intrinsic isolation provided by using containers.
- Matlab/GNU Octave is required for the execution of user-submitted code. Ideally, Matlab will be used but in the case that licensing/executing headlessly is not possible octave may be used.
- Git will be used as the version control software of choice
- GitHub will be used to store project code
- GitHub (actions) will be used as the CI/CD solution of choice.

- Make is a useful developer tool, and can be utilized to simplify running tests/linting code
- Bash is a common shell/shell language. Scripts written in bash will be utilized throughout the ci/cd pipeline and for developer productivity.
- ~~Pytest~~ unittest is a powerful unit testing framework for the Python language. It will be used for unit testing code
- ~~Postman is an HTTP API platform that can be used for writing API test cases, it will aid in integration testing of the platform~~

11 Coding Standard

~~The PEP8 standard for Python will be followed. To ensure that code is implemented following the standard, pre-commit hooks will be utilized to run the autopep8 linter.~~

Appendix — Reflection

Nathan Uy Reflection

1. It is important to create a development plan as it outlines all the expectations of the group members clearly and it is something that everyone can refer back to throughout the project in case of any conflicts/disagreements.

2. CI/CD allows for an easier and smoother workflow as it automates building, linting, testing, and even deployment. It will ensure potential issues in the code will be flagged early in the process, saving developers time and potentially money. A disadvantage to using CI/CD is the amount of work needed to set it up at the beginning of the project.

3. As our team has worked together numerously in the past already and we seem to have agreed on every decision so far, we have not had any disagreements in this deliverable.

Declan Young Reflection

1. It is important to create a development plan before starting the project, as it ensures that the project will start off on the right track and will avoid having to spend additional time and effort keeping the project coordinated and organized between different team members. Additionally, it will give a set of guidelines and best practices in order to keep the project

2. The advantages of using CI/CD are that it will greatly reduce the number of bugs that are introduced for each change that is merged, while also keeping the turnaround time for all changes relatively low, so that new features and bug fixes can quickly be quickly and safely deployed. There are some disadvantages, however, the first of which being that there can be some potentially large costs (in money, time and resources) to setting up a good CI/CD environment. Additionally, people may use the CI/CD environment as an excuse to not properly test the change that they made to code. It is important that changes are still tested, which includes updating the CI/CD environment to properly test any new or refactored code.

3. Our team did not have any disagreements regarding this deliverable.

Ben Dubois Reflection

1. It is important to create a development plan prior to starting a project because it helps to remove any ambiguity for the remainder of the project. The development plan helps to make crucial decisions before the project begins to ensure that it can be successful later. Also, the development plan helps to mitigate the chances of conflict in the future as it contains written rules that can be referenced in the future.

2. A large advantage of CI/CD is the ability to automate tests and quickly fix any bugs that are introduced during the development process. Bugs can be very annoying to deal with if a large amount of code has just been pushed and the cause of the bug is unknown. Therefore, using the automated tests and only pushing small code changes at a time allows for bugs to be easily identified and solved. A potential disadvantage is that frequently pushing small code changes may lead to a more time-consuming development process.

3. There were no major disagreements while working on this deliverable. There were a few moments of differences in opinion, but these were quickly resolved after more explaining.

Aidan Mariglia Reflection

1. It is important to create a development plan prior to starting the project to ensure that all team members are on the same page in terms of processes. Over the course of the project, this causes large efficiency gains, and reduces wasted time, ensuring all members can focus on what is important.

2. In my opinion, the largest benefit of CI/CD is preventing broken code from entering the main development trunk through the use of automated unit tests. Ensuring the integrity of the code is extremely important for a smooth development process. Additionally, automatic deployments reduce user error that may be present in manual deployments.

3. There were no serious disagreements regarding the deliverable, we deliberated somewhat over which technologies to use but amicably came to a final agreement.

Appendix — Team Charter

External Goals

One of the main goals of this project is to create a product that will be used in a real work setting. It will aid users in testing their battery algorithms stress-free and efficiently. In addition, the team's goal is to create well-written detailed documentation for this project so that stakeholders will always have something to refer back to in the future. As a result, the team will be able to talk to future employers regarding this project which involves working with industry stakeholders to identify an existing problem and producing a well-documented and well-designed working solution.

Attendance

Expectations

All team members are expected to arrive at least five minutes before every meeting to ensure a timely start. If a member needs to leave early, they should inform the team at the beginning so that all discussions relevant to them will be prioritized. Additionally, the team must be notified at least one day in advance if anyone cannot attend a meeting to allow for necessary arrangements. During meetings, it is very important that everyone stays focused on the task at hand and contributes to the discussion.

Acceptable Excuse

The team will be more lenient on missing a meeting, except for meetings where crucial design decisions will be made. Acceptable excuses for missing a meeting include sickness, injury, death of a family member/family emergencies, prior commitments, or celebration of personal events with family. Since each team member is given sufficient time to work on a task, only long-term illness (lasting a week or more)/family emergencies are accepted as excuses for missing work assigned.

In Case of Emergency

In the case of a team member missing a meeting, the team will try to accommodate the person by rescheduling to a time when everyone is available. Otherwise, a written summary of what was discussed during the meeting will be sent to the person who missed the meeting. If work is missed due to an emergency, the team will create an emergency meeting to discuss how the missed work will be shared among the team and set a deadline for it.

Accountability and Teamwork

Quality

Team members are expected to arrive at every meeting prepared and familiar with the agenda. Preparation includes understanding the next milestone and our current progress against the schedule, completing any assigned tasks from previous meetings, and being ready to ask/answer questions regarding his or others' work. Regarding the quality of deliverables, all work must be verified by at least one other team member to ensure quality and cohesiveness. Any submitted code should be fully tested (manual, unit, integration), adhere to the team's coding standards, and comply with the design specifications.

Attitude

Open communication and respect among team members are crucial to the team's success. Everyone is encouraged and expected to contribute their innovative ideas to the team. No idea is a bad idea and each proposed idea will be carefully considered and constructively revised. It is also expected that team members collaborate effectively by helping one another. Conflicts will be resolved through open communication, with the team coming together to reach a solution that will satisfy all parties involved.

Stay on Track

The team will create a Google calendar with all the deadlines for course deliverables as well as the team's weekly milestones. The weekly meetings will ensure transparency and that everyone is held accountable for their work. Quality and timeliness will be assessed during these weekly meetings. The team will also discuss any shortcomings and acknowledge any individual's exemplary work. Shortcomings from any team members will be discussed, specifically, what caused the failure to deliver and what actions the team could take to avoid drawbacks in the future. Moreover, support and constructive feedback will be provided to help the individual improve and prevent future deficiencies. If poor performance continues to happen from the same person, the team will make an appointment with the TA or professor to discuss possible solutions.

Team Building

In order to build team chemistry and boost morale, a bi-weekly team bonding event will be organized. We will create a schedule to determine which team member is responsible for planning each team bonding event. Team bonding events will differ each time, but could include activities such as online games and karaoke.

Decision Making

Decisions will be made by carefully weighing the pros and cons of each solution. Then, the team will reach a consensus on the best way to move forward. It is expected that team members show professionalism and respect for one another. Any disagreement should be addressed through open communication, focusing on arriving at a collaborative resolution.