Hazard Analysis
Software Engineering

Team #1, Sanskrit Ciphers
Omar El Aref
Dylan Garner
Muhammad Umar Khan
Aswin Kuganesan
Yousef Shahin

Table 1: Revision History

Date	**Developer(s)**	**Change**
October 10th 2025	Aswin Kuganesan	Added Critical Assumptions and Failure Mode and Effect Analysis table
October 10th 2025	Aswin Kuganesan	Added Reflection
October 10th 2025	Dylan Garner	Added initial hazard analysis
October 10th 2025	Omar El Aref	Added System Boundaries and Components
October 10th 2025	Umar Khan	Added safety and security requirements
October 10th 2025	Yousef Shahin	Created the roadmap

Contents

List of Tables

1 Introduction

1.1 Definition of Hazard

Following the definition from hazard analysis literature, a **hazard** is defined as a property or condition in the Sanskrit Manuscript Fragment Reconstruction Platform system together with a condition in the environment that has the potential to cause harm or damage (loss).

1.1.1 System Conditions

System conditions are properties or states of the software that create vulnerability or risk. Examples include:

- **Vulnerable database configuration**: Database exposed to unauthorized access attempts
- **Inaccurate Artificial Intelligence (AI) models**: Script classification model with low accuracy (¡50%)
- **Insufficient input validation**: System accepts malformed or malicious data
- **Poor error handling**: System crashes or enters unstable state on unexpected input
- **Inadequate session management**: Sessions persist beyond intended duration or lack proper isolation
- **Resource allocation issues**: Memory leaks, unbounded queries, or inefficient algorithms
- **Missing accessibility features**: Interface elements without keyboard navigation or screen reader support

1.1.2 Environmental Conditions

Environmental conditions are external circumstances or user behaviors that, when combined with system conditions, can cause harm. Examples include:

- **User attempts unauthorized data access**: User tries to download entire database or access restricted content
- **User trusts inaccurate AI predictions**: Scholar bases research conclusions on low-confidence model outputs
- **Network interruptions during critical operations**: Connection lost while saving research data
- **Concurrent user access to shared resources**: Multiple researchers editing same fragment simultaneously

- **Users with varying technical expertise**: Non-technical users attempting complex system operations
- **Users with accessibility needs**: Users requiring assistive technologies to interact with system

1.2 Project Context and Unique Hazard Considerations

The Sanskrit Manuscript Fragment Reconstruction Platform operates within a distinctive environment that creates unique user experience hazards not typically addressed in conventional software projects:

1.2.1 Diverse User Base with Varying Technical Skills

The system serves users ranging from graduate students learning paleography to expert scholars to archival staff, each with different technical comfort levels and research needs. Poor interface design or insensitive feature implementation could exclude or frustrate significant user groups, leading to abandonment of the tool or ineffective research workflows.

1.2.2 Research Workflow Integration Challenges

Scholars have established research methodologies and workspace preferences developed over years of practice. Software that disrupts familiar workflows, lacks intuitive navigation, or forces users to adapt to rigid system constraints can cause significant frustration and reduce research productivity. Poor performance or unreliable functionality breaks concentration and research flow.

1.2.3 Data Privacy and Confidentiality Concerns

Researchers work with sensitive data including unpublished discoveries, institutional collaborations, and potentially restricted manuscript materials. Unintended data exposure through system vulnerabilities, poor session management, or inadequate access controls could compromise ongoing research, violate institutional agreements, or expose confidential scholarly work to competitors.

2 Scope and Purpose of Hazard Analysis

2.1 Purpose

This hazard analysis identifies and evaluates potential risks associated with the Sanskrit Manuscript Fragment Reconstruction Platform to ensure safe, responsible deployment in academic research environments. The analysis focuses on protecting cultural heritage materials, maintaining scholarly integrity, and preventing negative impacts on Buddhist Studies research. Through systematic

verification testing and validation processes, this analysis helps identify potential system failures and ensures that main functionalities operate safely and reliably before deployment.

2.2 Potential Losses

Table 2 summarizes the potential losses that could occur from identified hazards in the system. These losses span data security, user experience, research productivity, and system reliability concerns.

Table 2: Potential Losses from System Hazards

Loss Category	Specific Losses
Data Security	• Exposure of confidential research data • Unauthorized access to manuscript database • Unintended data sharing between users • Violation of institutional data agreements • Loss of competitive research advantage
Research Productivity	• Lost research time due to crashes or slow performance • Workflow disruption from poor interface design • Research momentum loss requiring work restart • Inefficient task completion from confusing navigation
Data Integrity	• Data corruption compromising months of work • Inadvertent modification of research data • Loss of unsaved work due to system failures • Incorrect fragment matches from inaccurate AI
User Experience	• System unresponsiveness or freezing • Frequent errors reducing user confidence • Frustration from overly complex interfaces • Inconsistent behavior across browsers/devices
Accessibility & Inclusion	• Exclusion of users with disabilities • Frustration for non-technical users • Reduced adoption by diverse user groups

3 System Boundaries and Components

This hazard analysis treats the system as four major component groups that interact to support scholarly reconstruction of manuscript fragments. (For full details, refer to the SRS (Ciphers, 2025), Section S.1.)

Major Component Groups

- **Front-end**: Scholar-facing web UI for secure login, batch image uploads, an interactive canvas (arrange/rotate/zoom fragments), match discovery display, and session/annotation saving. Quick list of smaller components:
  - User Authentication Interface
  - Fragment Upload Interface
  - Interactive Canvas Module
  - Match Discovery Module
  - Progress Management Module
- **Backend**: API gateway, authentication/authorization, image preprocessing pipeline (normalization/orientation/format), database access layer, and a match-orchestration service coordinating analysis results. Concise breakdown of sub-components:
  - API Gateway
  - Authentication Service
  - Image Processing Service
  - Database Access Layer
  - Match Orchestration Service
- **Data Storage**: Fragment image store (originals and derivatives with metadata), user accounts/permissions, and project records (arrangements, match history, confidence scores, session snapshots). Brief overview of minor elements:
  - Fragment Image Database
  - User Database
  - Project Database
- **AI/ML**: Services/models for edge/damage matching, handwriting/script classification, OCR text extraction (with confidence), and content similarity/embedding comparisons. Summary of key smaller parts:
  - Edge Pattern Matching Model
  - Handwriting Style Classifier
  - Damage Pattern Recognition Model
  - Text Extraction Model
  - Content Similarity Model

Component Boundaries (Exclusions)

- **UI** presents results and collects inputs; it does not run ML models or make authoritative scholarly decisions.
- **Preprocessing** only standardizes images; it does not judge correctness of reconstructions.
- **Matching** produces scored suggestions; it does not auto-merge/link records without explicit user confirmation.
- **OCR/Transcription** is machine-generated with confidence indicators; final text requires human review before being marked confirmed.
- **Datastores** are system-of-record for this application only; they do not write back to external catalogues/corpora.

See SRS Section S.1 for the detailed component descriptions and interfaces. Note that here we have separated the backend and data storage to make it a little clearer to visualize the components but data storage will be treated as backend for the project.

4 Critical Assumptions

- Input images are provided in a usable format for the system and are not corrupted when uploaded
- Resolution of fragment images is sufficient for meaningful feature extraction using edges and damage patterns
- Users of the system are expected to understand the basic functionality of the system and be knowledgeable in religious studies
- External APIs and libraries used (image processing/segmentation) will be supported throughout the entire project lifetime.
- All images uploaded and stored in the database are images of ancient fragments that the user has been permitted to use

5 Failure Mode and Effect Analysis

Table 3: Failure Mode and Effects Analysis (FMEA) for Manuscript Recreation System

Design Function	Failure Modes	Effects of Failure	Causes of Failure	Recommended Action	SR
Manuscript Recreation	Image upload fails or corrupted file is accepted	Missing or unusable fragment in display	Invalid file format, incomplete upload, or corrupted image data	Validate file format during upload and provide error messages to users	SR.1
	Misalignment tools produce incorrect rotation/zoom	User arranges fragments inaccurately, leading to wrong conclusions	Calibration error or algorithmic inaccuracy in transformation logic	Allow users to undo and redo previous action	SR.2
	Unauthorized user gains access	Data breach or unauthorized edits to manuscripts	Weak password, unencrypted session, or bypassed authentication	Enforce strong passwords and 2 factor authentication	SR.3
	Data corruption or loss	User progress and annotations are lost	Database write failure or transaction interruption	Perform daily backups, implement transaction rollback and recovery mechanisms	SR.4
	False positives in edge similarity	Incorrect fragment matches suggested to user	Overfitting or poor training data distribution	Require human validation, display probability percentages and match explanations	SR.6
	User loses connection during upload or save	Session progress lost	Network interruption or unstable client connection	Autosave state locally, synchronize data once the connection is restored	SR.7

6 Safety and Security Requirements

The following safety and security requirements are derived from the identified hazards and their recommended mitigations. Each requirement corresponds to one or more failure modes discussed in the previous section. The intent is to ensure that failures in individual services or user actions do not compromise data integrity, security, or scholarly validity.

- **File Integrity and Upload Validation** *(Priority: High)*

  - The system shall validate all uploaded files for acceptable format (JPG, PNG) and integrity before ingestion.
  - The system shall reject corrupted or incomplete files and provide descriptive feedback to the user.
  - The system shall maintain a checksum for each uploaded image to prevent future corruption or mismatch.

  Rationale: Prevents data corruption and ensures scholarly work is based on intact manuscript images. Critical for maintaining research validity and preventing irreversible data loss.

- **User Interaction Safety** *(Priority: High)*
  - The system shall include undo/redo functionality for all manual alignment or manipulation tools on the interactive canvas.
  - The system shall display alignment guides or grid overlays to minimize misplacement of fragments.
  - The system shall maintain a revision history of all user adjustments to allow recovery from accidental operations.

  Rationale: Protects scholars from losing hours of painstaking manual work due to accidental clicks or misoperations. Essential for user confidence and adoption of the system.

- **Authentication and Access Control** *(Priority: High)*
  - The system shall enforce strong password policies, require secure session tokens, and store credentials in an encrypted manner.
  - The system shall support optional two-factor authentication (2FA) for privileged or administrative accounts.
  - The system shall log all authentication events (login, failed attempt, password reset) and perform periodic security audits.
  - The system shall automatically lock an account after multiple failed authentication attempts within a short time window.

  Rationale: Essential for protecting scholarly research data and preventing unauthorized access to valuable manuscript collections. Required for compliance with institutional security policies.

- **Database Integrity and Recovery** *(Priority: High)*
  - The system shall perform automated daily backups of all databases (fragment metadata, annotations, user projects).
  - The system shall implement transaction rollback and atomic commits to prevent partial data writes.

  - The system shall provide administrators with a verified recovery mechanism for restoring corrupted or lost data.
  - The system shall maintain referential integrity across related tables to prevent inconsistent state.

  Rationale: Critical for preserving years of scholarly research work and manuscript annotations. Prevents catastrophic data loss that could set back research projects by months or years.

- **Service Reliability and Fault Tolerance** *(Priority: Medium)*
  - The system shall detect service timeouts or crashes (e.g., ML inference, orchestration tasks) and automatically retry with exponential backoff.
  - The system shall provide user feedback when a service is unavailable and allow fallback to manual workflows.
  - The system shall implement failover handling for long-running processes to avoid complete job loss.
  - The system shall log all failures and exception traces for later analysis.

  Rationale: Ensures system remains usable even when individual components fail. Prevents user frustration and maintains productivity during system maintenance or component failures.

- **Model Transparency and Scholarly Control** *(Priority: High)*
  - The system shall display confidence levels and similarity metrics for all model-generated suggestions (matching, classification, transcription).
  - The system shall require explicit human confirmation before accepting any automated fragment match.
  - The system shall maintain version control for model outputs, ensuring that updated models do not overwrite prior validated results.

  Rationale: Essential for scholarly integrity and peer review. Ensures researchers can evaluate AI suggestions critically and maintain control over their research conclusions. Prevents over-reliance on potentially flawed automated decisions.

- **Network and Connectivity Safety** *(Priority: Medium)*
  - The system shall autosave the current workspace and session data locally every 60 seconds.
  - The system shall sync unsaved data automatically upon reconnection.
  - The system shall warn users of unsent uploads or unsynced work before closing the browser or session.

Rationale: Protects users from losing work due to network interruptions or browser crashes. Critical for maintaining productivity in academic environments with unreliable internet connectivity.

- **Security of Stored and Transmitted Data** *(Priority: High)*
  - The system shall use TLS (Transport Layer Security) for all client–server communication and encrypt sensitive data in storage.
  - The system shall sanitize and validate all user inputs to prevent injection or cross-site scripting (XSS) attacks.
  - The system shall segregate user data using role-based access to prevent unauthorized access or leakage.

  Rationale: Fundamental security requirement for any system handling academic research data. Protects against data breaches, unauthorized access, and ensures compliance with institutional and legal data protection requirements.

- **Auditability and Traceability** *(Priority: Low)*
  - The system shall maintain an audit log for all critical operations, including uploads, edits, deletions, and administrative actions.
  - The system shall associate every change with a user identity and timestamp.
  - The system shall provide administrators with a secure interface to review audit logs.

  Rationale: Important for troubleshooting, compliance, and understanding system usage patterns. However, can be implemented after core functionality is established and is primarily useful for administrative oversight.

7 Roadmap

7.1 Requirements to be Implemented During Capstone Timeline

The following safety and security requirements will be implemented as part of the capstone project, organized by project phase:

7.1.1 Phase 1: Proof of Concept (Term 1, Weeks 1–6)

- **File Integrity and Upload Validation (Partial):**
  - Validate uploaded files for acceptable formats (JPG, PNG)
  - Reject corrupted or incomplete files with basic error messages
  - This addresses the immediate need to handle fragment images reliably during POC demonstrations

- **User Interaction Safety (Basic):**
  - Implement basic undo functionality for manual fragment manipulation
  - This is critical for POC usability testing with non-technical scholars
- **Model Transparency (Core):**
  - Display confidence levels for model-generated fragment matching suggestions
  - Ensure human confirmation before accepting automated matches
  - This directly supports the scholarly validation workflow required for POC success metrics

7.1.2 Phase 2: Revision 0 (Term 1, Weeks 7–13)

- **File Integrity and Upload Validation (Complete):**
  - Implement checksum verification for uploaded images
  - Add comprehensive file validation and descriptive error feedback
- **User Interaction Safety (Enhanced):**
  - Add redo functionality to complement undo operations
  - Implement alignment guides or grid overlays for the interactive canvas
  - Maintain revision history for user adjustments (limited to current session)
- **Service Reliability and Fault Tolerance (Basic):**
  - Detect ML inference timeouts and provide user feedback
  - Implement basic error logging for failures and exceptions
  - Allow fallback to manual workflows when automated services fail
  - These are essential given the OCR performance risks identified in the development plan
- **Network and Connectivity Safety (Basic):**
  - Implement autosave functionality for workspace data every 60 seconds
  - Warn users of unsaved work before closing browser session
  - This addresses usability concerns for scholars working with large fragment collections
- **Security of Stored and Transmitted Data (Basic):**

  - Use HTTPS/TLS for all client–server communication
  - Implement basic input sanitization to prevent injection attacks
  - This is necessary given the confidential nature of manuscript fragments

7.1.3 Phase 3: Revision 1 and Final (Term 2, Weeks 1–13)

- **Authentication and Access Control (Basic):**
  - Implement secure user authentication with strong password policies
  - Store credentials in encrypted form
  - Log authentication events (login attempts, password resets)
  - Account lockout after multiple failed login attempts
  - Note: 2FA will be deferred to future work given time constraints
- **Database Integrity and Recovery (Basic):**
  - Implement automated daily backups of fragment metadata and user projects
  - Use atomic commits and transaction rollback for database operations
  - Maintain referential integrity across related tables
  - Note: Full administrative recovery mechanisms will be simplified for capstone scope
- **Service Reliability and Fault Tolerance (Enhanced):**
  - Add automatic retry with exponential backoff for failed ML inference requests
  - Implement failover handling for long-running fragment analysis processes
- **Network and Connectivity Safety (Complete):**
  - Implement automatic sync of unsaved data upon reconnection
- **Model Transparency (Enhanced):**
  - Add version control for model outputs to preserve validated results
  - Display similarity metrics alongside confidence levels for all suggestions
- **Auditability and Traceability (Basic):**
  - Maintain audit logs for critical operations (uploads, edits, deletions)
  - Associate changes with user identity and timestamp
  - Note: Administrative audit log interface will be simplified for capstone scope

7.2 Requirements Deferred to Future Work

The following requirements are important for a production-ready system but are deferred beyond the capstone timeline due to scope constraints and project priorities:

- **Authentication and Access Control:**
  - Support optional two-factor authentication (2FA) for privileged or administrative accounts
  - Perform periodic security audits
- **Database Integrity and Recovery:**
  - Provide administrators with a verified recovery mechanism for restoring corrupted or lost data
- **Security of Stored and Transmitted Data:**
  - Encrypt sensitive data in storage
  - Segregate user data using role-based access to prevent unauthorized access or leakage
- **Auditability and Traceability:**
  - Provide administrators with a secure interface to review audit logs

7.3 Rationale for Prioritization

The implementation roadmap prioritizes requirements that:

1. **Support core project goals:** Requirements directly related to OCR accuracy, fragment matching, and usability for Buddhist Studies scholars take priority (e.g., model transparency, file validation, user interaction safety).
2. **Mitigate identified risks:** Given the POC demonstration plan's focus on OCR performance and lack of ground truth data, we prioritize service reliability and model transparency requirements.
3. **Enable iterative testing:** Basic safety features like undo/redo and autosave are essential for the usability testing required to meet the 15-minute training goal.
4. **Protect confidential data:** Since manuscript fragments are confidential (Section 2 of Development Plan), basic security measures (HTTPS, input sanitization, authentication) are implemented before final delivery.

5. **Fit within capstone timeline:** Advanced features requiring significant infrastructure work (2FA, distributed orchestration, comprehensive auditing) are deferred to ensure the team can deliver a functional system within two terms while maintaining approximately 10 hours per week per team member.

References

Team Sanskrit Ciphers. Software requirements specification (srs). https://github.com/DylanG5/sanskrit-ciphers-requirements/blob/9938018b682709551d0a6248a0a9f6e5794112ee/index.pdf, 2025. Specifies functional and non-functional requirements for the system.

Appendix — Reflection

1. While writing this deliverable, the safety requirements were easy to create because the components were well defined. This allowed for us to determine the most likely hazards to occur based on the components and their functions. Then, we were able to create safety requirements to prevent these hazards from occurring. We also were able to determine the critical assumptions easily because many of them were assumed during the planning of the project and are assumptions that we have discussed in the past.

2. During this deliverable, we found that the FMEA table was difficult to complete because we had to think of multiple different ways the system could fail and the steps we would take to prevent these failures. This required us to think more critically about the system rather than surface level thinking. Creating the roadmap also required us to consider future work and how we would implement the safety requirements. This required us discussing the feasibility of implementing several safety features within the project timeline.

3. The more serious risks were considered first during project planning. This includes risks such as database corruption and image upload failure. These risks were considered first because they would have the most significant impact on the user and the system. Our discussions with our supervisor about requirements led to us thinking about potential risks that we had not considered. This included the risks considered during this deliverable, specifically connection loss and false positives in the algorithm. These risks were considered because although they do not have as big of an impact as the more serious risks, they could still cause the user to experience dissatisfaction with the system.

4. Many software applications store user data and rely on this data for their application to personalize the experience for the user. This means that a large amount of user data is stored in databases which need to be protected to prevent data breaches. Bad actors may try to gain access to this data for malicious purposes. This is why security is a major concern for software applications. Furthermore, software applications store important data that users do not want to lose. Data loss can lead to users losing progress on their work which can have devastating consequences. This is why data integrity needs to be considered when designing a software application.