Hazard Analysis
ProgName

Team #25, The Crazy Four

Ruida Chen
Ammar Sharbat
Alvin Qian
Jiaming Li

Table 1: Revision History

Date	Developer(s)	Change
10.9	Jiaming Li	FMEA
10.9	Ruida Chen	Introduction, Scope and Purpose, Components, Reflection Q4
10.9	Alvin Qian	Assumptions
10.14	Ammar Sharbat	Safety and Security Requirements, Roadmap, Reflection Q1 and Q2

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

The assumptions below focus on a realistic deployment and user population.

Assumptions deliberately not made

- We do not assume client input is trusted. All moves are validated on the server.
- We do not assume stable connectivity for the entire session. Rejoin and reconciliation are required features.
- We do not assume prior knowledge of base 12. In app explanations and hints are mandatory.

Table 2: Critical Assumptions and Implications

ID	Assumption	Rationale	If violated
A1	Users access the game on a supported desktop or laptop browser (Chrome, Firefox, Edge) with JavaScript, cookies, and local storage enabled.	Limits test matrix and aligns with target environment.	UI features malfunction, loss of state persistence, inaccessible components.
A2	Average network latency under 300 ms and packet loss under 1% during play.	Real time turns and fairness depend on timely messages.	Turn desync, duplicate actions, stale state decisions.
A3	Server and database share a synchronized time source via NTP and timeouts use a monotonic clock.	Consistent session expiry and replay ordering.	Expired tokens accepted, rejoin windows miscalculated, audit confusion.
A4	WebSocket over TLS 1.2 or higher is available from client networks.	Required for live play and low latency.	Falls back to polling or blocks play which raises latency and server load.
A5	Database durability is at least write ahead logging with fsync on commit.	Protects match history and user settings.	Corrupted or missing results after crashes or restarts.
A6	No personally sensitive data beyond username and optional email is stored and there are no financial transactions.	Reduces compliance and breach impact.	If sensitive data is later added, privacy risk and legal exposure increase.
A7	Client devices provide basic accessibility aids such as keyboard, focus navigation, and screen reader support.	Meets usability and accessibility requirements.	Some users cannot operate the game which creates rule misunderstanding hazards.

5 Failure Mode and Effect Analysis

Table 3: Failure Mode and Effect Analysis (FMEA)

Component	Failure Modes	Effects of Failure	Causes of Failure	Recommended Action
Game Engine	Incorrect Dozenal validation (e.g., misinterpreting sum = 12).	Invalid moves; breaks gameplay logic and educational purpose.	Logic bug; base–10/12 conversion error.	(a) Add unit tests for all Dozenal comparisons; (b) property-based tests for edge cases.
UI Layer	Hints/feedback not displayed during gameplay.	Player confusion; reduced educational value.	Rendering delay; missing event callback.	(a) Async UI event queue with logging; (b) fallback text-only hint when animation fails.
Login Manager	Player unable to log in or session lost.	Loss of access to saved data or multiplayer mode.	Session timeout; invalid credential cache.	(a) Guest-mode fallback; (b) local session cache; retry with exponential backoff.
Data Manager	Save corruption during file write.	Loss of player profile or progress.	Non-atomic save; unexpected shutdown.	(a) Atomic writes + checksum on load; (b) periodic backups and restore flow.
Score Calculator	Wrong Dozenal–decimal conversion or totals.	Inconsistent scores; undermines trust.	Arithmetic bug; type overflow.	(a) Secondary calculator cross-check; (b) property-based tests on random inputs.
Network (future)	Desynchronized multiplayer state.	Different states across clients.	Packet loss; high latency.	(a) State-hash verification; (b) timeout-based resync; authoritative host.
Educational Feedback	Incorrect/misleading Dozenal explanation.	Players learn wrong concepts.	Faulty rule mapping; stale content.	(a) Peer-reviewed hint knowledge base; (b) versioning and CI checks.
Security Layer	Unencrypted data at rest/in transit.	Privacy breach; data leak.	Missing encryption; insecure local storage.	(a) AES-256 at rest; (b) HTTPS/TLS in transit; (c) secrets in OS keystore.

6 Safety and Security Requirements

This section defines the safety and security requirements derived from the identified hazards in the FMEA. These requirements are intended to prevent or mitigate the effects of hazards related to gameplay integrity, data protection, and user learning outcomes.

6.1 Safety Requirements

- **SR-1: Dozenal Arithmetic Validation**
  The system shall correctly validate card combinations and arithmetic operations in base-12 to prevent incorrect gameplay outcomes or user confusion.

- **SR-2: User Interface Feedback Consistency**
  The system shall provide real-time and clear visual or textual feedback

after every action to prevent user uncertainty and ensure the learning goal of base comprehension is maintained.

- **SR-3: Reliable Data Persistence**
  All player progress and results shall be stored atomically to prevent data corruption, including crash recovery and session reconnection mechanisms.
- **SR-4: Accurate Scoring and Conversion Algorithms**
  The system shall verify numeric conversions between bases through redundant validation logic or automated testing to avoid incorrect scoring or educational misinformation.
- **SR-5: Controlled Session Timeout and Recovery**
  If a session disconnect occurs, the server shall maintain game state and allow the player to resume without desynchronization.
- **SR-6: Accessibility Safeguards**
  The interface shall meet WCAG accessibility requirements (keyboard navigation, readable colors, screen reader compatibility) to prevent user exclusion and reduce operational hazards.

6.2 Security Requirements

- **SR-7: Encrypted Data Transmission**
  All client-server communication shall use TLS 1.2 or higher to protect against eavesdropping and session hijacking.
- **SR-8: Secure Data Storage**
  All user-related data stored on the server shall be encrypted using AES-256 and stored without plaintext credentials.
- **SR-9: Authentication and Session Protection**
  Session tokens shall expire after a defined idle period, and invalid sessions shall not be reused to avoid unauthorized access.
- **SR-10: Input Validation and Sanitization**
  The system shall validate all client input at the server level to prevent malicious commands or injection attacks.
- **SR-11: Versioned Educational Content**
  All explanatory Dozenal rules and hints shall be version-controlled and verified for correctness before deployment to prevent misinformation hazards.

7 Roadmap

The following roadmap identifies which safety and security requirements will be implemented within the capstone timeline and which will be deferred for future development.

7.1 Implemented in Capstone (Crazy10s v1.0)

- **SR-1** — Dozenal Arithmetic Validation
- **SR-2** — User Interface Feedback Consistency
- **SR-3** — Reliable Data Persistence
- **SR-4** — Accurate Scoring and Conversion Algorithms
- **SR-7** — Encrypted Data Transmission
- **SR-10** — Input Validation and Sanitization

7.2 Deferred for Future Release (Crazy10s v1.1 and beyond)

- **SR-5** — Controlled Session Timeout and Recovery (v1.1)
- **SR-6** — Accessibility Safeguards (v1.1)
- **SR-8** — Secure Data Storage (v1.1)
- **SR-9** — Authentication and Session Protection (v1.2)
- **SR-11** — Versioned Educational Content (v1.2)

These later milestones will focus on improved reliability, broader user inclusion, and content safety, building on the foundation of a functioning, secure educational game platform.

Appendix — 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? The team effectively applied a structured hazard analysis approach based on the FMEA framework presented in the lecture. Delegation of parts in this deliverable enabled sequential identification of software-specific hazards. Since each member did everything by order, writing this Hazard Analysis was efficient and straightforward.

2. What pain points did you experience during this deliverable, and how did you resolve them? The primary challenge was time, as our SRS document was so demanding, and took the majority of our time and focus in this deliverable. Another pain-point/challenge was lack of preparation for building the Hazard Analysis. Unfortunately, no team members went to the HA lecture, and we also did not review it together as a team before delegating.
As far as the Hazard Analysis itself, the most difficult part was distinguishing between functional errors/hazards and genuine hazards which actually impact the players of our game. It was also difficult to quantify how certain risks (like UI confusion) could degrade educational outcomes. The team resolved these by referencing real examples from other projects in the capstone repository, communicating internally, and restructuring the FMEA table to better align with user-centric risks rather than only functional risks.

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? Before this deliverable, our team had already considered general risks such as scheduling conflicts, Git merge issues, and limited testing time near deadlines. During the Hazard Analysis, we identified new, more specific risks, including Dozenal validation logic errors, data corruption during saving, and inaccurate educational feedback. These emerged as we analyzed each system component in the FMEA table, which led us to think more deeply about technical failures and their impact on users.

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.