Hazard Analysis
ProgName

Team #25, The Crazy Four

Ruida Chen
Ammar Sharbat
Alvin Qian
Jiaming Li

Table 1: Revision History

Date	**Developer(s)**	**Change**
Date1	Name(s)	Description of changes
Date2	Name(s)	Description of changes
...	...	...

Contents

1 Introduction

For a card game that entertains users by applying different game rules in different numerical systems to teach various number bases, reliability of the software and the user safety are neccessary attributes. In this context, a hazard is defined as any situation within the system or in the surrounding environment that may cause harm, such as confusion, data loss or resulting in incorrect learning outcomes. In this project, hazards are not limited to physical concerns; they mainly focus on software-related risks. These issues may cause users to become frustrated or reduce the educational value of the game. The Failure Modes and Effects Analysis (FMEA) method is used to identify and evaluate these possible failure modes within key components, guiding the team to improve system fairness, learning accuracy, and overall robustness.

2 Scope and Purpose of Hazard Analysis

The purpose of this hazard analysis is to identify components of the card game system that may lead to undesirable outcomes and to understand the potential losses that could occur if such hazards are realized. These losses may include:

- Loss of learning effectiveness, where players receive incorrect results or misleading explanations about numeral bases.
- Loss of user trust or engagement, caused by unfair scoring, interface errors, or game desynchronization.
- Loss of data integrity, resulting in corrupted game records or incorrect user progress tracking.
- Loss of system availability, due to server failure or communication breakdowns during active sessions.

The analysis considers both situations when the system functions incorrectly and when it behaves as designed but under unexpected conditions (e.g., network lag or user misuse). By identifying potential hazards early, this document supports the design of a more stable, accurate, and user-centered learning experience.

3 System Boundaries and Components

3.1 Frontend

This component provides the main interface for players to interact with the system, it captures user inputs such as playing, drawing, or skipping cards and provides visual feedback and animations during gameplay.

3.2 Backend Server

This component manages the overall system logic, coordinating user sessions, handle requests, and enforcing gameplay.

3.3 Database

This component stores persistent data such as player account information, game history, and user configuration settings.

3.4 Real-Time Communication Layer

This component maintains live connections between clients and the server to synchronize player actions in real time.

3.5 Game Rule Module

This component enforces the game's core rules, such as base conversion calculations, card validation, and scoring.

3.6 Auth & User Management System

This component handles user registration, login, and session verification.

3.7 External Libraries and APIs

This component relies on several open-source frameworks and libraries.

4 Critical Assumptions

[These assumptions that are made about the software or system. You should minimize the number of assumptions that remove potential hazards. For instance, you could assume a part will never fail, but it is generally better to include this potential failure mode. —SS]

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?

    - Usability and human-error risks: Poorly designed user interfaces, low-quality GUI animation, unclear instructions can cause user to make mistake or misunderstand the system, these risks are important because they directly affect user satisfaction, learning outcomes and overall quality of the software.
    - Data integrity and privacy risks: These occur when user data is lost, corrupted or exposed due to software defects, inappropriate authentication. Such risks are critical because they can lead to permanent loss of user information and legal issues, breaches can severely damage credibility and violate data protection regulations.