Software Requirements Specification for McMaster Engineering Society Custom Financial Expense Reporting Platform: (Volere)

Team #12, Reimbursement Rangers
Adam Podolak
Evan Sturmey
Christian Petricca
Austin Bennett
Jacob Kish

January 6, 2025

Contents

Revision History

Table 1: Revision History

Date	Developer(s)	Commit	Change
10/11/24	All Team Members	Pull Request 21	All sections, including reflections
1/6/25	Adam Podolak	1d8c0cc	Making the constraints list bulleted

1 Purpose of the Project

1.1 User Business

The project aims to streamline the process of handling reimbursement requests for the McMaster Engineering Society, facilitating proper budget tracking, improved efficiency and user experience, as well as reducing human error.

1.2 Goals of the Project

Goals include:

- User-friendly reimbursement submission interface.
- Tracking of reimbursement requests.
- Tiered access to the reimbursement platform for different users.
- Invoice generation.

2 Stakeholders

2.1 Client

This software is designed for the McMaster Engineering Society and its staff, and by extension for McMaster University.

2.2 Customer

MES clubs/team members that are required to submit reimbursement requests for their projects and club work.

2.3 Other Stakeholders

University administration, future developers of the finance platform, and outside auditors.

2.4 Hands-On Users of the Project

Day to day, the software will be used by students submitting reimbursement requests for themselves or on their teams' behalf, as well as MES finance staff who are to review requests and handle them accordingly.

2.5 Personas

- **Club Treasurers**: Needs to be able to submit reimbursement requests easily and frequently, as well as track the status of requests.
- **Finance Staff**: Needs to be able to effectively manage all incoming reimbursement requests from different clubs/users.
- **Club Member/General Students**: Infrequent reimbursement request submission; user-friendliness is required.
- **Auditor**: Needs ready access to necessary financial documentation submitted to and generated by the finance platform.

2.6 Priorities Assigned to Users

Highest to Lowest:

1. MES Finance Staff
2. Club Treasurers
3. Club Members/General Students
4. Auditors

2.7 User Participation

User testing will be conducted with Club Treasurers/Club Members as well as the MES Finance staff to ensure requirements are met.

2.8 Maintenance Users and Service Technicians

MES staff will maintain the system after its deployment.

3 Mandated Constraints

3.1 Solution Constraints

- The finance platform will adhere to the data privacy regulations of McMaster University.
- The finance platform will integrate with existing McMaster IT infrastructure.
- The finance platform will only be available to McMaster affiliated individuals, including students, team/club members, treasurers, MES finance staff, McMaster Administration, and outside auditors.

3.2 Implementation Environment of the Current System

- The current system operates using a combination of Google Forms submissions and Google Sheets for tracking.
- MongoDB is used for database management, and Digital Ocean is used for cloud management.

3.3 Partner or Collaborative Applications

- While the finance platform will involve a technology stack, it will be standalone in its operations.
- It will integrate with existing MES IT infrastructure but will wholly replace the existing reimbursement platform.

3.4 Off-the-Shelf Software

- Some existing off-the-shelf solutions include Intuit QuickBooks and Xero.
- QuickBooks could be used for ledger tracking and invoice generation, while Xero can also streamline the invoice process.
- It is unclear at this stage which off-the-shelf software will be employed, if any.

3.5 Anticipated Workplace Environment

- The finance platform will primarily be accessed online via a website. Mobile access will also be possible.
- Although access will be restricted to McMaster affiliated individuals, the platform will be accessible from anywhere.

3.6 Schedule Constraints

- The finance platform must be completed by the end of the Winter 2025 term; however, completing it sooner is ideal to address existing MES finance concerns.

3.7 Budget Constraints

- The project at this stage is not projected to require additional funds.

3.8 Enterprise Constraints

- As previously mentioned, the finance platform must adhere to McMaster University's data privacy regulations and cybersecurity standards.

4 Naming Conventions and Terminology

4.1 Glossary of All Terms, Including Acronyms, Used by Stakeholders Involved in the Project

- **MES**: McMaster Engineering Society
- **IT**: Information Technology
- **UAT**: User Acceptance Testing
- **API**: Application Programming Interface
- **UI**: User Interface
- **CTA**: Call-to-Action - an element of a web-page that elicits an action from the user
- **FAQ**: Frequently asked questions

5 Relevant Facts And Assumptions

5.1 Relevant Facts

The current MES reimbursement process relying on Google Forms and Google Sheets has proven cumbersome, and as a result, students are not inclined to make timely reimbursement requests, introducing avoidable problems to the finance team. The MES does not receive its due fees from McMaster University unless they pass their audits, making the process of submitting reimbursement requests integral to the MES staying operational and receiving its due funding. Outside of efficiency and availability, the existing platform also does not include enough tracking/updates along the journey of a given request, making some information susceptible to being lost.

5.2 Business Rules

- All reimbursement requests must include a receipt or invoice as proof of purchase.
- The MES Finance team has the authority to modify the reimbursement process as needed to handle extraneous circumstances.
- The platform's functionality is subject to change in the future as per the needs of the MES.

5.3 Assumptions

- Unless otherwise stated, the MES finance team will be able to review requests in a fixed, predetermined timeline.
- Reimbursement requests can only be made with all requisite information and documentation.
- Reimbursement requests will only be made from MES Clubs/Teams to the MES.
- Internet access will be required to use the finance platform.

6 The Scope of the Work

6.1 The Current Situation

The McMaster Engineering Society (MES) currently manages financial reporting, reimbursements, and expense tracking through a disjointed system involving Google Forms, PDFs, and spreadsheets. This fragmented approach leads to inefficiencies, delays in reimbursements, errors, and increased administrative burden. MES requires a streamlined platform to centralize these processes, ensuring compliance with audit standards and reducing the time spent on financial management tasks.

6.2 The Context of the Work

The MES supports around 60 student groups and numerous individuals, organizing various academic, professional, and recreational activities. Effective financial management is essential for ensuring that these student groups receive timely funding and reimbursements. Without an efficient system, MES risks delayed payments, financial inaccuracies, and non-compliance with audit requirements. Developing a centralized platform is critical for MES to continue supporting student initiatives and ensuring smooth financial operations.

6.3 Work Partitioning

The project will be divided into the following key tasks:

- **Research and Requirements Gathering:** Analyze current MES workflows and identify the specific financial management needs of the organization.
- **Design and Architecture:** Create the architecture for a centralized financial platform, focusing on scalability, security, and integration with existing MES infrastructure.
- **Development and Implementation:** Develop core features, including reimbursement requests, payment requests, automated ledger tracking, and audit compliance monitoring.
- **Testing and Quality Assurance:** Conduct unit and integration testing to ensure the platform is robust, error-free, and complies with financial regulations.

- **Deployment and Training:** Deploy the system within MES and provide training for staff and users to familiarize them with the new platform.

6.4 Specifying a Business Use Case (BUC)

6.4.1 Business Use Case: "Submitting a Reimbursement Request"

- **Actors:** MES student group , MES finance administrator.
- **Goal:** To submit, review, track, and approve a reimbursement request for a student group expense.
- **Main Success Scenario:**
  1. The user logs into the MES financial platform.
  2. The User selects the "Reimbursement Request" option and fills out the form, attaching relevant receipts.
  3. The user is automatically notified to await approval fromthe MES finance officer.
  4. The fiinance admin is notified of the request
  5. The finance admin reviews the submission and approves the request.
  6. The reimbursement is processed, and the treasurer is notified of the successful completion.
- **Extensions:**
  - If the request lacks sufficient information, the finance officer may request additional details from the treasurer.
  - If the reimbursement exceeds a predefined limit, additional approval may be required from higher-level officers.
  - The status of the request is accesible throughout the process, i.e "awaiting confirmation"

6.4.2 Business Use Case: "Creating and Sending an Invoice Using a Pre-existing Template"

- **Actors:** Student club, MES finance officer, external client (recipient of the invoice).

- **Goal:** To create and send an invoice using a pre-existing template, simplifying the process for student clubs to bill external clients for services or events.
- **Main Success Scenario:**
  1. The student club logs into the MES financial platform.
  2. The user navigates to the "Create Invoice" section.
  3. The platform presents a set of pre-existing invoice templates that conform to MES's standardized financial format.
  4. The user selects an appropriate template, fills in the required details (e.g., client name, services rendered, amounts, due date).
  5. The invoice is automatically formatted and previewed for review.
  6. The user reviews the invoice and submits it.
  7. The platform routes the invoice to the MES finance officer for approval and verification.
  8. Once approved, the invoice is sent to the external client directly through the platform.
  9. The user and MES finance officer receive a confirmation that the invoice has been successfully sent.
- **Extensions:**
  - If any required fields are missing, the system alerts the user to complete them before submission.
  - The finance officer may request edits to the invoice if any discrepancies are found during the review process.
  - If the client is an internal MES entity, the invoice may be flagged for internal tracking and not require external delivery.
  - Automated reminders are sent to the client if payment is not received by the due date.

7 Business Data Model and Data Dictionary

7.1 Business Data Model

The Business Data Model (BDM) defines the structure and relationships between different types of data involved in the MES financial platform. Due to the early

nature of the requirements these data models are subject to change, however this serves as a good starting point.

- **Student Club:** Represents a student group affiliated with MES.
- **Reimbursement Request:** A financial record detailing a student's or club's request for reimbursement.
- **Invoice:** Represents a bill sent to an external client for services or events hosted by the student clubs.
- **External Client:** An entity or individual external to MES who receives an invoice.
- **MES Finance Officer:** A staff member responsible for approving reimbursement and payment requests.
- **Audit Log:** Tracks changes to financial records for compliance purposes.
- **Payment Request:** A request for funds that needs to be approved by an MES finance officer.

7.2 Data Dictionary

The Data Dictionary provides detailed descriptions of the key fields in the MES financial system, ensuring consistent terminology and data handling across the platform. **Please note, these fields are subject to change.**

- **Student Club ID (int):** A unique identifier and/or categorizer for each student club.
- **Reimbursement Request ID (int):** A unique identifier for each reimbursement request.
- **Invoice ID (int):** A unique identifier for each invoice.
- **Client Name (string):** The name of the external client receiving the invoice.
- **Amount (decimal):** The amount of money involved in the reimbursement request or invoice.
- **Submission Date (date):** The date a reimbursement or payment request is submitted.

- **Approval Status (string):** The status of a request, such as "Pending," "Approved," or "Rejected."
- **Audit Timestamp (datetime):** The exact time any modification is made to a financial record.
- **Payment Due Date (date):** The date by which payment is expected for an invoice.
- **Club Treasurer (string):** The name of the user responsible for submitting financial requests on behalf of a student club.

8 The Scope of the Product

8.1 Product Boundary

The finance platform will focus on handling reimbursement requests, invoice generation, and financial tracking for MES-affiliated student groups. It will automate the submission, review, and approval of reimbursement requests while integrating with existing MES infrastructure. The platform will not support unrelated financial operations such as payroll or student tuition payments.

8.2 Product Use Case Table

ID	Use Case
PUC-1	Submit a reimbursement request
PUC-2	Review and approve reimbursement requests
PUC-3	Generate an invoice for services
PUC-4	Track the status of a reimbursement
PUC-5	Access financial documentation for audit purposes

8.3 Individual Product Use Cases (PUC's)

PUC-1: Submit a Reimbursement Request

- **Primary Actor**: Student (Club Member)
- **Goal**: Submit a reimbursement request for club-related expenses
- **Preconditions**:

  - User is logged in and is a member of a registered MES club
  - User has relevant documentation, such as receipts
- **Main Success Scenario**:
  1. User accesses the reimbursement submission form
  2. User fills in required details (club affiliation, amount, etc.)
  3. User uploads the receipt
  4. User submits the form
  5. System validates the submission and confirms receipt
  6. System forwards the request to the finance team for review
- **Postconditions**:
  - The reimbursement request status is set to "Submitted"
  - User can track the request's progress in the system
- **Extensions**:
  - If validation fails, the system provides feedback on missing or incorrect fields
  - If the amount exceeds a certain threshold, additional approval may be required

PUC-2: Review and Approve Reimbursement Requests

- **Primary Actor**: MES Finance Staff
- **Goal**: Review and approve submitted reimbursement requests
- **Preconditions**:
  - Reimbursement request has been submitted by a user
  - Finance staff has the necessary access permissions
- **Main Success Scenario**:
  1. Finance staff logs into the platform
  2. Staff reviews the reimbursement details and attached documentation

3. Staff approves or rejects the request
4. If approved, the system updates the status to "Approved"

- **Postconditions**:
  - Request is moved to the appropriate workflow step based on approval status
- **Extensions**:
  - If the request is missing documentation, the staff can request additional information
  - If the request is rejected, the system notifies the submitter with the reason for rejection

9 Functional Requirements

9.1 Functional Requirements

9.1.1 FRs for Reimbursement and Payment Request Feature

- **FR1.1.** The system must allow users to submit reimbursement/payment requests by inputting relevant information, such as budget line items, receipts, documents, and payment details.
- **FR1.2.** The system must allow admins to review and approve/reject submitted requests.
- **FR1.3.** The system will enable users and admins to edit submitted requests.
- **FR1.4.** The system must track and display the status of each submitted reimbursement/payment request (submitted, under review, approved/rejected, disbursed/completed, requires attention)
- **FR1.5.** The system shall notify users of any changes to the status of their submitted request via email or SMS.
- **FR1.6.** The system must store and categorize submitted reimbursement/payment request information, linking each request to the appropriate budget category for ledger tracking.

- **FR1.7.** The system shall restrict access to view sensitive financial information submitted by users based on security roles. General users must only have access to view the requests they submitted. Admin users must have access to view all submitted requests.
- **FR1.8.** The system must allow multiple admins to collaborate and review submitted requests without data conflicts
- **FR1.9.** The system shall track and store all actions performed concerning a single reimbursement or payment request. Actions to be tracked include submission, editing, review, approval/rejection, disbursement, and withdrawal/deletion.

9.1.2 FRs for Custom Invoice Generation Feature

Note: this may become a stretch goal later on, depending on how the project progresses.

- **FR2.1.** The system shall allow users to generate custom invoices by allowing users to input relevant information such as budget categories, and payment details/amounts into a pre-built template.
- **FR2.2.** The system shall enable the customization of invoice templates, including specific logos and details, for all 60 MES clubs and individuals.
- **FR2.3.** The system will automatically incorporate applicable GST/HST numbers into each custom-generated invoice based on MES's tax regulations and standards.
- **FR2.4.** The system will securely store generated invoices to allow users to view templates they have generated, and admins to view all templates generated by all users.
- **FR2.5.** The system shall allow users to edit and reuse invoice templates that have been previously generated.
- **FR2.6.** The system must allow users to preview the custom invoice visually before generation.
- **FR2.7.** The system must store and categorize generated invoice payment information, syncing with MES's financial ledger and linking to the appropriate budget category.

9.1.3 General Functional Requirements

- **FR3.1.** The system shall securely store account or profile information on each user.
  - Explanation: basic account information includes full name, email, password, and phone number. (subject to change)
- **FR3.2.** The system shall authenticate users to ensure they are registered students of McMaster University.
- **FR3.3.** The system shall allow users to edit their profile information.

10 Look and Feel Requirements

10.1 Appearance Requirements

- **APR1.** The system's UI shall be responsive across various devices and screen sizes (desktop, tablet, mobile).
  - Explanation: no information or functionality should be unavailable to the user because of varying device screen sizes.
- **APR2.** The system's UI shall integrate MES branding throughout its UI screens, including MES brand colours, and logos.

10.2 Style Requirements

- **STYR1.** The system's UI shall incorporate universally recognizable icons, with at least 1 icon being used per screen.
- **STYR2.** The system's UI shall maintain a typography standard by using no more than 2 different fonts across the platform. The system shall use standard fonts used by the MES (if applicable).

11 Usability and Humanity Requirements

11.1 Ease of Use Requirements

- **EUR1.** Users shall be able to complete a specific task, such as editing a request, with no more than 3 CTAs.

- **EUR2.** For all input fields, the system shall include error messages to guide users in correcting input errors.
- **EUR3.** Users shall be able to search for and locate a piece of information within the system's UI within 10 seconds.

11.2 Personalization and Internationalization Requirements

- **PIR1.** Users shall be able to edit their account/profile information with less than 3 clicks or actions from the landing page.
- **PIR2.** Personalization actions shall be reflected on the UI within 2 seconds of a user action.

11.3 Learning Requirements

- **LER1.** All learning materials, including user documentation and FAQs shall provided on the landing page and accessible to the user at any time.
- **LER2.** The system shall be learnable such that users repeating a specific task within the system for the third time do not need to reference user documentation.

11.4 Understandability and Politeness Requirements

- **UAPR1.** All text and instructions within the system's UI will be written in plain language and use no technical jargon.

11.5 Accessibility Requirements

- **ACSR1.** The system should allow for keyboard navigation within input forms.
- **ACSR2.** All text should be no less than size 12 font to ensure all content is readable.

12 Performance Requirements

12.1 Speed and Latency Requirements

- **SPLR1.** The system shall process payment requests within 5 seconds.

- **SPLR2.** Invoices shall be generated within 3 seconds.
- **SPLR3.** Reimbursement requests shall be processed within 5 seconds.

12.2 Safety-Critical Requirements

- **SFCR1.** Sensitive financial information such as credit card numbers shall be encrypted when stored.
- **SFCR2.** Access controls shall restrict access to confidential information.

12.3 Precision or Accuracy Requirements

- **POAR1.** Monetary values would be rounded off to the nearest cent (i.e., x.yz)

12.4 Robustness or Fault-Tolerance Requirements

- **RFT1.** Backup servers shall store critical information in case of failure, including daily backups.
- **RFT2.** Graceful error handling would preserve sensitive information in case of improper use.

12.5 Capacity Requirements

- **CPR1.** The system shall be able to handle 1000 reimbursement requests, 1000 invoices, and 500 payment requests daily.
- **CPR2.** The system would be able to handle up to 10 requests simultaneously.
- **CPR3.** The system shall be able to store up to 10 years' worth of backed-up data.

12.6 Scalability or Extensibility Requirements

- **SER1.** API integration would support future third-party connectivity.
- **SER2.** The system would be sufficiently scalable so as to be able to accommodate double its current maximum usage within one year.

12.7 Longevity Requirements

- **LOR1.** The system shall be serviceable, at its current usage rate, for at least the next 10 years without needing considerable updates or overhauls made to it. MES plans to continue working on the platform as needed after the completion of this capstone. As such, maintainability of this platform is an important consideration.

13 Operational and Environmental Requirements

13.1 Expected Physical Environment

- **OR1.** The system shall run on Windows 10 and 11 operating systems.
- **OR2.** The system shall require similar minimum specs to Intuit Quickbooks to run.

13.2 Wider Environment Requirements

- **WR1.** Financial institutions integrated with the payment/reimbursement features shall make use of industry-standard systems and processes for Canadian banks.

13.3 Requirements for Interfacing with Adjacent Systems

- **IAR1.** The system shall be adaptable enough to gracefully handle interactions with different financial institutions.
- **IAR2.** The API shall be adaptable for use with other (Future) third-party connections.

13.4 Productization Requirements

- **PRD1.** Ensure server and database can handle increased usage and data with automated scaling capabilities.
- **PRD2.** Implement data encryption, secure authentication (e.g., OAuth, 2FA)
- **PRD3.** Develop an intuitive interface with role-based access control (RBAC) and adhere to WCAG accessibility guidelines.

- **PRD4.** Ensure compatibility with MES infrastructure (e.g., DigitalOcean) and integrate with external services (e.g., payment processors, audit tools).
- **PRD5.** Provide comprehensive documentation, automated testing (CI/CD) via GitHub Actions, and ensure error handling with clear logs and user feedback.
- **PRD6.** Create training materials for users and provide help desk support during the transition to the new platform.
- **PRD7.** Monitor infrastructure costs, plan for scalable cloud resource usage, and design the platform to be modular and future-proof.

13.5 Release Requirements

- **RLR1.** Stable releases shall be released twice a year.

14 Maintainability and Support Requirements

14.1 Maintenance Requirements

- **MR1.** Data backups shall be performed daily to protect sensitive information.
- **MR2.** The system shall experience scheduled maintenance downtime for 10 percent of each year (specifics to be determined).
- **MR3.** Security updates and patches shall be released as needed (separate from the twice-yearly stable releases).

14.2 Supportability Requirements

- **SR1.** Documentation for users and the team shall be kept up to date as new releases are made.

14.3 Adaptability Requirements

- **AR1.** The system shall be robust enough to adapt to changes from external and connected sources such as McMaster University or Canadian financial institutions without losing core functionalities.

15 Security Requirements

15.1 Access Requirements

- **SCR1.** The system shall ensure that users only have access to the data they are authorized to.
- **SCR2.** The system shall have a secure login portal to prevent unauthorized access.

15.2 Integrity Requirements

- **INR1.** The system shall log 100% of transactions to ensure financial history is traceable. To ensure complicity with audit requirements.

15.3 Privacy Requirements

- **PVR1.** The system shall encrypt sensitive information in transit and at rest.
- **PVR2.** The system shall return anonymous data when generating reports that do not require personal info.

15.4 Audit Requirements

- **ADR1.** The system shall maintain a detailed log of 100% of transactions to ensure financial transparency.
- **ADR2.** The system shall retain audit logs for a minimum of 10 years.

15.5 Immunity Requirements

- **IMM1.** The system shall be resilient to data corruption, ensuring that data can be recovered within 4 hours of a failure.

16 Cultural Requirements

16.1 Cultural Requirements

- **CLTR1.** The system shall be scaleable to implement future multi-language support to accommodate international students or staff.

17 Compliance Requirements

17.1 Legal Requirements

- **LR1.** The system shall follow user data protection laws to ensure proper handling of sensitive information.
- **LR2.** The system shall follow any relevant financial regulations for non-profit organizations.

17.2 Standards Compliance Requirements

- **STR1.** The system shall follow web security standards.

18 Requirements Traceability Tables

Table 2: Traceability Matrix between NFRs and FRs (Part 1)

NFR / FR	**FR1.1**	**FR1.2**	**FR1.3**	**FR1.4**	**FR1.5**	**FR1.6**	**FR1.7**	**FR1.8**	**FR1.9**
APR1	X	X	X	X	X	X	X	X	X
APR2	X								
STYR1	X								
EUR1			X						
EUR2	X		X						
EUR3				X					
ACSR1	X		X	X					
SPLR1	X								
SFCR1	X					X	X		
SCR1		X					X		
INR1		X	X	X	X	X	X	X	X
STR1	X	X	X	X	X	X	X	X	X

Table 3: Traceability Matrix between NFRs and FRs (Part 2)

NFR / FR	FR2.1	FR2.2	FR2.3	FR2.4	FR2.5	FR2.6	FR2.7
APR1	X	X	X	X	X	X	X
APR2	X						
STYR1	X						
EUR1					X		
EUR2	X				X		
ACSR1	X				X		
SPLR2	X					X	
SFCR1	X			X			X
INR1	X	X	X	X	X	X	X
STR1	X	X	X	X	X	X	X

Table 4: Traceability Matrix between NFRs and FRs (Part 3)

NFR / FR	FR3.1	FR3.2	FR3.3
APR1	X	X	X
EUR1			X
EUR2			X
PIR1			X
PIR2			X
ACSR1			X
SCR2	X	X	X
INR1	X	X	X
STR1	X	X	X

19 Open Issues

The key open issues currently revolve around undecided functionality, particularly regarding the integration of automated reporting features and the appropriate level of customization for the needs of various student groups. Other considerations include:

- How granular the platform should be in tracking compliance for audit purposes.
- Whether the system will support all financial workflows such as reimbursement requests, intramural funding, and payment tracking.
- Ensuring that the user experience is intuitive for both administrators and students.

These decisions will significantly influence the overall architecture of the solution and the development timeline.

20 Off-the-Shelf Solutions

Accounting and invoicing software is readily available, however MES refrains from implementing these existing solutions due to a variety of factors. Commonly cited concerns include:

- Lack of control over platform customization to meet the specific needs of MES.
- High subscription fees that are unsustainable for a non-profit organization.
- Data privacy concerns, especially in handling sensitive financial information.
- Limited support for integration with existing MES infrastructure.

20.1 Ready-Made Products

Examples of existing financial management tools that were considered but not implemented include:

- **Wave Accounting** (https://www.waveapps.com/) - A free accounting platform suited for small businesses but lacks customization for non-profit needs.
- **FreshBooks** (https://www.freshbooks.com/) - A paid tool offering invoicing and expense tracking, but deemed too expensive and restrictive for MES.
- **Xero** (https://www.xero.com) - Cloud based accounting platform, for invoicing, inventory and bank reconcillation
- **QuickBooks Online** (https://quickbooks.intuit.com/) - Widely-used platform for small to mid sized businesses

20.2 Reusable Components

There are numerous reusable components that could be adapted to suit the needs of the MES. These include:

- **LedgerSMB** (https://ledgersmb.org/) - An open-source accounting tool that could be modified for more specific MES needs.

- **Plaid API** (https://plaid.com/docs/api/) - For integrating bank data and managing student group transactions.
- **Stripe API** (https://docs.stripe.com/api) - For handling payments and reimbursements.

20.3 Products That Can Be Copied

While no single product fully meets MES requirements, certain features from established platforms could inspire the design of the financial management system:

- **QuickBooks** - Budgeting tools and expense tracking can provide a model for managing student group funds.
- **Expensify** - Reimbursement workflows and receipt tracking features.

21 New Problems

21.1 Effects on the Current Environment

The implementation of a new centralized platform for financial management may affect MES's current environment in the following ways:

- Increased demand on server resources, especially as more financial processes move from local and manual to online and automated.
- Potential need for server upgrades if the platform requires more CPU, RAM, or storage than the existing DigitalOcean infrastructure provides.

21.2 Effects on the Installed Systems

MES currently uses DigitalOcean for its infrastructure. Introducing a financial management platform could result in:

- Increased usage limits, potentially necessitating the purchase of higher-tier virtual machines.
- A larger database capacity for tracking all transactions and handling potentially higher loads during peak reporting periods.

21.3 Potential User Problems

User challenges may arise from:

- The steep learning curve for administrators and student groups transitioning from the current system of Google Forms, PDFs, and spreadsheets to a new, fully automated platform.
- Potential resistance from users who are accustomed to the manual processes and may find it difficult to adapt to a new system.

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

Potential limitations include:

- Resource constraints on the DigitalOcean infrastructure, which could limit the performance and scalability of the platform.
- Increased operating costs due to the need for more powerful virtual machines or cloud services.

21.5 Follow-Up Problems

After implementation, follow-up issues could arise, such as:

- Reconciling the codebase of the new platform with existing MES systems and ensuring seamless integration.
- Ensuring that the new platform adheres to financial compliance and audit standards, particularly in the first few months of operation.

22 Tasks

22.1 Project Planning

Key tasks in project planning include:

- Defining the scope of the system and establishing clear, achievable goals for each phase of development.
- Setting a timeline for system rollout, including key milestones like system design, development, testing, and deployment.

22.2 Planning of the Development Phases

Development should proceed in phases:

- **Phase 1: (Present - Nov)** Initial research and requirements gathering from MES staff and student groups.
- **Phase 2: (Dec - March)** Core system design, focusing on the backend ledger management and reimbursement workflows. This phase would be the most intensive.
- **Phase 3: (Feb - March)** UI/UX design to ensure the platform is user-friendly and accessible to all MES members.
- **Phase 4: (Feb - March)** System testing and debugging, including user feedback to refine functionality.
- **Phase 5: (March - April)** Final deployment and training for users.

23 Migration to the New Product

23.1 Requirements for Migration to the New Product

To successfully migrate to the new financial platform, MES will need to:

- Train existing staff and student group representatives on how to use the new system.
- Ensure all historical financial data is correctly transferred to the new system and remains accessible for auditing purposes.

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

The data to be migrated includes:

- Historical financial data such as past reimbursements, transactions, and budget allocations.
- Any audit-related data to ensure compliance with financial reporting requirements.

24 Costs

While MES already has existing infrastructure on DigitalOcean, additional costs may arise from:

- Server upgrades to accommodate increased resource demands from the new platform.
- Potential third-party services for payment processing or APIs (e.g., Plaid, Stripe).

25 User Documentation and Training

25.1 User Documentation Requirements

User documentation must:

- Be clear, concise, and accessible to both administrators and student group representatives.
- Include examples for common tasks, such as submitting reimbursements and tracking group budgets.
- Be easy to navigate, with a search feature and organized sections for quick reference.

25.2 Training Requirements

Training will involve:

- "Getting Started" guides for both administrators and general users.
- Video tutorials or live training sessions to walk users through the system's core functions.
- Ongoing support through help desks, FAQs, and troubleshooting documentation.

26 Waiting Room

The following features are under consideration for future development but are not part of the immediate project scope:

- SMS notifications to update users on the status of reimbursement requests.
- Authentication via McMaster CAS 2FA (Central Authentication Service) for enhanced security.
- Integrated Phone Application

27 Ideas for Solution

The proposed solution involves leveraging a form of the traditional web full stack, potentially (MongoDB, Express, NextJs, React) to handle both frontend and backend requirements. This stack allows for:

- Efficient data integration with existing MES systems.
- A scalable, modular design that can be extended with additional features like payment APIs or compliance tools.
- Maintainability after the capstone is complete

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 went well while writing this deliverable?

   Jacob: Writing the requirements went well as requirements have been discussed and practiced in several prior courses, such as 3A04 and 3RA3

   Christian: Sections 1-6 required a lot of information that had already been discussed in prior meetings, so their inclusion was more of a formality.

   Evan: Splitting up the work was once again an easy process for this deliverable. Everyone was present for our meetings and active in our group chat, making it easy to communicate and stay on the same page.

   Austin: The proactiveness and eagerness as a team to setup frequenent meetings with our supervisor to really ensure we get our requirements accurately represented the needs of our stakeholders.

   Adam: Writing the functional requirements seemed fairly straightforward. The things that the system must do functionally are clear to us at the current stage of the project.

2. What pain points did you experience during this deliverable, and how did you resolve them?

   Christian: Some parts of Section 3 (Mandated Constraints) I was unsure if I had covered all bases of what constraints we would work under, and opted to mention only the constraints that felt imposed on us by the situation and stay away from potentially detailing more requirements rather than constraints.

   Jacob: The main pain points were from NFRs where I did not understand the definition. I solved it by looking at old lecture slides that explained what each non-functional requirement entailed. 13.4 was also a pain point as "productization" is not a word, so I was unable to generate any requirements, but I just made a note of that.

   Evan: The biggest pain point of this deliverable was how close the due date was to our TA meeting. Thankfully, the deadline was extended but we got a lot of feedback from our TA meeting that we would have had to implement within two days. For future deliverables, spacing the TA meeting to the deliverable deadline might help students to implement the TA feedback.

Austin: A large painpoint was wrapping my head around the meaning of some of the requirments that come from the volere template, leveraging past projects as well as generate AI to just get the gears turning was helpful.

Adam: A major pain point for me was writing some of the non-functional requirements. The names/labels given for the NFRs weren't always clear. For example "Personalization and Internationalization Requirements". I was not entirely clear what was meant by "internationalization", but I did my best to come up with requirements that pertained to at least one of those things. It would have been helpful if this template had a brief description for each section.

3. How many of your requirements were inspired by speaking to your client(s) or their proxies (e.g. your peers, stakeholders, potential users)?

   Group: Several requirements relating to security and scalability were generated or inspired by talking to Luke. However, the functional requirements would be more heavily drawn from speaking with stakeholders, as the focus during these discussions was functionality of the system rather than its characteristics.

4. Which of the courses you have taken, or are currently taking, will help your team to be successful with your capstone project?

   Group: Our engineering design courses, as well as requirements and design courses such as 3XB3, 3RA3, and 2AA4 will help us be successful. User Interfaces 4HC3, important in designing UI.

5. 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.

   Christian: The ability to culminate domain knowledge from different aspects of software development that we've been taught throughout courses into a full-fledged software product for a client will be crucial to the flow of the project. We have all learned about databases, cybersecurity, object-oriented programming, data structures, etc., and have even developed outside of school. However, the ability to bring this knowledge together in a cohesive and intuitive way

will avoid countless problems during development concerning the integration of multiple technologies together.

Austin: A mindset: being able to overcome the anxiety that can arise from approaching a problem that you have never seen before especially at this scale. For most team members this is the largest scale project to have be worked on. This can obvioiusly be daunting. A message to those that feel this way, just dive head first, learn and fail. Each failure is a stepping stone to success, do not be afraid to fail.

Jacob: For most of us this will be our first experience building a software system of this scale, so fullstack development knowledge will be necessary in order to succeed in this project.

Evan: Software development knowledge will obviously be essential to completing this project. Specifically, we will likely need to acquire knowledge of security principles and best practices. This project has us working with sensitive information so ensuring compliance with privacy regulations will take some research.

Adam: A lot of this project will be heavily reliant on full-stack development and the technologies relating to it. I think some of the biggest challenges will come from having to integrate all these technologies together and getting them to work in unison. With that being said, the team will collectively need to acquire skills in the frontend, backend and database technologies that are required for the project, like React, Next.js, TypeScript, and MongoDB. The team will also need to gain knowledge on how to integrate with existing applications that the MES is using like DigitalOcean. A big learning curve for me personally will be learning how to push the system with all its parts (frontend, backend, database) to a production environment and figuring out how to accurately test it.

6. 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?

   Christian: Two approaches to familiarizing ourselves with full stack development could be researching existing software systems and how their design and structure work together. Another approach could be taking the technology one is most familiar with and learning how it integrates with other technologies, to start from a place of familiarity.

Austin: Well to reiterate again, I think this choice of "skill" is widely applicable to many areas of life, the capstone provides a safe mechanism for learning this. An approach to aquiring the skill of facing a challenge is to embrace the mindset of continuous failure towards eventual success. (*A bit unorthodox of a response, but I still think is equally valuable*)

Jacob: One approach to gain fullstack development knowledge would be to gain experience by practicing small-scale development or expanding on prior projects. Another way would be by watching tutorials on the internet or reading textbooks.

Evan: Two approaches to familiarizing ourselves with implementing security principles can involve pursuing online training or in-person workshops. Online training can involve online courses, YouTube videos or any other online certifications that teach about cybersecurity. The benefit of in-person workshops or hackathons is the ability to network and learn hands-on since it would be in-person.

Adam: There's many ways to acquire skills in full-stack development. A great source for learning and familiarizing yourself with new technologies is Youtube tutorials. That's what I will most likely pursue to gain knowledge on specific technologies like MongoDB and Next.JS. Once I have a general understanding of the technology, another good approach to acquiring and solidifying the skill is to build a small side project that uses the tech stack. Out of these two approaches, I think I will start with the first one, and if there is time I can build a small side project in my free time.