Development Plan
Software Engineering

Team 11, technically functional
Matthew Huynh
Cieran Diebolt
Vaisnavi Shanthamoorthy
Maham Siddiqui
Eman Ashraf

Table 1: Revision History

Date	**Developer(s)**	**Change**
September 22nd, 2025	All	Added First Draft

1 Introduction

This document contains details pertaining to the proposed development plan by the team. Sections 1 - 3 contain legal and regulation information, sections 4 - 6 contain team dynamics related information and section 7-10 contain timeline and technology logistics. The team charter is also included within this document.

2 Confidential Information?

There is no confidential information involved in this project. Any information from supervisors or stakeholders is trade-wide information and contains no trade-secrets or other types of confidential information.

3 IP to Protect

Guaranteed intellectual property for the project consists solely of written content, namely the code and documentation. This written content is covered under our copyright agreement; more detail in the copyright license section below.

No additional creations requiring a trademark agreement or any further protection of intellectual property, such as logos or slogans, are foreseen at this point of the project.

4 Copyright License

The copyright license adopted by the team for this project is the MIT license. This license provides others the right to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the software under the necessity that the copyright notice and permissions are provided in all copies of the software.

The copyright license for the project can be found in the team repository.

5 Team Meeting Plan

The weekly meetings will be on Thursdays at 4:30 pm at any library at McMaster University. These meetings will be scheduled for an hour and the duration may be adjusted as required. Each member is required to have read the upcoming deliverable document and ready to address any comments or questions that other members can answer. Additional virtual meetings may be required when approaching a deliverable deadline. Industry supervisor meetings will be held virtually on a biweekly or as needed basis. All members will be encouraged to chair meetings and take minutes.

The proposed template for meeting agendas:

- *Date, time*
- *Attendees*
- *Wellness check*
- *Re-visit last deliverable submission: positives, negatives and improvements*
- *Discuss any potential inquiries that need to be addressed by supervisor or professor*
- ***If new deliverable:***
  Brief introduction to upcoming deliverable
  Decide on division of labour
  Assign tasks based on preference and availability
  Tasks that are TBD - assign at least 2 members to complete if they finish their assigned tasks early.
  ELSE:
  Brief discussion of in progress deliverables.
  Team member updates:
    Completed tasks
    Outstanding tasks
    Tasks that require assistance (if any)
- *Discuss any potential inquiries that need to be addressed by supervisor or professor*
- *Ensure that all team members are satisfied with their tasks and will check in on a regular basis*

6 Team Communication Plan

Our team communication plan will be as follows:

- Issues: GitHub/GitHub Projects
- Weekly Meetings: In-person on campus
- Meetings outside the weekly meetings: Teams
- Meeting minutes: GitHub
- Meetings with stakeholders: Zoom
- Project discussion (asynchronous): Teams

7 Team Member Roles

See Table 2: Member Roles for team assignments

Editor: This member will read through all deliverables throughout development and a final read-through before submission. They will be responsible for any text alterations to ensure conciseness and effective communication within documents/code.
Liaison: This member will be responsible for communicating with all external parties and relaying their comments back to the team.
Researcher: They will conduct initial research before the team commences a task and convey their findings to the rest of the team.
Task Manager: They will decide how to allocate the workload between the members.
Github Manager: They will ensure that all proposed rules pertaining to the use of Github are followed and that information provided on the platform is complete and precise.
Back-end developer: They will be developing and ensuring that the backend is functional and up to code standards.
Front-end developer: They will be responsible for development of a user friendly interface.
Comment Reviewer: This member will review code and verify the presence of and clarity of comments. Maham will be reviewing the final version.
Github Wiki Manager: This member is responsible for verifying the appropriateness of all content on the Github Wiki.

Table 2: Member Roles

Role	**Name**	**Possibility of Change**
Editor	All	Yes
Liaison	Matthew H	No
Researcher	All	Yes
Task Manager	Eman A	Yes
Front end developer	All	Yes
Back end developer	All	NA
Github Manager	Cieran D	Yes
Comment Reviewer	All	NA
Github Wiki Manager	All	NA

8 Workflow Plan

- **Git usage** :We will primarily be using GitHub for managing the project files. Each team member has their individual branches on which they will

engage with. The member should open a pull request after completion of their assigned task. If multiple members are collaborating on the same task, they will use either member's branch. Once completed, a pull request can be opened which will be reviewed by the other team members. Issues that focus on a single aspect of the project may be opened and will combine all relevant items. Sub-issues can be used if additional refinement is needed.

Overall Workflow Plan

1. Issue Creation: For each deliverable, an issue will be opened to divide the respective sections and keep track of assigned tasks. As mentioned above, sub-issues will be used if additional refinement or breakdown of work is needed.
2. Branching: Each team member will create a dedicated branch for themselves to work on their respective tasks. If multiple members collaborate on the same task, they will work within either member's branch to avoid merge conflicts.
3. Checklist: As each member is working through their respective sections, they will keep track of the checklist in our repository. Once a specific task is complete, the member will check off the associated item.
4. Commit Process: All changes will be regularly committed to the respective branches.
5. Pull Requests (PRs): Once any issues have been resolved, a pull request on GitHub will be created. Changes must be reviewed for any merge conflicts before merging to the main branch.
6. Team member review: Another team member will review the PR against the rubric and provide feedback through comments. The author of the section should respond or implement any feedback provided by the member who reviewed the submission.
7. Merging: Once the PR has been reviewed, approved, and confirmed that all of the rubric criteria has been met, it will successfully be merged into the main branch.

Issue Management: We will be using GitHub Issues to keep track of the following items: tasks, meetings, attendance, etc. GitHub Issues will be used as a project log (i.e. attendance, meeting minutes or code review) as well as a tracker (i.e. deliverables, bugs, enhancements).

There are template issues readily available from our professor, as shown below, and we will be using these throughout our capstone.

Issue Templates Available:

- Lectures: Used to track team details during lectures (i.e. important notes, attendance)

- Peer Review: Individuals outside of the team review one of our items
- Supervisor Meeting: Agenda template for meetings with supervisors/stakeholders
- TA Meeting: Agenda template for meetings with TA
- Team Meeting: Agenda template for weekly meetings with our team
- Blank Issue: Blank template to be used to create a new issue

Issue Labels Classification:

There are a set of labels in GitHub Issues that we can use when creating issues to organize them accordingly. Some of the labels we may use can be seen below with a description of when they may be used:

- Bug: Used for issues that indicate unexpected behaviour and require correction.
- Lecture: Used for lecture-related tasks/issues.
- TA Meeting: Used for issues that include meeting notes, feedback and actions from TA sessions.
- Team Meeting: Used for issues where team meeting discussions are noted.
- Supervisor Meeting: Used for issues related to the project supervisor, including follow-ups from discussions.
- Documentation/Deliverable: Used for issues corresponding to any work related sections of the deliverables.

- **Use of CI/CD**: We will be using GitHub Actions as our CI/CD platform to ensure that all of our documentation and project files are being built, formatted, and reviewed properly before they are merged into the main branch of the project. Workflows which are defined in the YAML files will essentially trigger the workflows to run automatically on pushes and pull requests. Each run compiles the LaTeX document, checks formatting and structure, and lastly, also compiles a PDF with the latest changes. To ensure consistency across all files, merges require both teammate approval and passing CI checks. This forces errors to be caught early and also reinforces us as a team to follow industry-standard practices. Implementing these practices throughout the entire project will aid in achieving a consistent, polished and coherent final product and documentation.

9 Project Decomposition and Scheduling

- **GitHub Projects**

  Our team will be using GitHub Projects to keep track of and manage our tasks and deliverables. The board will be following a Kanban style where

all issues will be presented by cards on the board and they will be moved across the board across various columns. The columns listed across the board will be as follows: Backlog → Ready → In Progress → In Review → Done. There will also be custom fields set that you will see across the board which will be listed as: Milestone (M1 to M6), Type (task, meeting, bug, doc), Priority (High/Medium/Low), Assignee/Team Member, Deadline. All in all, each card will be linked to specific issues, milestones as well as fields to ensure we stay on track with the overall Capstone Schedule.

The project schedule follows the timeline and deadlines listed in the course outline. The deliverables outlined in the course outline have been grouped into six milestones (M1 to M6) for organizational purposes. These milestones align with major phases of the course and we have grouped them together based on similar timelines and tasks. The milestones mentioned above represent the following deliverables, listed below:

- M1 (Weeks 3 to 4): Problem Statement, Proof of Concept (POC) Plan, Development Plan, Team Charter
- M2 (Weeks 5 to 6): Requirements Document (SRS) Revision 0, Hazard Analysis Revision 0, Design Document Revision -1
- M3 (Week 7): Verification & Validation (V&V) Plan Revision 0
- M4 (Weeks 8 to 9): Integrated Draft Document (Problem Statement, SRS, Hazard Analysis, V&V Plan), Peer Reviews (whole document), Update draft with feedback
- M5 (Weeks 10 to 12): Proof of Concept Demonstration (document + results), Design Document Revision 0, V&V Report Revision 0, Refinement of SRS, Hazard Analysis, and V&V Plan based on results and feedback
- M6 (End of Term): Final Demonstration - Revision 1, EXPO Demonstration, Final Documentation Revision 1 (including Problem Statement, Development Plan, POC Plan, Requirements Document, Hazard Analysis, Design Document, V&V Plan, V&V Report, Extra Documentation 1, Extra Documentation 2 and Source Code)

- **Link to project**:
  GitHub Project Link

10 Proof of Concept Demonstration Plan

The application will follow the steps below (high-level):

1. A user will record them performing a physio exercise (TBD on body part and exercise)

2. Determine if the exercise is “good” form or “bad” form based on requirements from the stakeholders
3. From the analysis, the application will provide guidance to improve the efficacy of the exercise

Some potential risks from this process flow are:

- Variations in how different users will record each video.
  - Having a guide overlay to indicate where they should film the exercise from.
- Inaccuracies with depth or monitoring a 3D motion on a 2D plane
  - Prompting the user to also perform the exercise from another angle and then analyzing two different recordings.
- Hardware limitations - as the application would be easiest to use on a phone due to the camera already being on there, it may have limitations when processing the footage.
  - Using a lightweight solution to process the recording to reduce lag.
- Accuracy in the footage may be different in a non-controlled environment and lead to inaccurate assessment of the exercise.
  - A demonstration can consist of seeing if we are able to obtain the same feedback in different light levels/clothing to see if it will affect our input data.

11 Expected Technology

It’s important to note the technologies listed in this section are expected to be used within this project but are not set as requirements or bounding limitations on the technology for the project. See Table 3: Expected Technologies.

Table 3: Expected Technology

Technology Role	Name	Explanation
Repository	GitHub	Used to store project code and documentation for combined contributions.
Project Management	GitHub Projects	Used to track timelines for deliverables, meetings, and specific issues for implementation.
Language	Python	Backend object-oriented programming language.
Linter	Pylint	Linter for consistent formatting and coding standards.
Platform	VSCode	Standard development platform for the team.
Platform	Android Studio	Platform for creating AndroidOS applications. Has integrated linter.
Library	OpenCV	Library for extracting data from video footage.
Plugin	Chaquopy	Plugin for integrating Python with Android Studio applications.
Unit Testing	Python Extension	Plugin for additional Python support including unit testing and code coverage.
Unit Testing	Pytest	Specific unit testing software with more options.

Continuous integration within the project will be used for simple checks like latex build success, compilation success, and simple test cases throughout the project to ensure to changes to the main branch break the project and cause unnecessary pain points for the team. Additionally, using continuous integration to enforce our coding standards makes sure our formatting and naming conventions stay consistent throughout the entire duration of the project.

12 Coding Standard

The coding standards used for this project will follow those made by the creators of the language. For example, if we proceed with using Python as the main development language for our project then PEP8 will be enforced as it was made by the language developers. Some standards made by developers lack clear cut rules in certain cases where developers didn't intend for the language's usage, but we will discuss as a team to negate any ambiguity in such cases. Linters will be used within our development platforms to enforce the selected

coding standards, this will keep a clean and consistent format to the code as well as clear naming conventions for well manageable variables.

Appendix — Reflection

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

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

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

Maham:
Having a plan eliminates a lot of uncertainty, whether it is timeline, team roles or conflict resolution methodologies. It allows the team to focus on what is important without being bogged down by frivolous details. Plus, it allows effective allocation of resources. For example, if one person is better at front end development, members who do not want to be or cannot be involved in it, do not have to start from scratch and master a new skill within a small time frame.

The main advantage of CI/CD is a smoother integration of code while reducing the amount of mistakes and code conflicts that would need to be fixed at the end of the project. However, there is one troublesome disadvantage - conflicting merge requests (this is in the context of using github) and the very time consuming endeavour of fixing them.

As for group disagreements, I do not think we had any. All members carefully listened to any input from their teammates and implemented any changes that were deemed to be beneficial. We were professional and respectful of each other's opinions and gave equal importance to everyone's feedback.

Matthew:
I think it is important to create a development plan prior to starting the project because it helps our team navigate conflict down the line. Not everyone works the same and so this development plan acts as a formal document to communicate with the team and find common ground for everyone to perform well on. Furthermore, it is a form of having an awkward conversation now compared to down the line when this work wasn't done and there is no basis to critique a member's performance.

An advantage of CI/CD is the reduced manual effort to test and evaluate code. If there is a problem with one of our documents that we upload or a file that is incorrect, we can have the tests evaluate it and be flagged for issues with it. A downside of CI/CD is the overhead or the amount of time that it takes to set up. For a shorter project it will take a bit longer to set up and thus less time is put towards coding the features at the cost of automating the building and testing.

The group did not have any disagreements during this deliverable. Everyone was able to communicate to each other and each meeting went with a plan and left with actionable items to complete. As the term gets busier and there are more deliverables for other classes I think that the team charter below will help

guide us for the rest of the project.

Vaisnavi:
Creating a development plan before starting the project is extremely important as it allows us to map out all of our ideas and thoughts. As someone who is a visual learner, I have to see all the tasks written out before I start anything. Writing out a plan has always been something that aided me in successfully completing any task. Ultimately having a plan helps as it allows you to take a step back and see the entire picture and be able to map out every little thing in order to reach the final product.

One of the main advantages of CI/CD is the fact that it is able to provide us with quick feedback. It directs our attention to a merge conflict from a PR right away, which allows us to eliminate that error immediately without it breaking our main branch. On the other hand, one of the main disadvantages is that although it does catch errors, sometimes there could be false positives in terms of if there is a simple formatting issue but the actual syntax and document is perfectly fine. This ultimately will harm us in terms of efficiency as it will hinder our progress if we are trying to find a bug that does not exist.

No disagreements took place within this deliverable, everyone on the team was able to voice their concerns. All team members have been respectful and have been taking in the feedback and applying it. With that being said, we have had some suggestions come into play, specifically with our respective branches. We started off by having branches for each document we were working on, but we decided that it would be easier to keep track of each member having just one branch for their version of the document and from their multiple PR's could be reviewed and approved. We discussed this during a team meeting and quickly came to this resolution, which showcases how key of a role communication plays in a team setting.

Eman

Before starting a project, it is essential to create a development plan because it gives the team a clear path forward. In my personal experience, it helps in assigning duties, defining the scope, outlining deliverables, and setting reasonable deadlines. Brainstorming a plan allows us to save wasted effort and lower risks by recognizing problems in the early stages of our project by narrowing down our scope. Additionally, it provides a way to make sure that everyone is on board with the project's objectives.

Faster development, early problem detection, and consistent code quality are some of the primary benefits of CI/CD. In this method, testing and deployment save time and push the release of updates faster. Although it is relatively time. A downside to CI/CD would be that frequent releases can be problematic if not handled properly, but generally speaking, the advantages exceed the disadvantages.

The team members were respectful and professional in our first few conversations and meetings together. Everyone has had the chance to equally voice their opinion or thoughts and be taken into consideration. That being said,

there were a few differences in preference for the focus of the project. While some members were leaning toward one project idea, others including myself were striving to choose another project idea. While both project ideas are meaningful and have potential, the group decided to put it to a vote. While my choice of project wasn't favored by the majority, I was willing to take on my team member's choice of project and everyone was very professional about it.

Cieran:
The development plan sets the expectation for not only team members in their roles and responsibilities but also for the direction of the project as a whole. Creating the development plan prior to starting the project gives everyone a good idea, hopefully, of how to approach any foreseeable situation. I think it also makes everyone think of the project in advance and gets some of the team 'storming' out of the way early on.

The advantages of using CI/CD are excellent in my opinion; continually automatic test cases, compilation checks, and more are extra assurance for the team that the project is still moving forward. It makes it easier for developers on the team to contribute quickly and get assurance they haven't dropped the ball on the way. The downsides are the setup and the upkeep; setting up the CI/CD isn't hard but making sure it's testing the right things takes more in-depth knowledge of the project. Same with upkeep, if you add a new method, how do you apply CI/CD to that now? It becomes trickier, but I do believe the pros outweigh the cons.

There was some disagreement on which project to select initially and a lack of pushback on the team charter. We have group members that are passionate, and that's great, but it felt bad to not be able to realize each person's dream of their capstone project. We settled the former issue with a vote after exploring both options in further detail, and all group members were professional about the outcome. The latter issue is something I fear; ambiguity or oversight here could make future conflicts a little more awkward.

Appendix — Team Charter

External Goals

Our group's primary goal is to create a project that we are proud to have our names on, can be talked about in interviews and to peers. It is important for us to build a solution to our problem statement while also adhering to code quality such as readability and documentation. Through this, we seek to develop our technical skills through tackling a real world problem. Additionally, we are also aiming for above-average grades as a secondary goal.

Attendance

Expectations

Our team expects full attendance to meetings scheduled in advance for the entire duration of the meeting. If a member cannot attend a meeting or is going to miss a portion of the meeting, they must communicate it in advance to the team and catch-up on the missed portion of the meeting (i.e. meeting minutes). Missing 3 meetings in a row when a deliverable is not the main point of discussion will trigger a meeting with the rest of the team. Moreover, if a deliverable is the main topic, 2 missed meetings will be the limit. These standards are for non-emergency situations. For emergencies, see below.

Acceptable Excuse

Acceptable excuses include: Academic sessions (classes, tutorials), academic requirements (due-dates prior to the capstone deliverable) and medical appointments. An advance notice for medical appointments will be appreciated. It is the responsibility of the missing member to make up for a missed meeting. Repeated excuses will not be acceptable; please refer to attendance expectations. Please see below for emergency cases.

In Case of Emergency

In the event of an emergency case, the team member should contact the team. Extended cases of emergency (more than 2 weeks or time taken off that impedes deliverables) should be brought to the attention of the teaching staff to ensure the work load is manageable for the rest of the team.

Accountability and Teamwork

Quality

In order to maintain consistent quality for deliverables, each weekly meeting will briefly go over deliverable expectations. Each deliverable should be checked against the rubric [on avenue] and also against the repository checklist by the individual that wrote that deliverable but also by a teammate.

Furthermore, for meetings, team members are expected to arrive, review relevant material and provide any problems or status updates with their work to ensure the meeting stays on-topic and productive.

Lastly, during weekly meetings we will discuss "what went well, what to change" to see if there is a way to streamline communication to attain this quality. We also encourage each team member to ask for help in our informal chat in a timely manner while being respectful of each person's time.

Attitude

For this project, the expected attitude from group members is to stay positive, respectful, and collaborative. Everyone should contribute their fair share, communicate openly, and be willing to listen to each other's ideas. Mistakes and challenges will come up, but approaching them with patience and a problem-solving mindset will keep the group moving forward smoothly.

Stay on Track

To stay on track, we plan to use a project management tool like Jira to assign tickets and organize tasks for each group member. This will help us clearly see who is responsible for what, monitor progress in real time, and make sure deadlines are being met. By breaking down the work into smaller tasks and tracking them, we can stay organized, avoid confusion, and keep the project moving steadily toward completion. We can also measure each task and its success by breaking it down into smaller tasks. That way we can measure whether a group member has completed a task, or only 75 percent completed a task.

Team Building

The team will engage in social activities together if schedules allow. Team building activities will strongly be encouraged. There may be sessions of working together in person and spending some time together with any member that is available. This will also depend on work habits and preferences.

Decision Making

When making a decision on the team, we will lean towards a unanimous vote. We want each member of the team to feel their feedback is heard to improve the project communication. In cases where this is not the case, we will hold a vote. Since there are 5 members, no one can abstain that way a solution can be achieved.

Each member is allowed to voice their opinion to the team without interruption so everyone is heard and we can reach an objective solution (without personal feelings swaying the decision).

In the event that voting cannot be held properly, we will select a neutral mediator to guide us further.