Development Plan

Table 1: Revision History

Date	Developer(s)	Change
September 25, 2023	Q.H, R.V, D.A, D.C, U.R	Competed Development Plan Document

1 Team Meeting Plan

The team will hold weekly one hour meetings on Wednesdays at 12:30pm for project updates, questions, and concerns. Ad-hoc meetings will follow as needed, as agreed upon by team members. All team members agree to attend weekly meetings, unless unforeseen situations arise. Should a team member not be able to attend, adequate notice will be given, and it is that team member's responsibility to ensure they provide their updates, and remain on track.

In addition, bi-weekly 30 minute meetings will occur with stakeholders on Mondays at 12:30, beginning October 2. This meeting will be for updating stakeholders, gathering requirements, and asking questions as needed.

2 Team Communication Plan

All official team communication will be done through the "Lower Earth Orbit" (LEO) discord server. This will be the platform for daily communication, weekly team meetings and the main medium for contact.

Communication with stakeholders will take place via email when needed. It's each member's individual responsibility to communicate effectively and consistently to ensure a seamless project experience.

3 Team Member Roles

This is a preliminary description of the roles of the group members. During the period of the course and project, more roles and responsibilities will be added

and current roles may change.

Name and Role	Responsibilites
Buu Ha (Quinn) - Stakeholder Manager and Developer	• Will be in charge of handling communications with the stakeholders. • Will be an active contributor in the coding, testing, reviewing and documentation process.
Rishi Vaya - Repository Maintainer and Developer	• Will maintain the project repository and be in charge of code conflicts and merging. • Will be an active contributor in the coding, testing, reviewing and documentation process.
Umang Rajkarnikar - Infrastructure Maintainer and Developer	• Will set-up/maintain the tools used during the project including Trello, Databases and Authentication platforms. • Will be an active contributor in the coding, testing, reviewing and documentation process.
Diamond Ahuja - CI/CD Manager and Developer	• Will ensure an ongoing flow of new features and bug fixes into production in an efficient manner. • Will be an active contributor in the coding, testing, reviewing and documentation process.
Dhruv Cheemakurti - Course Liaison and Developer	• Will be in charge of handling communications with the professor and TAs. • Will be an active contributor in the coding, testing, reviewing and documentation process.

4 Workflow Plan

4.1 Git Workflow

For this project, Git is the version control system that will be used to manage and update source code history, and Github will be used to manage Git repositories. In addition, the Git Feature Branch workflow will be adopted to manage

all feature development. This way the development of new features will be in isolated branches, providing a layer of encapsulation which in turn ensures that the central (main) branch's state is preserved.

4.2 Pull Requests (PR)

Collaborators will create pull requests (PRs) to merge their featured development into the central (main) branch. This is to ensure that their changes do not present any conflicts or other challenges which may compromise the state of the central branch. Upon opening a PR, a rule will be set in place such that it requires approval from at least one other collaborator before the changes can get merged in.

4.3 Issues

Github Issues will be used to track and manage tasks. Tasks can include bug tracking, feature requests, documentation, and testing. For each Github Issue, a minimum of one assignee will be added to each task. In order to standardize and streamline the process, custom issue templates will be used such that contributors can provide necessary information when reporting an issue. Additionally, issues will be tracked and referenced in the corresponding pull request (PR) that resolves the issue.

4.4 Milestones

Github Milestones will be used to organize and keep track of the project's progress throughout its life cycle. Each milestone will be used to group issues and PRs related towards a common goal. This tool will be used to measure the progress and assess the status of the project objectives. Once the deadline of a milestone has reached, the recorded progress will be referenced during the subsequent team meeting for feedback and workflow evaluation.

4.5 Labels/Tags

Labels will also be used in Github to group the various types of tasks. For example, bugs and features are two separate labels which can be assigned to a task. Furthermore, labels will also be used to categorize tasks in terms of priority. This allows the team to focus on the critical tasks and be in alignment of the project's goals.

The following are a list of tags which will be incorporated in future workflows, however may be changed or modified throughout the course of this project:

- Bug - Groups issues for software bugs
- Feature - Groups tasks for new development features

- Frontend - Groups tasks related to changes in the user interface of the application
- Backend - Groups tasks related to changes in the application's server side development
- Testing - Groups tasks for testing the application
- CI - Groups tasks that will modify the continuous integration pipelines

4.6 Project Management Tool - Trello

The project management and collaboration tool, Trello will be used to help organize tasks and assignments for this project. It is a ticket/card-based system which allows teams to separate tasks at different stages of development, where each card represents a task. These cards can also be moved across different stages of the development cycle. The Trello board will have lists, which will be used to group related cards together. For example, all cards pertaining to testing will fall under the Testing list. Similar to Github Labels, labels will also be attached to the cards. These labels can reflect some categorization such as priority or the type of task.

4.7 Contributor's Guide

A Contributing Guide file will reside in the root level of the repository to communicate best practices when contributing to the project. This file will have guidelines on creating issues and PRs, as well as external links that reflect relevant industry-recognized coding guidelines.

5 Proof of Concept Demonstration Plan

The project's main risk is the **orbital prediction accuracy**. NEUDOSE CubeSat, McMaster Interdisciplinary Satellite Team's first-ever launched satellite, struggled with the communication and the execution of commands. The satellite orbits earth up to 16 times per day, but only pass over the Hamilton sky 3-5 times during a day. This can be problematic as often these overpasses are not ideal because their maximum elevations are far from the center of the sky, where the signals are the strongest. Additionally, operators may not be available during the suitable passes, which can happen any time during a day. This number decreases further for the PRESET CubeSat as it follows Sun-Synchronous Orbit (SSO). The number of access windows for the McMaster ground station is currently unknown.

In addition to using the orbital prediction software such as Gpredict, N2YO, GMAT, and PyOrbital, there are a few methods and techniques that can be applied to overcome this risk.

1. **Comparing prediction with historical data**: Setting a target satellite, such as NEUDOSE, over a defined period and comparing the predicted passes with the historical passes to determine the accuracy of the prediction. Creating a record for any discrepancies between the results of pass times and elevations.

2. **Real-time testing**: This can be achieved by predicting passes for the upcoming weeks or month. System's predictions should be adjusted based on the data gathered from real-time predictions and actual passes.

The steps mentioned above can be used to demonstrate the risk being mitigated. However in the web-application, a built-in orbit prediction software will be used which is based on SatNOGS. This mean integrating the methods could be a challenging task. Additionally, due to PRESET CubeSat's change in orbit, the number of passes where the signals are the strongest, is reduced.

The resulting POC will be a bare-bones web app for while a user can track a satellite given their TLE (Two Line Element Set), track future overpasses, and schedule/send commands to the ground station server. As the ground station server may not been fully configured by MIST by the POC deadline, a mock server will be created to mimic connection to the ground station.

6 Technology

6.1 Development Stack

- Front-End: Typescript, Next.js
- Back-End: Node.js
- Database: MongoDB

The coding languages above were chosen for a multitude of reasons, including strong typing, industry standards, and performance reasons. Firstly, utilizing a strongly typed language helps catch type-related errors at compile time (vs runtime), for increased code quality. In addition, a strongly typed language makes it easier for multiple parties to quickly understand the code, and build on top of it. Next, all of the tools are commonly used in industry-levelled code for its robust, yet flexible capabilities, and can be leveraged in our capstone application. In addition, a combination of Next.js and Node.js provides optimized loading strategies on the server-side, which is crucial for when MIST wants to maximize usage when NEUDOSE flies overhead. It's also worthwhile to note that this stack is also well documented, and a popular choice amongst developers, hence a healthy amount of documentation and support is present.

6.2 Linter

- ESLint (extension available on VS Code)

ESLint was chosen as a linter for it's reputation amongst developers as an industry standard. In addition, a VS Code (code editor of choice) extension is available for ease of use and integration.

6.3 Testing and Code Coverage

- Unit Testing Framework: Jest
- Code Coverage Measuring Tool: Istanbul(nyc)

Jest and Istanbul (with nyc library) were chosen for the unit test framework and code coverage tool for their simplicity, ease of use, and support for various features such as mocking, assertions, and running tests in parallel. In addition, Jest is maintained by Meta, with ample documentation and support available.

6.4 Libraries

- GMAT and Pyorbital: Since these technologies are already integrated within the project, we will need to reuse it to create our application.
- MaterialUI: A great front-end library that allows a wide variety of features and options for us to create a great UI. Will change and new libraries will be added as needed.

6.5 Tools

Gpredict, N2Y0. These two are orbiting tracking technologies that will help us locate the satellite in the orbit which is necessary to know for our project. Tools might change and be added as needed.

6.6 Continuous Integration (CI)

For Continuous Integration (CI), Github Actions offers a pipeline to automate workflows. In terms of this project, Github Actions will be used to set up integrated status checks which includes deployment checks. This will primarily be used to compare pull requests with the main branch, such that the code must pass a series of deployment checks before merging. To add, the project will also use Github Actions to manage secrets such as private API keys. This way sensitive information can easily be accessible in the code while maintaining confidentiality and integrity of the secrets.

7 Coding Standard

As mentioned earlier, we will be using ESLint within VS Code (our code editor of choice). ESLint has its own coding standard and it outlines the rules relate to possible logic errors in code and syntax related issues. More can be found in this link: `https://eslint.org/docs/latest/rules/`

We will be following certain practices to be as efficient as possible. Some of these include constantly backing up our saved progress/edits so it becomes instinctive. Additionally, we will try to comment as we build our application in order to avoid any confusion. Other practices include choosing appropriate and clear naming conventions for variables and functions, and aiming to write concise lines of code. We will also make use of indentations to make our code readable and constantly build our documentation.

8 Project Scheduling

Deliverable	**Deliverable Date**	**Meeting Managers**
Problem Statement, POC Plan, Development Plan	Sept 25	Buu Ha (Quinn)
Requirements Documentation Revision 0	Oct 4	Rishi Vaya
Hazard Analysis 0	Oct 20	Diamond Ahuja
V&V Plan Revision 0	Nov 3	Umang Rajkarnikar
POC Demonstration	Nov 13	Dhruv Cheemakurti
Design Document Revision 0	Jan 17	Buu Ha (Quinn)
Revision 0 Demonstration	Feb 5	Rishi Vaya
V&V Report Revision 0	March 6	Diamond Ahuja
Final Demonstration Revision 1	March 18	Umang Rajkarnikar
EXPO Demonstration	April 1	Dhruv Cheemakurti
Final Documentation	April 4	Buu Ha (Quinn)