Software Requirements Specification for : Mission Control Terminal MC

October 9, 2023

Contents

Revision History

Date	Developer(s)	Change
October 6, 2023	Q.H, R.V, D.A, D.C, U.R	Completed SRS documentation
October 9, 2023	U.R	Added missing relevant facts and assumptions

1 Introduction

1.1 Purpose of Document

The purpose of this document is to explicitly define ground knowledge for the project, including definitions, context around the system, requirements, and more. This document is a collaborative effort across stakeholders, and aims to provide a clear and unambiguous understanding of the project's purpose, functionality, and constraints.

1.2 Scope of Requirements

The scope of requirements is constraint to an application which operators can use to interact with various satellites, including, but not limited to, NEUDOSE, PRESET, and engineering test satellites. This project is not concerned with any other development that is currently undergoing at MIST.

1.3 Characteristics of Intended Reader

The intended reader is already familiar with the goals and objectives of the project, and might be reading this document for clarity around the project's purpose, functionality, and constraints. Should the reader not be familiar with the project, please refer to the Problem Statement and Goals document found here

2 Purpose of the Project

2.1 User Business

The primary objective of our project is to provide satellite operators with a platform to automatically schedule and send commands to satellites as they pass overhead. In addition, the system will keep logs of all commands sent, responses, with concurrent access available.

2.2 Goals of the Project

See Problem Statement and Goals document, section 2.

3 Stakeholders

3.1 Client

- Dr. Soohyun Byun
  - Role: Supervisor and Principal Investigators at MIST.
  - Interest: Dr. Byun, as the supervisor of this project, wants to ensure that the application facilitates efficient communication and success of the NEUDOSE and PRESET CubeSats.
- Austin Liu
  - Role: PRESET Systems Team Lead
  - Interest: Austin is responsible for the overall system integration and performance of the PRESET CubeSat.

3.2 Customer

- McMaster Interdisciplinary Satellite Team (MIST)
  - Role: Organization
  - Interest: As the organization, their primary focus is on the success of the Mission Control Terminal application as it affects the entire satellite team's operations. Their main concern is with the satellite operations and communication being reliable.
- Any other Lower earth orbiting satellite commands or controls team
  - Interest: Since the software will deal with satellite operation commands handling, any organization that sends lower earth satellites can be a customer of the software.

3.3 Other Stakeholders

- Muhammad Danyal
  - Role: Software Specialist on Mission Operations and Control Team

  - Interest: As a software specialist, Muhammad's primary focus is in the functionality and usability of the Mission Control Terminal application.

- Jay Patel
  - Role: Command and Data Handling Team Leader
  - Interest: Jay is interested in this project's success as it directly relates to the handling of commands and data for the CubeSat.

- Dr Eric Johnston and Dr Andrei Hanu
  - Role: Principal Investigators of MIST
  - Interest: As principal investigators of MIST, they will be one of the primary users of the data received from the satellite through MCT.

3.4 Hands-On Users of the Project

- Satellite Engineers
  - Role: Engineers responsible for the operation of PRESET CubeSat
  - Interest: Satellite engineers may be involved in the testing and integration of the application with the satellite systems.

- Command Operators
  - Role: Users who utilize the application for mission control activities
  - Interest: These are the end users of the Mission Control Terminal application. Their main focus is in the functionality, intuitiveness and responsiveness of the application.

3.5 Personas

- Application Administrators
  - Responsible for user management where they can add, edit and define roles for operators

  - Allowed to modify the permission list for each target and user.

- Operators
  - Users who are primarily scheduling commands to a target satellite. This includes creating, editing and deleting scheduled commands and sequences

3.6 Priorities Assigned to Users

Ordering of priorities based on the different user roles:

- Application Administrators: highest priority as they manage user and target access control.
- Operators: second and base role that a user can have. A user with this role is allowed to access the application and schedule commands to a target satellite.

3.7 User Participation

- Operators: Operators are users who will be benefiting from this application and they will be actively interacting with the software application to schedule commands and gain data from the satellite. As a result, they will have a high involvement when ensuring the functionality of the application.
- Administrators: administrators will also have an active involvement in the usage of this software application. They will benefit by being able to easily manage user accounts in addition to scheduling commands to a target satellite. Therefore, the level of participation that application administrators have in this application is high.

3.8 Maintenance Users and Service Technicians

- Muhammad Danyal and Mission operations and control team
  - Role: Software Specialist on Mission Operations and Control Team
  - Interest: As a software specialist, Muhammad's primary focus is in the functionality and usability of the Mission Control Terminal application.

4 Mandated Constraints

4.1 Solution Constraints

Solution Constraints are provided by MIST:

- The MCT shall facilitate user creation and authentication through the OpenID Connect protocol.
- The MCT command port shall use the UTF-8 character encoding and exchange commands and data in plain text.
- The MCT shall use TLE (two-line element set) data periodically obtained from the internet to predict satellite overpasses and solar illumination cycles.

5 Naming Conventions and Terminology

5.1 Glossary of All Terms, Including Acronyms, Used by Stakeholders involved in the Project

- MCT - Mission Control Terminal
- MIST - McMaster Interdisciplinary Satellite Team
- NEUDOSE - Neutron Dosimetry & Exploration (Satellite)
- PRESET - Pitch Resolving Spectroscopy for Electron Transport (Satellite)
- JSON - JavaScript Object Notation
- JWT - JSON Web Token
- Open standard to exchange security details over a network.
- GUI - Graphical User Interface
- CPU - Central Processing Unit
- RAM - Random Access Memory

- Disk space - The amount of storage that a computer hard disk/drive can store
- CI/CD - Continuous Integration/Continuous Development
- SSH - Secure Shell
- UTF-8 - Unicode Transform Format 8 bit
- Node.js - JavaScript runtime environment
- TypeScript - Extension of the JavaScript programming language which enforces strong typing
- Next.js - JavaScript framework for developing applications
- HIL Testing - Hardware in the Loop Testing
- Real-time simulation that allows for the testing of embedded systems
- Operator - Trained staff who are users of the application

6 Relevant Facts And Assumptions

6.1 Relevant Facts

- UTF-8 does not produce conversion or loss of information when text is encoded in ASCII.
- OpenID Connect's authentication protocol uses JSON Object Signing and Encryption (JOSE), including JSON Web Token (JWT) to encrypt information, which can be securely used for authorization and authentication flows.
- Github provides built-in support for Semantic Versioning.
- MongoDB offers auto-scaling capabilities, allowing for reactive vertical scaling.

6.2 Business Rules

- Only users with the application administrator role can access and modify user data, such as permissions and roles
- Only authenticated and authorized users can send commands through the MCT.
- Logs of command sent to the satellite are stored in a database. The web application must be accessible through a web browser.

6.3 Assumptions

- The application will be hosted on Linux-based servers.
- The application will not be hosted beyond a single server.
- The application does not store any science data collected by the satellite.
- Commands to operate the satellite will be provided to the application, where it does not need to design specific commands.
- The application will be compatible with commonly used web browsers, including Chrome, Firefox, and Edge.

7 The Scope of the Product

7.1 Product Boundary

Figure 1: Scope and Out of Scope

In Scope	**Out of Scope**
Command Scheduling: Operators need to be able to schedule commands for engineering and NEUDOSE satellites. They should be able to work concurrently with other users.	Satellite Hardware and Control Systems: The application will not directly interface with the satellite's hardware; it will send commands to the ground station for transmission.
Orbital Prediction: Operators need to be able to calculate and predict satellite overpasses and satellite illumination cycles.	Ground Station Infrastructure: The application will not manage or control the ground station equipment.
User Authentication and Authorization: Users must be able to log in, with a special admin user which controls access to satellite usage.	Logging: Users must be able to retrieve a log of commands sent, and responses from the application.

7.2 Product Use Case Table

Table 1: User Use Cases

User	Use Case
Operator	- Schedule a set of commands to NEUDOSE and engineering test satellites via GUI & Command Line - Modify & Delete a set of commands to NEUDOSE and engineering test satellites - Access historical log data on satellite, operations, and responses - Login/Logout - View current orbital state for satellites of interest, and future overpasses
Admin	- Grant access to Operators on different access levels - Edit access permissions per operator

8 Functional Requirements

8.1 Functional Requirements

1. The MCT must run as a web application consisting of a graphical UI accessible through a web browser and TCP port to communicate commands.
   Rationale: We believe a web application is the most appropriate way to achieve communication between users and the satellite. We chose Linux because of its flexibility, reliability and overall security.

2. The MCT must manage a list of users and its data, specifying the roles and all its associated actions.
   Rationale: This is to ensure that the system has a list of users to verify who's allowed to enter and who isn't.

3. The MCT will present a graphical interface allowing users to enter, select, and execute satellite commands while receiving text output from each command.
   Rationale: This is important because this is one of the most basic functions of our application, which includes three fundamental functions. The text output allows the user to understand the process as it's happening so they are in the know.

4. The MCT will log all commands sent to and received from each target with timestamps.
   Rationale: This is helpful when the system runs into errors, because it's easy to trace and determine what command caused it.

5. The MCT must present a graphical user interface to create, edit, and delete automated command sequences.
   Rationale: A graphical user interface enables intuitive interaction between the user and the system and improves the overall user experience.

6. The MCT must have an option to cancel an automated command sequence during execution.
   Rationale: This is to allow the user to undo if a command was sent as a mistake or incorrect on their part. This will save plenty of time for the user rather than having to wait till the original command has been processed.

7. The MCT shall store a list of allowed commands and automated sequences for each target, specified using regular expressions.
   Rationale: This will ensure the system can verify a command when inputted by a user so that it matches to an allowed list. Otherwise, any command might influence a target which can cause damage.

8. The MCT shall store a list of allowed commands and automated sequences for each user, specified using regular expressions.
   Rationale: This will ensure the system can verify the user so that it matches to an allowed list of users. Otherwise, any command by any user can negatively influence the satellite.

9. The application administrator shall be able to edit the permission list for each target and user.
   Rationale: An admin user who is responsible enough has been given

permission to edit the list of targets and users. Without this functionality, if a user or command is attempting to cause damage it cannot be reprimanded. The admin user can remove or delete them to avoid said damage.

10. The MCT will be able to reject a user from executing a command unless that command matches the permission list criteria for the originating user.
    Rationale: This is vital because a command has to fit the permission list criteria in order for it to be allowed to run. Without this constraint, any command would be eligible to run which would be out of scope for this project. Specifically, it has to fit the criteria for that user group as different groups have different permissions for commands.

11. The MCT will be able to reject a user from executing a command unless that command matches the permission list criteria for the command target.
    Rationale: This is important because a command has to fit the permission list criteria for it to run. Specifically, it has to fit the criteria of commands for that command target, because each target has commands specifically associated to it.

12. The MCT must be able to schedule commands and automated command sequences for future execution.
    Rationale: This is important because a user may not have the time to wait or schedule a command for later. Through this, a user can save time to run commands without having to wait.

13. The MCT shall present a graphical user interface to create, edit, and delete scheduled commands and sequences.
    Rationale: A graphical user interface presents an intuitive interface for the user to interact with the satellite through create, edit and delete functions. A visual UI is most intuitive for every human to interpret and work with.

14. The MCT shall present a graphical user interface to select and edit the satellites of interest.
    Rationale: A graphical user interface presents an intuitive interface for the user to interact with the select and edit the satellites of interest. A graphical UI is most appropriate solution for this function.

15. The MCT shall display the current orbital state for satellites of interest, elevation, and solar illumination.
    Rationale: This is very important because it displays what the user is working with. For each satellite of interest, its elevation, orbital state and its illumination are vital pieces of information for the user to know. Presenting these on the screen is appropriate.

16. The MCT shall be able to schedule commands when a satellite enters a user-specified elevation threshold over the ground station coordinates.
    Rationale: This allows the user to manipulate any satellite within the threshold. This enables the user to play around with more satellites as their threshold is subject to preference.

17. The MCT shall be able to schedule commands when a satellite enters or exits in the covered area.
    Rationale: This is a core function of the application. The application can only manipulate and schedule commands for satellites within this area. Beyond this area would be too difficult for the application to manage.

9 Look and Feel Requirements

9.1 Appearance Requirements

- **Look and feel:** The MCT look and feel should be consistent with the branding guidelines of MIST and the PRESET mission. This implies that the visual appearance and design aspects outlined by MIST and PRESET should be reflected in the user interface. The guidelines will be presented in section 5.3 of the Verification and Validation Plan document.
  - **Rationale:** The client has outline this specification.
  - **Phase in Plan**: This requirement will start phasing in at the time of designing the front-end of the application which starts after October 20, 2023.

9.2 Style Requirements

- N/A

10 Usability and Humanity Requirements

10.1 Ease of Use Requirements

- **Ease of use:** The MCT web interface must be easy to use and accessible for all users, regardless of their level of technical expertise or disability.
  - **Rationale:** It is essential to follow well-established usability and accessibility best practices, such as using clear and concise language, consistent design elements, and helpful feedback.
  - **Phase in Plan**: This requirement will start phasing in at the time of designing the front-end of the application which starts after October 20, 2023.

10.2 Personalization and Internationalization Requirements

- N/A

10.3 Learning Requirements

- **Ease of admin:** The MCT application should be easy to administer in order to reduce the workload on system administrators and make it easier to keep the system running smoothly.
  - **Rationale**: It is important to follow well-established administration best practices, such as using a centralized administration console, a simple and intuitive GUI, online support resources, and automation.
  - **Phase in Plan**: This requirement will start phasing in at the time of designing the front-end and back-end of the application which starts after October 20, 2023.

- **Documentation:** The MCT project should contain documentation for its web interface and key components. This documentation should be clear, concise, and easy to understand.
  - **Phase in Plan**: This requirement started phasing in at since the start of the project which was September 18th, 2023.
- **Training:** An average user should take at most 10 minutes to understand how to use the MCT application.
  - **Rationale:** It is a good practice to make an application less complicated and easier to learn by using appropriate headings, layouts, themes etc.
  - **Phase in Plan**: This requirement will start phasing in at when the application is ready for test deployment, which is February 5th, 2024.

10.4 Understandability and Politeness Requirements

- N/A

10.5 Accessibility Requirements

- **Compatibility:** The MCT web interface should be compatible with major browsers, operating systems and device types to ensure that all users, regardless of their preferred technology, can access and use the application effectively.
  - **Rationale:** It is important to follow well-established web development best practices, such as using standard web technologies, avoiding browser-specific features, and testing the interface on a variety of devices and platforms.
  - **Phase in Plan**: This requirement will start phasing in at the time of front-end development which starts after October 20th, 2023.

11 Performance Requirements

11.1 Speed and Latency Requirements

- **Responsiveness:** The MCT web interface should be responsive to user interactions. This is the measure of the time it takes for the interface to respond to user interactions.
  - **Phase in Plan**: This requirement will start phasing in during the Verification and Validation phase, which is on November 3rd, 2023.
- **Concurrent users:** The MCT should allow multiple users to access the service simultaneously to ensure that the latest information is displayed when a user hits refresh.
  - **Phase in Plan**: This requirement will start phasing in at the deployment of the test application which is at February 5th, 2024.

11.2 Safety-Critical Requirements

- **Availability:** The MCT web service should be available outside of scheduled maintenance and not be affected by the availability of any targets.
  - **Phase in Plan**: This requirement will start phasing in at when the application gets updates, which will be after its first test deployment on February 5th, 2024.

11.3 Precision or Accuracy Requirements

- **Time accuracy:** The MCT should ensure that the system date and time are accurate to the nanosecond as it is part of scientific research. The choice for the source will be presented in section 5.3 of the Verification and Validation Plan document.
  - **Phase in Plan**: This requirement will start phasing in at the time of Verification and Validation Plan on November 3rd, 2023.

- **Testing:** The MCT source code should have test suites to identify regressions and verify compliance with requirements. These test suites should support various features, including mocking, assertions, and parallel test execution.
  - **Phase in Plan**: This requirement will start phasing in at the time of Verification and Validation Plan on November 3rd, 2023.

11.4 Robustness or Fault-Tolerance Requirements

- **Exceptions:** The MCT application should handle internal exceptions to ensure that the errors are catched when they occur during runtime.
  - **Phase in Plan**: This requirement will start phasing in at the time of Verification and Validation Plan on November 3rd, 2023.

11.5 Capacity Requirements

- **Resource usage:** The MCT will minimize CPU, RAM, and disk space to be efficient.
  - **Rationale:** It is a good practice to use efficient algorithms and data structures, requiring the system to use fewer resources.
  - **Phase in Plan**: This requirement will start phasing in at the time of starting code which is after October 20th, 2023.

11.6 Scalability or Extensibility Requirements

- N/A

11.7 Longevity Requirements

- **Service lifetime:** The MCT service should be operational until 2026, subject to extensions and improvements.
  - **Rationale:** The client has outlined this specification.
  - **Phase in Plan**: This requirement will start phasing in at 2026 when the application will no longer be maintained.

12 Operational and Environmental Requirements

12.1 Expected Physical Environment

- N/A

12.2 Wider Environment Requirements

- **Staging environment:** A staging environment should be available to verify functionality before deploying to the production environment.
  - **Rationale:** It is important to verify that the application is functioning before deploying to the production environment as it can catch any errors that might occur at production stage.
  - **Phase in Plan**: This requirement will start phasing in at the first demonstration of the product, which is on February 5th, 2024.
- **Development environment:** A local development environment for the MCT should be self-contained, meaning that the developer does not need to install any additional software or tools on their local machine to develop and test the application.
  - **Phase in Plan**: This requirement will start phasing in at during the Verification and Validation Plan, on November 3rd, 2023.

12.3 Requirements for Interfacing with Adjacent Systems

- N/A

12.4 Productization Requirements

- N/A

12.5 Release Requirements

- **Commission date:** The MCT should be ready to use by September 2024.

- **Rationale:** The client has outlined this specification.
- **Phase in Plan**: This requirement will start phasing in at September 2024.

13 Maintainability and Support Requirements

13.1 Maintenance Requirements

- **Databases:** The MCT should use well-established database technologies to store application data and avoid text or plain files.
  - **Rationale:** A well-established database is reliable and efficient. It is also essential to ensure data integrity, performance and scalability.
  - **Phase in Plan**: This requirement started phasing in at the time of deciding the tools for the project, which was on September 25th, 2023.
- **CI/CD process** The MCT should use a CI/CD process for software deployment and incorporate automated testing and static analysis. This is necessary to improve the speed and reliability of software deployments, to reduce the risk of human error, and to ensure the quality of the software.
  - **Phase in Plan**: This requirement will start phasing in at during the Verification and Validation Plan, on November 3rd, 2023.
- **System updates** The MCT should be able to update its libraries and dependencies to improve security, performance, or functionality.
  - **Phase in Plan**: This requirement will start phasing in at during the Verification and Validation Plan, on November 3rd, 2023.
- **Code quality:** The MCT source code should be well-organized, modular, and follow fundamental principles for software engineering. The standards will be defined in section 5.3 of the Verification and Validation Plan document.

  - **Phase in Plan**: This requirement will start phasing in at the start of the code for the project, after October 20th, 2023.

- **Code formatting:** The MCT source code should use a code formatting tool to ensure that the code is consistent and easy to read.
  - **Rationale:** Since a project can have many contributors, it is important to be consistent with the style of code.
  - **Phase in Plan**: This requirement will start phasing in at the start of the code for the project, after October 20th, 2023.

- **Static analysis:** The MCT source code should use a linter or other static code analysis tools to catch errors and faults before runtime.
  - **Phase in Plan**: This requirement will start phasing in at during the Verification and Validation Plan, on November 3rd, 2023.

13.2 Supportability Requirements

- **Containerization:** The MCT should be a containerized application in order to be portable, scalable, and efficient.
  - **Rationale:** It is easier for a user to run tests on a containerized application as it does not require them to install any additional resources.
  - **Phase in Plan**: This requirement will start phasing in at during the Verification and Validation Plan, on November 3rd, 2023.

- **Backups:** The MCT should allow an administrator to back up its data to protect it from loss or corruption.

- **Phase in Plan**: This requirement will start phasing in at during the Verification and Validation Plan, on November 3rd, 2023.

- **Application Logs:** The MCT application should store logs to facilitate debugging.
  - **Rationale:** It is important to keep track of the logs in order to debug the application. This will also help in finding the methods/functions where an error occurs.

  - **Phase in Plan**: This requirement will start phasing in after the Verification and Validation Plan, on November 3rd, 2023.

13.3 Adaptability Requirements

- N/A

14 Security Requirements

14.1 Access Requirements

- **User security** The MCT should implement a multi-layered security protocol as it is necessary to protect the security of its user accounts.
  - **Phase in Plan**: This requirement will start phasing in after the system has completed every other core functionality, which will be a few days before February 5th, 2024.

14.2 Integrity Requirements

- N/A

14.3 Privacy Requirements

- **Service security** Network ports exposed by the MCT should be secured using a reputed cryptographic protocol.
  - **Phase in Plan**: This requirement will start phasing in at during the Verification and Validation Plan, on November 3rd, 2023.

14.4 Audit Requirements

- N/A

14.5 Immunity Requirements

- N/A

15 Cultural Requirements

15.1 Cultural Requirements

- N/A

16 Compliance Requirements

16.1 Legal Requirements

- N/A

16.2 Standards Compliance Requirements

- Source code and releases should be version controlled adhering to the Semantic Versioning 2.0 standard
- Commands and data shall be encoded using UTF-8 character encoding standard

17 Open Issues

N/A

18 Off-the-Shelf Solutions

18.1 Ready-Made Products

- OpenID Connect offers an authentication protocol that can be used to implement a user-based authentication flow for the application.
- SSH or TLS can be used as the cryptographic protocol to ensure security.
- ESLint is a linter that can be used to format and catch errors in the source code.

18.2 Reusable Components

- Material UI is an open-source React component library which offers prebuilt user-interface components that can be used out-of-the-box.

18.3 Products That Can Be Copied

- N/A

19 New Problems

19.1 Effects on the Current Environment

- N/A

19.2 Effects on the Installed Systems

- N/A

19.3 Potential User Problems

- Accessibility to the command terminal requires authentication to be granted through the OpenID Connect Protocol. However, this is an industry-standard protocol for authentication and authorization, making it highly unlikely to encounter this problem.

19.4 Limitations in the Anticipated Implementation Environment That May Inhibit the New Product

- N/A

19.5 Follow-Up Problems

- Currently the scope of this project is to be hosted on a single Linux server. However, extending this application to a generic MCT can cause data storage capacity issues when using the Free tier of MongoDB. This may require an upgrade in the tier of the database service.

20 Tasks

20.1 Project Planning

- October 10, 2023: Revise requirements document
- October 20: Hazard Analysis
- November 3: Verification and Validation Plan
- November 13 - November 24: Form a Proof of Concept (POC)
- January 17: Design Document Revision
- March 8: Create a user documentation for the project
- March 18 - March 29: Final Demonstration

20.2 Planning of the Development Phases

Phase 1: Project Initiation and Planning

- Divide into two teams: Frontend and Backend
- October 20: Hazard Analysis
- Research into technology stack
  - For frontend: Look into Next.js and TypeScript
  - For Backend: Look into node.js and MongoDB Integration
  - Authentication: Look into ConnectID protocol
  - Set up testing, development, and production environments

Phase 2: Development

- Frontend
  - Design mockups for the user interface and user experience (UI/UX)
  - Build the frontend architecture
  - Integrate APIs provided by the backend team

  - Integrate client-side authentication and authorization
  - Continuously test and debug frontend features
- Backend
  - Design database schema
  - Implement security measures and role-based authorization
  - Continuously test and debug backend endpoints

Phase 3: Integration Testing

- Perform testing to ensure the the functionality of the overall application is working as intended
- Identify and address issues

Phase 4: Prototype Launch

- Gather feedback from Revision 0 Demonstration
- Address any issues or bugs identified

Phase 5: Final Deployment and Launch

- Create user documentation and any required training materials
- Present the complete web-application

21 Migration to the New Product

The purpose of this product is to develop MCT (Mission Control Terminal), a new web application with the following features and goals:

- Enable scheduling commands using built-in orbit prediction
- Unify control of the Engineering and Flight satellite models
- Facilitate operator training and hardware in the loop (HIL) testing
- Ensure storage and logs of commands sent to the satellite
- Manage operator accounts and access control
- Improve ease of use and accessibility of mission control software

21.1 Requirements for Migration to the New Product

Some changes will be required for the migration to the new product. Specific details about requirements will be updated during the development process.

- Users will no longer need to access MCT from the machine at mission control. After the web application is live, users will have the freedom to access the MCT from any location.
- Strict role access will be implemented which will involve users to go through an authentication procedure to access MCT.
- The command communication process will migrate from the terminal to web based interface, making the application accessible to users of non-technical backgrounds.

Insert your content here.

21.2 Data That Has to be Modified or Translated for the New System

At this point, no data will require modification or translation

22 Costs

Provided below is a list of potential costs for the application. More details will be added as the project progresses

- Server Cost - Since the application is a website, it will require a host to serve the application.
- Storage Cost - Having an extensive log of data collected and stored multiple times a day requires a large amount of storage space.
- Authentication Costs - Robust authentication might require use of external applications.
- Testing Cost - Testing the application will involve several steps which will include moving heavy machinery (Transmitters, receivers and the ground satellite).

23 User Documentation and Training

23.1 User Documentation Requirements

The MCT project will contain documentation for its web interface and key components. This documentation will be clear, concise, and easy to understand.

23.2 Training Requirements

As a part of the final product, a training module will be added to the application. This training module will go through the components of the application and will allow a user to navigate through the system on a "test" satellite. Here, new users will have an opportunity to explore the system functionalities and get to test them, without affecting the 'real' satellites.

24 Waiting Room

At this stage in the development cycle, there are no more requirements that will be added to the final product. This part of the document may be revised and updated as required during the development process.

25 Ideas for Solution

At this stage in the development cycle, there are no more requirements that will be added to the final product. This part of the document may be revised and updated as required during the development process.

Appendix — Reflection

The information in this section will be used to evaluate the team members on the graduate attribute of Lifelong Learning. Please answer the following questions:

1. What knowledge and skills will the team collectively need to acquire to successfully complete this capstone project? Examples of possible knowledge to acquire include domain specific knowledge from the domain of your application, or software engineering knowledge, mechatronics knowledge or computer science knowledge. Skills may be related to technology, or writing, or presentation, or team management, etc. You should look to identify at least one item for each team member.

2. For each of the knowledge areas and skills identified in the previous question, what are at least two approaches to acquiring the knowledge or mastering the skill? Of the identified approaches, which will each team member pursue, and why did they make this choice?

Quinn's Reflection

1. Throughout this capstone project, will have to collectively learn various skills around project management, front-end/back-end development, devops, and more. Even with the collective years of experience of the team, building an entire system from scratch will prove to be a challenge. For me, I anticipate acquiring finer front-end development, UX and project management skills, due to my current interests in development, team management, and as they are skills needed for this project.

2. Two approaches of learning front-end development and UX skills are online courses, and lots of development inside and outside the capstone project. These approaches are suitable for learning hard skills, and fill in the gap of knowledge that may exist. In addition, learning project management skills will be done through online courses, and speaking to professional project managers, to learn from. Project management as a skill is very personal to the individual, and being able to pick up tips and tricks from various professionals in the industry will ensure growth and eventually mastery of the skill.

Diamond's Reflection

1. To ensure the success of this project, the team will have to follow all of the Software Development Life Cycle (SDLC) principles. This includes processes from gathering requirements, designing, coding and implementation to deploying and maintenance. These process require skills like documentation, front-end / back-end development, CI/CD and good teamwork. My current interest is in the back-end development of our project and acquiring more skills relating to it such as Node.js and database integration.

2. I am currently taking a web-development course in which there are tutorials on developing a strong back-end for your web applications. It also shows integration with the database, which in our project is MongoDB. Another way is to make small personal projects to apply and refine the skills I currently have. I will pursue the first approach since back-end technology is very vast and our project is also time constrained which means creating small projects could result not being able to contribute to my full capacity to our capstone project.

Dhruv's Reflection

1. To complete this capstone project, our team will have to be well acquainted with various technologies, software languages and other skills such as testing and deployment. Although I have some experience working with front and back end applications, creating a project with this level of depth and requirements is new to me. I would like to work on my front end skills through online resources such as courses or checking out other's projects on GitHub. With this exposure, it would allow me to expand on my own creativity but also learn some tips and tricks. My interest would be more front end related because I find the creativity and aesthetic part of creating a great user interface very fun and challenging.

2. I would like to learn more about front end and back end through Youtube, online courses and by trying mini projects throughout the term. I believe that by practicing and actually building is the most effective way to learn. Although there are time constraints, enough mini courses and tutorials online can provide enough knowledge for me to build this capstone project well.

Rishi's Reflection

1. The capstone project will require deep understanding of multiples technological and project management concepts. These concepts will be related to backend, frontend, databases and authentication on the software side, and Development cycles, continuous integration/continuous delivery (CI/CD) and documentation on the project management front. I would like to focus my development on the backend, databases and CI/CD. Although I have used these concepts during my internships and projects, I will need to acquire them in more detail for the successful completion of the capstone project.

2. There are a lot of resources available to gain a better understanding of the technologies mentioned. These include online courses, youtube videos, numereous articles and help guides. By learning these concepts through the resources and continuously applying them in projects, I will be able to develop these skills in a capacity that will enable me to fullfill the project's requirements.

Umang's Reflection

1. In order to make sure this project achieves all of its targeted goals, the team will have to become familiar when applying the software development principles, in conjunction with the technological and project management skills. The scope of this project involves real-life stakeholders which requires a level of organization and good communication to ensure certain goals are met. In addition, since this is also a software development process, developing a strong understanding of the technology stack will also help streamline the development process. As a group, we have to sharpen our concepts on the applications of the software development life cycle, as well as frontend, backend, and CI/CD technologies. I am aiming to build on my backend development skills, particularly Node.js and MongoDB's database service. While I do have some knowledge on these systems, I will have to be more fluent in these technologies in order to develop a successful product.

2. There are several resources, whether it is in the form of books or videos, to learn more about the technologies and processes mentioned above. I will learn by reading through the documentation of the technologies, as well as online articles and possibly courses.

Next steps:

- Backend development courses and documentation: Rishi, Diamond and Umang will take the initiative to gain a deeper understanding of the backend development technologies by looking through available courses and documentation provided by Node.js and MongoDB.
- Frontend development courses and documentation: Quinn and Dhruv will take the initiative to gain a deeper understanding of the frontend development technologies by looking through available courses and documentation provided by TypeScript and Next.js.
- CI/CD courses: Dhruv and Diamond will take the initiative to gain a deeper understanding of the CI/CD through available courses and guides.