Development Plan
Software Engineering

Team #5, Money Making Mauraders
Zhenia Sigayev
Justin Ho
Thomas Wang
Michael Shi
Johnny Qu

Table 1: Revision History

Date	**Developer(s)**	**Change**
09-21	Johnny Qu	Added section 8
Date2	Name(s)	Description of changes
...	...	...

This document outlines the development plan for the MES Club Payment Tracking System, a web application designed to streamline the reimbursement process for McMaster Engineering Society (MES) clubs. The current process using Google Forms and spreadsheets is inefficient, error-prone, and difficult to search. This plan provides the roadmap for design and implementation, describes team and workflow organization, lists the expected technologies, and identifies major risks and mitigation strategies.

[Additional information on the development plan can be found in the lecture slides. —SS]

1 Confidential Information?

This project does not contain industry confidential information.

2 IP to Protect

The McMaster Engineering Society (MES) will be the primary owner of the system and its deployed instance. The student development team will be credited as contributors. Unless MES requests otherwise, the project will be published under an open-source-friendly license allowing MES and future student teams to maintain and extend the system. If MES requests proprietary handling, we will follow the agreement provided by MES and notify course staff.

3 Copyright License

Given that the MES's website source code is open-source, we will be using the MIT License.

4 Team Meeting Plan

- Weekly team meetings: 60 minutes, fixed weekday time (set by team availability) for planning and progress updates.
- Bi-weekly stakeholder check-ins: 30 minutes with MES representative(s) to gather requirements and demo progress.
- Meeting format: rotating chair; agenda created as a GitHub issue before each meeting; minutes recorded in the issue and a short summary added to the project wiki.
- Ad-hoc technical sessions as needed for design decisions or integration tasks.
- Meeting Notes: Will be documented in GitHub Issues linked to the relevant sprint or task.

5 Team Communication Plan

- GitHub Issues for tasks, bugs, feature requests and meeting agendas.
- Pull requests for all code changes with at least one reviewer before merging.
- Discord for quick coordination.
- Email for formal stakeholder communications.
- Weekly status updates posted to a shared project board or README for the MES contact.

6 Team Member Roles

- Project Coordinator/lead: Oversees the overall project direction and ensures milestones are met in a timely fashion. Facilitates communication between faculty advisors and the team, keeps team aligned with project goals and deadlines
- Meeting Chair (rotates weekly): Prepares agenda and leads team meetings. Ensures discussion stays focused and all voices are heard. Summarize action items at end of all meetings.
- Notetaker (rotates weekly): Record meeting minutes, decisions, and assigned tasks. maintains organized project documnetation. Ensures deliverables are well documented, both for technical and non-technical audiences.
- Technical Development (everyone): Ensures coding practices, frameworks, and tools are consistent across team. SUpports integration of all coding components.
- QA/Testing (everyone): Review code correctness, readability, and adherence to standards. Develops and run test cases to validate functionality
- User Experience and Requirements (everyone): Gathers and refines requirements from MES stakeholders. Focuses on usability, interface design, and accessibility.

7 Workflow Plan

- Git branching: single protected `main` branch for production; feature branches named `feature/short-description`; pull requests for merges.
- Pull request policy: descriptive PRs, link to issues, at least one approving review and passing CI.

- Issue management: issue templates for bug, feature, and documentation; labels for priority and component (frontend/backend/ops).
- CI/CD: automatic test runs on PRs; deployments to a staging environment on merges to `develop` (optional) and production on `main` after review.
- Code reviews: enforce linting and basic tests; use GitHub Actions for automated checks (linters, tests, security scanning).

8 Project Decomposition and Scheduling

- We will use GitHub Issues and GitHub Projects to manage work and track progress. Issues will represent tasks, bugs, and feature requests; Projects will be used for sprint planning and high-level roadmaps.
- GitHub Projects board (create and link when available): Project board placeholder
- Capstone Repository: https://github.com/zheniasigayev/MES-Finance-Tracking
- MES Website Repository: https://github.com/McMaster-Engineering-Society/MES-Website-App-Router

The following are the key dates and deadlines for completion of different milestones for this project:

- **09-22** – Project plan, development plan, proof of concept plan, team charter
- **10-06** – Software Requirements Specification + Hazard Analysis
- **10-27** – Validation and Verification Plan
- **11-10** – Design Doc Revision -1
- **11-17** – Proof of concept demos start
- **11-28** – Proof of concept demos end
- **01-19** – Design Documentation due
- **02-01** – Revision 0 presentation starts
- **02-13** – Revision 0 presentations end
- **03-09** – Verification and Validation Report due
- **04-06** – Final Documentation Due

This allows us to break down the development of the MES Finance Tracker into the following stages.

Phase 1 – Scoping (September 22 to October 27)

During this time, we will work to refine our high-level goals into a comprehensible list of requirements. We will be conducting interviews as well as experimentations with existing processes to determine what we can and should address within our project. At the end of this phase, we should have a concrete plan of how we will approach the rest of the term by producing a requirements specification, hazard assessment, and a validation and verification plan.

Phase 2 – Prototyping (October 27 to November 17)

In this phase, we will start initial development on our application, focusing on the features mentioned in our proof of concept plan. The goal of this phase is for our team to ensure our core features are reasonable and achievable by us, and to gather feedback regarding our features.

Phase 3 – MVP (November 17 to February 1)

During this phase, the bulk of the development work needs to be done. The prototype developed in the previous phase should be improved to become a holistic user experience and all the requirements outlined in the SRS should be addressed. During the demo, we should be aiming to collect valuable feedback to improve the project.

Phase 4 – Finalizing (February 1 to April 6)

The goal of this phase is to polish what was built in the previous phase by iterating on feedback. However, we should also be aiming to optimize towards goals outlined in previous phases and work on stretch goals.

9 Proof of Concept Demonstration Plan

Two main high-risk functionalities have been identified regarding the success of our project that, if not completed, severely hinder achievement of project goals:

1. **Financial data extraction from photos and screenshots:**

   Importance Although manual data input will be supported, screenshots and images of financial data provides a much more efficient manner of data entry. As streamlining the financials managing process is one of the project's main goals, this is a functionality of high importance.

   Difficulty of Addressal The proposed input space includes physical photos: receipts, printed statements, as well as digital formats: screenshots of statements, account balances , etc. Although the narrow context (financial data in alphanumerical format) reduce the difficulty of addressing the risk, the wide medium of formats may prove a technical challenge.

2. **Integration with McMaster Engineering Society (MSE) Monorepo:**

   Importance As the MSE is the client body, integration with their existing systems to ensure long-term supportability is critical.

   Difficulty of Addressal As of the planning stage of the project, the unknown nature of the monorepo, presents potential difficulty. Although many group members already have experience integrating with company and other third party repositories, integration still requires thorough communication with MSE representatives and team supervisors.

A series of minimum Proof of Concept (PoC) functionalities have been proposed to address the risks outlined above:

1. **Text Extraction from Receipt Photos:** Physical print representations of financial data are more noisy than digital screenshots, and are more difficult for both traditional computer vision and machine-learning based computer vision techniques to extract data from. Receipts represent one of the potentially more difficult mediums, and is chosen as a representative case for the PoC.

2. **Basic WebApp hosted within MSE Monorepo:** As part of the project, the MSE is providing the group with rights and access to two (2) existing codebases to expand on. To reduce the magnitude of risk involved with integrating into the MSE's repository, integration will be prioritized as part of the PoC to minimize conflicts and sunken-cost efforts.

10 Expected Technology

[What programming language or languages do you expect to use? What external libraries? What frameworks? What technologies. Are there major components of the implementation that you expect you will implement, despite the existence of libraries that provide the required functionality. For projects with machine learning, will you use pre-trained models, or be training your own model? —SS]

[The implementation decisions can, and likely will, change over the course of the project. The initial documentation should be written in an abstract way; it should be agnostic of the implementation choices, unless the implementation choices are project constraints. However, recording our initial thoughts on implementation helps understand the challenge level and feasibility of a project. It may also help with early identification of areas where project members will need to augment their training. —SS]

Topics to discuss include the following:

- **Specific programming language:** JavaScript with Node.js; MongoDB
- **Specific libraries:** Not exactly libraries, but extensions in use: Error Lens, Tailwind CSS Intellisense, Babel JavaScript, Auto Close Tag, Auto Rename Tag, Better Comments

- **Pre-trained models**
- **Specific linter tools:** ESLint, Prettier
- **Specific unit testing framework**
- **Investigation of code coverage measuring tools**
- **Specific plans for Continuous Integration (CI):** GitHub Actions (or explain if CI is not being done)
- **Specific performance measuring tools (e.g., Valgrind), if appropriate**
- **Tools you will likely be using**

[git, GitHub and GitHub projects should be part of your technology. —SS]

11 Coding Standard

[What coding standard will you adopt? —SS]

Appendix — Reflection

[Not required for CAS 741 —SS]

The purpose of reflection questions is to give you a chance to assess your own learning and that of your group as a whole, and to find ways to improve in the future. Reflection is an important part of the learning process. Reflection is also an essential component of a successful software development process.

Reflections are most interesting and useful when they're honest, even if the stories they tell are imperfect. You will be marked based on your depth of thought and analysis, and not based on the content of the reflections themselves. Thus, for full marks we encourage you to answer openly and honestly and to avoid simply writing "what you think the evaluator wants to hear."

Please answer the following questions. Some questions can be answered on the team level, but where appropriate, each team member should write their own response:

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?

Appendix — Team Charter

[borrows from University of Portland Team Charter —SS]

External Goals

[What are your team's external goals for this project? These are not the goals related to the functionality or quality fo the project. These are the goals on what the team wishes to achieve with the project. Potential goals are to win a prize at the Capstone EXPO, or to have something to talk about in interviews, or to get an A+, etc. —SS]

Attendance

Expectations

Team members are expected to:

- Attend scheduled meetings and be on time.
- Notify the Project Manager and meeting chair in advance if they cannot attend.
- Act professionally and responsibly when absent (review minutes and follow up on assigned actions).

Preferred notification is the day before; same-day notification is acceptable for emergencies.

Acceptable Excuse

Any one of the following reasons that McMaster University recognises, will also be accepted. Below is a list of McMaster's acceptable extenuating circumstances:

- **Medical issues:** a physical or mental health condition that negatively affected participation or performance. Provide a physician's note or equivalent documentation when available.
- **Family crises:** significant personal or family situations (e.g., bereavement, serious illness) that create hardship.
- **Other personal emergencies:** unforeseen and significant events that disrupt academic or project commitments.
- **Religious observances:** conflicts with religious obligations where accommodation is required.
- **Varsity reasons:** official university sports commitments.

- **Business commitments (part-time students):** job-related duties that prevent attendance.

For urgent or sensitive circumstances where documentation cannot be produced immediately, notify the Project Manager promptly and follow up with documentation as soon as reasonably possible. Trivial reasons (e.g., social events) are not considered acceptable without prior agreement.

In Case of Emergency

If a team member has an emergency and cannot attend a meeting or complete agreed work, follow these courteous steps:

1. Notify the Project Manager and meeting chair as soon as possible (direct message and a short message in the team chat). If unavailable, notify any team lead.

2. Create a short GitHub Issue titled "emergency: ¡YourName¿" describing current status, outstanding tasks, and the expected return time (if known). Link related issues or PRs.

3. Where appropriate, assign a backup or delegate specific issues/tasks to another team member and update issue ownership. Use the handover checklist in the issue description.

4. Share any quick access notes or pointers needed to continue work (location of credentials, relevant commands, how to run tests). Do not post sensitive credentials in chat — use the team's secure credential store or inform the Project Manager privately.

5. If the absence affects an upcoming demo or deadline, request an immediate short meeting or asynchronous decision from the Project Manager about scope adjustments or re-assignment.

6. When able, provide a brief follow-up update and, if requested, supporting documentation. On return, do a short handover with whoever covered your work.

These steps prioritise clear communication and minimise disruption while respecting the privacy of the person experiencing the emergency.

Accountability and Teamwork

Quality

[What are your team's expectations regarding the quality of team members' preparation for team meetings and the quality of the deliverables that members bring to the team? —SS]

Attitude

[What are your team's expectations regarding team members' ideas, interactions with the team, cooperation, attitudes, and anything else regarding team member contributions? Do you want to introduce a code of conduct? Do you want a conflict resolution plan? Can adopt existing codes of conduct. —SS]

Stay on Track

[What methods will be used to keep the team on track? How will your team ensure that members contribute as expected to the team and that the team performs as expected? How will your team reward members who do well and manage members whose performance is below expectations? What are the consequences for someone not contributing their fair share? —SS]

[You may wish to use the project management metrics collected for the TA and instructor for this. —SS]

[You can set target metrics for attendance, commits, etc. What are the consequences if someone doesn't hit their targets? Do they need to bring the coffee to the next team meeting? Does the team need to make an appointment with their TA, or the instructor? Are there incentives for reaching targets early? —SS]

Team Building

[How will you build team cohesion (fun time, group rituals, etc.)? —SS]

Decision Making

[How will you make decisions in your group? Consensus? Vote? How will you handle disagreements? —SS]