Hazard Analysis
Software Engineering

Team #12, Streamliners
Mahad Ahmed
Abyan Jaigirdar
Prerna Prabhu
Farhan Rahman
Ali Zia

Table 1: Revision History

Date	**Developer(s)**	**Change**
2025-10-02	Abyan	Scope and Purpose
2025-10-02	Farhan	Critical assumptions made for the project
2025-10-01	Prerna	Add initial version of introduction
Date2	Name(s)	Description of changes
...	...	...

Contents

1 Introduction

This document outlines the hazard analysis of the **Large Event Management System (LEMS)** for the McMaster Engineering Society (MES). The LEMS is a centralized platform designed to streamline event workflows such as **ticket sales, registrations, waivers, payments, bus/table sign-ups, notifications, and event check-ins**.'

For this deliverable, the analysis will specifically focus on the features within the scope of **Project B** which include:

- Payment System
- Role-Centric/Feature-Based Access Control (RBAC/FBAC)
- Bus Sign-Ups
- RSVP Sign-Ups
- Table Sign-Ups

Hazards in this system refer to conditions or failures that may lead to financial loss, security/privacy breaches, accessibility failures, or reputational damage for MES and McMaster University. While the system does not directly involve hardware or pose risks of physical harm, software risks are significant. These include data loss, payment errors, registration corruption, check-in failures, and unauthorized access to sensitive information.

The purpose of this hazard analysis is to:

- Identify risks associated with the Payment, Access Control, and Signup modules.
- Define their potential impacts on users, organizers, and the MES organization.
- Introduce mitigation strategies by specifying safety and security requirements that will guide system design and implementation.

By proactively addressing these hazards, this analysis ensures the system will be reliable, secure, accessible, and trustworthy for both organizers and attendees.

2 Scope and Purpose of Hazard Analysis

The purpose of this hazard analysis is to identify and evaluate potential hazards that could arise during the development and operation of the Large Event Management System (LEMS). Since LEMS is a software-based platform that supports event registration, ticketing, payments, and participant management,

hazards are primarily related to data integrity, system availability, security, and user interactions. By analyzing these risks early, the project team can define mitigation strategies, incorporate safety and security requirements into the design, and reduce the likelihood of organizational or reputational harm.

The scope of this analysis covers all major components of LEMS, including the backend services, web and mobile applications, and the database. It considers hazards introduced by user error, software defects, integration failures across modules, and security vulnerabilities. Hazards that fall outside the team's control, such as third-party cloud hosting failures or issues with external payment providers (e.g., Stripe), are acknowledged but not analyzed in detail.

The goals of this hazard analysis are:

- To proactively identify risks related to security, data integrity, privacy, and availability in LEMS.
- To evaluate the potential consequences of failures, such as data loss, financial errors, or unauthorized access to sensitive information.
- To define mitigations or controls that reduce these risks to acceptable levels.
- To guide architectural and design decisions and support the definition of non-functional requirements such as reliability, security, and maintainability.

This hazard analysis will be refined as the project design evolves, ensuring that new risks are addressed as they emerge. It is part of the broader quality assurance process and ensures that LEMS meets the standards of safety, security, and reliability expected by the McMaster Engineering Society.

3 System Boundaries and Components

[Dividing the system into components will help you brainstorm the hazards. You shouldn't do a full design of the components, just get a feel for the major ones. For projects that involve hardware, the components will typically include each individual piece of hardware. If your software will have a database, or an important library, these are also potential components. —SS]

4 Critical Assumptions

The following assumptions have been identified as critical to the safe and reliable operation of the MacSync platform.

- **Reliable Internet Access:** It is assumed that both attendees and organizers will have access to stable internet connections during registration,

payment, and check-in. While temporary connectivity loss may occur, the system must handle these cases gracefully.

- **Third-Party Service Availability:** The platform depends on external services such as payment processors (e.g., Stripe, PayPal) and hosting infrastructure. It is assumed these services provide high availability, but the system will still account for outages or delays to prevent complete operational failure.
- **Device Compatibility:** It is assumed that attendees will primarily use modern smartphones and organizers will have access to laptops or mobile devices capable of running the dashboard. Reliance on outdated devices or unsupported browsers must be minimized through compatibility testing.
- **User Data Accuracy:** The system assumes that users provide correct information (e.g., dietary restrictions, accessibility needs, payment details). However, hazards tied to incorrect or incomplete inputs will be addressed by validation checks.
- **Organizational Oversight:** It is assumed that event organizers will actively monitor the system for anomalies (e.g., payment disputes, capacity errors, failed notifications). This system will assist and automate many of the tasks and centralize information, but human oversight is still necessary to manage unexpected situations.
- **Security Measures:** It is assumed that standard security practices (encrypted storage, secure authentication, and role-based access control) will be implemented and maintained. Failure to enforce these could expose sensitive student data or enable fraudulent event access.

5 Failure Mode and Effect Analysis

[Include your FMEA table here. This is the most important part of this document. —SS] [The safety requirements in the table do not have to have the prefix SR. The most important thing is to show traceability to your SRS. You might trace to requirements you have already written, or you might need to add new requirements. —SS] [If no safety requirement can be devised, other mitigation strategies can be entered in the table, including strategies involving providing additional documentation, and/or test cases. —SS]

6 Safety and Security Requirements

[Newly discovered requirements. These should also be added to the SRS. (A rationale design process how and why to fake it.) —SS]

7 Roadmap

[Which safety requirements will be implemented as part of the capstone timeline? Which requirements will be implemented in the future? —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. What went well while writing this deliverable?
2. What pain points did you experience during this deliverable, and how did you resolve them?
3. Which of your listed risks had your team thought of before this deliverable, and which did you think of while doing this deliverable? For the latter ones (ones you thought of while doing the Hazard Analysis), how did they come about?
4. Other than the risk of physical harm (some projects may not have any appreciable risks of this form), list at least 2 other types of risk in software products. Why are they important to consider?