Verification and Validation Report: McMaster Engineering Society Custom Financial Expense Reporting Platform

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

March 10, 2025

1 Revision History

Date	**Version**	**Notes**
Date 1	1.0	Notes
Date 2	1.1	Notes

2 Symbols, Abbreviations and Acronyms

symbol	**description**
T	Test

Contents

List of Tables

List of Figures

This document ...

3 Functional Requirements Evaluation

4 Nonfunctional Requirements Evaluation

4.1 Usability

4.2 Performance

4.3 etc.

5 Comparison to Existing Implementation

This section will not be appropriate for every project.

6 Unit Testing

Most files, excluding the 3rd-party library, top-level and GUI modules had corresponding unit tests. All tests were performed using jest for testing javascript, and each commit to the main branch must pass all the tests. All the tests passed as seen in the Test Report 12.1.

7 Changes Due to Testing

8 Automated Testing

The tests were set up to automatically run in GitHub Actions whenever a commit was pushed to the `main` branch. The configuration for the CI/CD automation can be found at https://github.com/ausbennett/mes-finance-platform/blob/main/.github/workflows/test.yml and the test pipeline can be viewed at https://github.com/ausbennett/mes-finance-platform/actions/workflows/test.yml.

9 Trace to Requirements

9.1 Trace table

Test Suite	Functional Requirement
emailer.test.js	FR1.5, FR3.2
reconciler.test.js	FR1.1, FR1.4, FR1.6, FR1.9, INR1, SPLR1-3
requests.test.js	FR1.2, FR1.3, FR1.8, PVR1
Account management test suite (TBD)	FR1.7, FR3.1, FR3.2
Usability testing (via checklist)	APR1, APR2, STYR1, STYR2, EUR1-3, PIR1, LER2, UAPR1, ACSR1, SCR1-5, CLTR1
Static analysis (via checklist)	POAR1, LOR1, WR1, RLR1, ACSR2, MR2, SR1, LR1, LR2, STR1
Load testing	CPR1-3, SER2

Table 1: Mapping between test suites and requirements

10 Trace to Modules

Test Suite	Module
emailer.test.js	Notification Module (M4), Authentication Module (M6), Emailer API (M7)
reconciler.test.js	Clubs, User, Request Database(M10-12)
requests.test.js	Requests Module(M3), Requests Controller (M9)
Account management test suite (TBD)	Account Management Module (M2), User Dashboard Module (M5), Account Management Controller (M8)
Usability testing (via checklist)	Graphical User Interface (M13)

Table 2: Mapping between test suites and modules

11 Code Coverage Metrics

12 Appendix

12.1 Test Report

```
  PASS  tests/requests.test.js
   Request Endpoints and Model Validations
     Model Validations
       should require necessary reimbursement fields (3 ms)
     Reimbursement Endpoints
       should create reimbursement with file upload (34 ms)

Test Suites: 1 passed, 1 total
Tests:       2 passed, 2 total
```

```
Snapshots:   0 total
Time:        0.927 s, estimated 1 s
Ran all test suites.
```

Appendix — Reflection

The information in this section will be used to evaluate the team members on the graduate attribute of Reflection.

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. What went well while writing this deliverable? Jacob: What went well for me was being able to easily find and reference our prior documentation to fill out this deliverable as consistently as possible.

2. What pain points did you experience during this deliverable, and how did you resolve them? Jacob: the biggest pain point for me was the terrible implementation of charts and tables in LateX. I resolved this by being patient and wishing we could just use Word like the rest of society.

3. Which parts of this document stemmed from speaking to your client(s) or a proxy (e.g. your peers)? Which ones were not, and why?

4. In what ways was the Verification and Validation (VnV) Plan different from the activities that were actually conducted for VnV? If there were differences, what changes required the modification in the plan? Why did these changes occur? Would you be able to anticipate these changes in future projects? If there weren't any differences, how was your team able to clearly predict a feasible amount of effort and the right tasks needed to build the evidence that demonstrates the required quality? (It is expected that most teams will have had to deviate from their original VnV Plan.)