Development Plan
SyncMaster

Team 15, SyncMaster
Kyle D'Souza
Mitchell Hynes
Richard Fan
Akshit Gulia
Rafeed Iqbal

Table 1: Revision History

Date	**Developer(s)**	**Change**
2024/09/19	Richard Fan	Conversion of tex to markdown
2024/09/24	Whole Team	Initial development plan document completed
2025/03/15	Mitchell Hynes	Convert from markdown into LaTeX
2025/04/03	Kyle D'Souza	Update performance goals

This document outlines the development plan for the project. Further information regarding the problem is located in ProblemStatement.pdf.

1 Confidential Information?

This project will not contain extensive confidential information warranting a confidentiality agreement. After a team meeting with MILO, it was determined our team will inform the City to advise us of any confidential information. We will ensure it is not included in the source code or at the capstone expo.

2 IP to Protect

The project does not have any IP to protect.

3 Copyright License

Our team will be licencing our project under the MIT license (https://github.com/Spitgranger/capstone/blob/main/LICENSE). As our project is done in collaboration with industry this license makes sense due to its flexibility. It will allow our industry partners to freely use and modify the code as they see fit in both open source and proprietary projects. This may be the case after the conclusion of this capstone project, hence the choice of this license.

4 Team Meeting Plan

Weekly team meetings will occur on Monday evenings. Meetings to be held online or in-person depending on the groups preference that week.

We will meet with our industry advisor bi-weekly, but this could be more or less depending on the needs of the capstone team as we progress through the deliverables. Meetings will be on Microsoft Teams, with the option to meet in-person if it is convenient for the particular meeting. Meetings will be structured with a prepared agenda by the capstone team and the discussion with the stakeholder chaired by the team liaison. Meeting minutes and action items will be recorded by the capstone team.

5 Team Communication Plan

Type	**Method**
Regular communications	Discord
Assignment of tasks	GitHub issues

Communications with professors, supervisor, the City, and other stakeholders.	Handled by Team Liaison Mitchell Hynes, in-person or over email.
Meetings	Discord, scheduled through outlook calendar.

6 Team Member Roles

Meeting Chair: This position will coordinate meetings, ensure an agenda is set, and that all parties are prepared for the meeting. Chair of meeting depends on the expert of the subject matter at hand.

Backend Developer: Richard is the lead developer on the backend for the project.

Cloud Developer: Kyle is the lead developer of the cloud infrastructure for the project.

Frontend Developer: Akshit is the lead developer of the frontend for the project.

Project Manager: Responsible for ensuring that all team members are up to date on work being performed and that action items for needed tasks are completed.

Subject Matter Expert and Team Liaison: Mitchell is the team liaison. They are responsible for maintaining communications between the McMaster instructional team and the City for the group. This role is responsible for assisting the rest of the team to understand the business logic.

7 Workflow Plan

Git Workflow

For this project, Git Feature Branch workflow will be utilized for feature development. This strategy will enable the development of new features in isolated branches, enhancing collaboration and reducing the risk of introducing bugs in the main branch. In general branches will follow the naming convention of /. Some examples are given below:

- feature/<name of feature>
- fix/<name of fix>
- docs/<name of doc changed>

Pull Requests

Pull Requests (PRs) will be created by the collaborators of the project to merge their respective feature branches into the main branch. Upon opening a PR, a short description of what the changes are should be provided. PRs should be linked to at least one issue in the repository to ensure traceability of work. This workflow will ensure that minimal conflicts are introduced and also will reduce the probability of introducing bugs into the main branch. To merge a PR, it will need to be approved by at least one of the collaborators.

Github Project, Github Issues, Github Milestones, and Labels/Tags

Github Projects along with Github Issues will be utilized for the management and tracking of tasks in this project. Github Projects will be used to track issues. Our project will classify the status of the issue as: "No Status" (the issue has no status assigned to it), "Backlog" (we need to refine the issue by for example, adding more details to it issue before it is ready to be picked up), "Ready" (issue is ready to be picked up for development), "In Progress" (issue is currently in development), "In Review" (issue is currently under code review), "Done" (issue has been completed). Additionally, issues will be assigned one of the three priority levels: low, medium, and high. The issues will be created with different custom predefined templates to standardize the process and save time containing the following details at a minimum:

- An outline of what needs to be done
- A soft deadline on when the work should be done
- A short description of what it means for this issue to be done/closable

Github Milestones will be used to keep track of the changes which need to be done for the current/next deliverable (e.g., "Dev Plan and Problem Statement" deliverable).

Issues will be classified using the following labels (the following is subject to change):

- Feature - For new development features
- Documentation - For adding or modifying existing documentation
- Bug - For fixing and tracking bugs
- Chore - Simple tasks such as removing comments
- Improvement - An improvement to the existing codebase (e.g. efficiency, complexity)

A minimum of one person will be assigned to each issue.

Additionally, custom issue templates will be used to streamline the process and to ensure all the necessary information is provided while creating the issue.

Tags will be utilized to mark any significant releases (for example, "poc-release").

CI/CD

For continuous integration (CI), we will be using Github Actions Workflows which will involve running linters, tests (like unit tests) on the project. Additionally, actions will be created to automate builds and to ensure that no breaking changes are introduced via pull requests. We currently plan to start out by introducing an action to perform linting and formatting, as well as an action to perform unit testing. These actions will run anytime a pull request is made against the main branch. In the future, we plan to set up actions to automatically update dependencies to mitigate security risks.

For continuous deployment (CD), we will create Github Actions for deploying the code into staging (testing) and production environments.

8 Project Decomposition and Scheduling

- We will be using Github Projects to decompose our milestones. Each Milestone will be divided into individual tasks which will then be assigned to different team members.
- Each document will be split into sections, and assigned individually or to groups as needed.
- The proof of concept, revision 0 and revision 1 will be split feature-wise and assigned to individuals or groups as necessary.
- Time will be alloted for verification, validation and review after the completion of each milestone and before the due date.
- Our GitHub Project

Milestone	Schedule
Requirements Document Revision 0 Due: October 9th	Start: September 24th End: October 5th
Hazard Analysis 0 Due: October 23rd	Start: October 9th End: October 19th
V&V Plan Revision 0 Due: November 1st	Start: October 16th End: October 28th
Proof of Concept Demonstration Due: November 11-22	Start: after completion of the Requirements document End: by November 4th, with improvements being made and testing being done until the day of our demonstration
Design Document Revision 0 Due: January 15th	Start: after completion of the Proof of Concept demo End: by January 11th
Revision 0 Demonstration Due: February 3-14	Start: alongside proof of concept (after finalizing requirements in the meeting with the City) End: by February 1st
V&V Report Revision 0 Due: March 7th	Start: alongside the development of Revision 0 of the project End: by March 4th
Final Demonstration (Revision 1) Due: March 29th	Start: after the completion of Revision 0 End: by March 20th
EXPO Demonstration Due: April 8th	Start: 2 weeks before the day of the expo End: April 1st
Final Documentation (Revision 1) Due: April 2nd	Start: March 10th End: by March 30th

9 Proof of Concept Demonstration Plan

One of the main goals for this project is contractor transparency and accountability when working on unsupervised city property. This leads us to one of the main risks of this project: ensuring that geolocation data provided by contractor devices is accurate enough for location verification. Our system plans on using contractor laptops to ensure that contractors are actually on site when they go through the process of reviewing and uploading documents. To do this, we have to ensure that the location reported by laptops is accurate up to 10m to accommodate for the smallest sites. However, since most laptops do not have GPS, the location may only be accurate to a few hundred metres, which could pose a

problem when it comes to verifying contractor presence on site. To mitigate this risk, we plan on tying a mobile device with GPS into the system. We will use the user's phone to provide us with more accurate location data. Our proof of concept will rely on demonstrating that a user's location can be accurately and reliably given to the system through a simple scan of a QR code. The POC will then be as follows: a simple web application running on a laptop that allows users to view content only after their location is verified by scanning a QR code on their phone.

10 Expected Technology

We will be using Git for version control, the repository for the project will be hosted on GitHub and we will be using GitHub projects to track issues and schedule tasks for the project. The programming languages that we intend to use for this project are Javascript for our frontend and Python for our backend. On the frontend, we will use the Next.js framework. We intend on using AWS to host our application, for this, we will use AWS Cloudformation templates to define our infrastructure.

Some of the linter tools we will be using are Ruff, Bandit, and Pylint for Python. We will use cfn-lint for linting our Cloudformation templates. For Javascript, we will be using Prettier and ESLint. In terms of testing frameworks, we will be using Jest for Javascript and Pytest for Python. For Python, a code coverage tool we intend on using is Pytest-cov.

For the CI we will be creating Github actions workflows for running linters, formatters and unit tests on the project. We will also create actions to automate builds to ensure PRs do not contain breaking changes, and create actions for deploying the application into production and testing environments.

Of course, the technology mentioned here is not final and is subject to change as the project progresses.

11 Coding Standard

To enhance uniformity across developers and adherence to best practices, the following coding standard will be adopted:

- **Linting:** As mentioned above, we will be using ESLint with @typescript/eslint plugin for TypeScript code and Pylint for Python code. The default ruleset for both tools will be used to enforce best practices pertaining to both syntax and semantics as identified by the community.

- **Code Style and Formatting:** Prettier will be the main code formatter used for this project. Prettier was chosen because of its opinionated

nature by default with regard to formatting, as well as compatibility and popularity with the TypeScript language. Code written in Python will use Ruff. Naming conventions of variable and function names will follow the style guides provided by the creators/maintainers of the aforementioned tools.

- **Code Documentation:** To enhance readability and maintainability, all code which does not perform a trivial task shall be documented. When documentation is needed, it will follow JSDoc or Pydoc conventions so that auto-generated documentation can be produced.

- **Testability:** Although subjective, the team should try their best to write code in a way that is testable. This includes following the principles of high cohesion, low coupling and single responsibility.

Appendix — Reflection

1. Why is it important to create a development plan prior to starting the project?

   It is well known that figuring out what to do is harder than actually doing it. We have experienced this issue firsthand, and are aware of the value that a well-written development plan provides. We found that through the creation of the development plan, ambiguities surrounding the project became clearer. For example, we were able to identify strengths and weaknesses in the team's skill set which allowed us to get started filling ourselves in on any knowledge gaps before starting the project. Additionally, creating this development plan allowed us to research potential technologies relevant to our project before starting, giving us time to think about alternatives and choose the best tools for the job. Finally, it encouraged us to start communication with our stakeholders and allowed us to develop some common ground about the project, especially when it came to expectations and goals. So in summary, creating a well structured and organized development plan allows everyone to stay on track and provides milestones to see if the project is progressing smoothly. It also gives an idea of what to expect for the project and helps align the team and stakeholders prior to starting a project.

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

   CI/CD has several advantages but also has a couple of major disadvantages that are often not talked about. CI/CD makes development at every stage cyclical and reduces time between development cycles and release. The features such as the ability to set up pipelines to do automated continuous testing, builds, and deployments with every incremental change allows for flexibility in terms of catching errors and fixing them. This leads to improved quality as bugs can be caught early and fixed, reducing technical debt and putting the focus on current goal. CI/CD also perfectly compliments modern development ideologies such as scrum/agile which prioritize regular review and multiple iterations. However CI/CD also has several disadvantages, most notably giving developers a false sense of security when it comes to the correctness and quality of their code. This leads to developer "laziness", an effect where developers are less likely to scrutinize and manually check what they have written. This is because the automated checks and builds are not perfect, but the illusion of them being perfect is there. Developers will often see that all checks have passed and assume that their code is correct, but this is false; checks passing do not necessitate correctness.

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

We did not have many disagreements about this deliverable. One small disagreement was over the tech stack. We were debating whether to use Firebase or using AWS. The arguments in favour of Firebase were that from research it looks like it is easier to set up and easier for a complete beginner to use. The arguments in favour of AWS are that there are a couple of team members who are more familiar with it and have used it before. We voted and decided to go with AWS because some team members had experience with it.

Appendix — Team Charter

Borrows from University of Portland Team Charter.

External Goals

The goal of the project is to produce a well developed piece of software per the requirements of the course which the City is satisfied with and could use in their workflow.

Attendance

Expectations

Team members are expected to attend all scheduled meetings as reasonably possible. If a team member needs to miss a meeting, they should inform the team beforehand. During the meeting, team members are expected to be on time and stay for the entire duration of the meeting unless prior arrangements have been made, something urgent comes up during the meeting (see Acceptable Excuse), or the meeting goes over the scheduled amount of time.

Acceptable Excuse

Acceptable excuses would include personal/family emergencies as well as conflicts and events beyond one's control such as a sudden power outage or a commitment that cannot be rescheduled. These excuses, however, should be kept to a minimum and each team member shall do their best to meet deadlines and attend meetings. Unacceptable excuses would include, but are not limited to:

- Poor time management: Claiming that there wasn't enough time or forgetting
- Personal issues that are not urgent: Being tired, or sleeping in
- Last Minute Non-Urgent Events: Missing a meeting or deadline to attend a party at the last minute

The examples above provide a guideline as to what is acceptable or unacceptable. Ultimately excuses will be considered by the team on an ad-hoc basis so that the best solution can be agreed upon given the circumstances.

In Case of Emergency

If a team member has an emergency, team members will inform the rest of the team at their earliest convenience to ensure the other team members are aware that their responsibilities in the project will be impacted.

In case of an emergency, a team member will:

1. Write a message in the project discord at their earliest convenience regarding what work will be missed, and during what time frame they may be unavailable

2. Email team liaison Mitchell regarding the emergency to leave a record which can be shared with the course coordinators if necessary

Accountability and Teamwork

Quality

Each team member is expected to come prepared with the agreed upon meeting prerequisites. This includes any action items, questions, or code reviews. Each team member to is to put forth their best effort in the course and ask for help from their teammates or the instructional team if needed to meet the course deliverables at a high quality of work.

Attitude

Our team will be using the following as our code of conduct:

- Respect: All team members are to be treated with respect and courtesy. Diverse contributions and perspectives of each team member are to be valued.

- Communication: All team members are expected to communicate openly and honestly with use of clear and concise language to ensure no misunderstandings arise.

- Commitment: All team members are expected to make an honest effort to attend all the meetings and complete all assigned tasks on time and to the best of their abilities.

- Integrity: All team members are to uphold academic integrity and avoid plagiarism. Team members are also expected to be candid about their progress and any difficulties currently faced by them.

If any conflicts arise, team members are to first identify the issue by describing the problem clearly along with any relevant information. After the problem has been identified, a meeting will be held with the parties involved for an open discussion concerning the problem at hand. Each person in the meeting is encouraged to provide their insight. If necessary, a neutral third party will be involved. Once a mutually agreed solution is found, a document containing the record of the resolution and any action items will be created to ensure that all parties involved understand the agreement. Finally, the situation will be monitored until the issue is fully resolved.

Stay on Track

To keep the team on track we will use GitHub projects to keep track of tasks we need to complete and their deadlines. We will ensure that team members contribute to the team by assigning tasks for deliverables equally amongst the team so everyone has an equal workload and knows what they need to do. When someone's performance is below expectations we will first try to figure out internally in the team if there is anything that can be done to improve their performance. If nothing can be found within the team, then we will contact our TA for advice. A punishment for performing below expectations might include assigning more work to the individual in the following deliverables. For team members who do well, we will reward them by giving them Timbits at the end of the course.

The performance of team members will be judged based on the following:

- Completion of all assigned issues before their deadlines
- ~~Capstone lecture attendance for software engineering lectures* >= 50%~~ Did not end up using lecture attendance as a performance metric
- Internal meeting attendance* >= 80%
- Stakeholder meeting attendance* >= 90%
- ~~For commits, we will have to see how often things are getting committed into the repository. Still, as a general rule, we can say that if a group member has 35% fewer commits than the average number of commits for all other team members and also 35% fewer additions than the average for all other team members, then they are likely not meeting expectations.~~ Commits ended up as a non-representative metric of performance as people had different sizes of their commits

*For attendance-based goals if someone has an exception for some meetings, which is covered by the acceptable excuses, then the meetings that they were excused for are not counted towards their attendance.

If team members do not meet their goals we will first internally discuss with the team member why they were unable to achieve the goals and what can be done to prevent it from happening going forward. If no resolution can be reached we will discuss the situation with our TA or the instructor. We will not implement punishments like "the team member needs to bring coffee to the next meeting" because this is not productive and generally does nothing in the way of solving the root issues. There won't be any particular incentives for reaching targets early other than not having to worry about reaching the targets.

Team Building

We have been studying in the same program for nearly 5 years now, and so there are many shared experiences which bring us together as a team. While it

may not be necessary to schedule additional activities for team building, it may prove beneficial for stress release and increasing team cohesion.

- Team Lunches: Between classes, days when everyone is on campus and has time free around noon.
- Ad-hoc team activities: Other fun activities can be scheduled according to the free time of the team members

Decision Making

To make decisions, the group member who's role is to direct that task will organize the rest of the group around their respective action items. All team members will be able to contribute their opinion, and in general decisions will require a consensus amongst the group. If an issue is not resolved through discussion, it will be brought to the attention of the instructional team for mediation and feedback.