Hazard Analysis
Software Engineering

Team 21, Alkalytics
Sumanya Gulati
Kate Min
Jennifer Ye
Jason Tran

Table 1: Revision History

Date	Developer(s)	Change
25 October 2024	Kate Min	Add sections 2 and 7
25 October 2024	Jason Tran	Add sections 5 and 6
25 October 2024	Jennifer Ye	Add sections 1 and 4
26 October 2024	Sumanya Gulati	Add sections 3 and Appendix - Reflection
25 November 2024	Jennifer Ye	Updated section 4 to adhere to 107, 108, 109.
29 December 2024	Kate Min	Updated section 2 to adhere to 157.
25 March 2025	Jason Tran	Updated section 5 to adhere to 111, 158.
1 April 2025	Kate Min	Updated section 6 to adhere to 156.

Contents

1 Introduction

A hazard is any property or condition that has the potential to cause harm. This document serves as a hazard analysis for the application revolving around the capstone project "Alkalytics". This project aims to aids in the data management and analysis of an ocean alkalinity enhancement experiment. This document identifies the components of the system, and then the possible software hazards in these components, as well as ways to mitigate the risks they impose.

2 Scope and Purpose of Hazard Analysis

The purpose of the hazard analysis is to identify potential hazards and its causes, assess the effects, and set mitigation strategies to eliminate or lessen the risk of the hazard. This analysis aims to address all relevant hazards within the system boundaries and components defined in Section 3, ensuring the application's functionality, reliability, security, and performance. The hazard analysis will focus on identifying hazards that may lead to the following:

- Data loss: Unintentional loss, deletion, or corruption of data.
- Data integrity: Incomplete, inaccurate, and/or incorrect data, which can lead to computational errors.
- System availability: Users unable to access the system due to factors such as server downtime or system overload.
- Security: Unauthorized access to the data or user information due to insufficient authentication measures.

In addition to these areas, the analysis will also consider hazards that may impact overall user experience, specifically addressing hazards related to usability, accessibility, and system interaction. The analysis will not consider hazards that may be associated with hardware failures or other external dependencies beyond the control of the application unless they directly impact the operation of the application.

3 System Boundaries and Components

The system boundaries are used to define what is within the scope of the hazard analysis, in essence, where the system interacts with external systems, users and hardware. Some key boundaries for Alkalytics include:

- **User Interaction Boundaries**:
    1. Interactions happen through the frontend interface, where users authenticate, input data, and retrieve visualizations and analytics.
    2. Limit interactions in terms of user access levels, as only authenticated and authorized users are allowed to view or modify certain data.
- **Data Boundaries**:

1. Alkalytics handles data import (CSV files), processing, analysis, and export, requiring clear boundaries on data validity and format conformity.
2. The system is also bounded by its data storage capabilities, including access controls, data integrity safeguards, and secure handling of sensitive information.

- **Network and External Connectivity Boundaries**:

1. External connections, like internet-based server access and database queries, form a boundary where system security, speed, and availability must be managed.
2. Network performance, internet stability, and server downtime impact the accessibility and reliability of the system.

The system can be broken down into the following major components:

1. **Authentication System**: Ensures that only authorized users can access the Alkalytics platform and limits actions based on user permissions.
2. **Comma-Separated Values (CSV) Data Migration Module**: Handles the import of data from CSV files into the database, including checks for data format, completeness, and correctness.
3. **Data Visualization Module**: Generates graphs and visual representations of data, requiring accuracy in rendering and efficient performance.
4. **Query Functionality**: Processes user queries to retrieve data from the database, ensuring that correct data is returned promptly.
5. **Data Export Module**: Allows users to export data in CSV for external use, requiring accuracy and completeness.
6. **Backend Database**: Stores and retrieves all user and application data, requiring reliable connection management, data integrity, and sufficient storage.
7. **Frontend Interface**: Provides users with access to Alkalytics' functionalities through a web-based UI, handling data input validation and responsiveness.
8. **Error Tracking System**: Logs and categorizes system errors for troubleshooting and performance monitoring.
9. **Machine Learning Analysis Module**: Processes data to provide predictive insights or data analyses, requiring well-trained models and accurate data handling.

4 Critical Assumptions

The following are assumptions made about the software of the system.

- The user of this application is not intentionally trying to misuse it
  - This assumption mitigates the risk of someone intentionally damaging the system

- Local server infrastructure will always be available, and will not suddenly go down and compromise the system
  - This assumption mitigates the risk of local server connections interrupting the system as there can be immidate resolutions and issues will only affect one user uniquely
- Users using this application understand how to use the application, whether through documentation or a tutorial
  - This assumptions ensures that user errors caused by lack of knowledge do not occur
- The system is regularly maintained for security and bug fixes
  - This assumption prevents any threats to the system due to poor maintenance

5 Failure Mode and Effect Analysis

Table 2 outlines the Failure Modes and Effects Analysis (FMEA) for each component of the Alkalytics system.

Table 2: Failure Mode and Effect Analysis.

Component	Failure Modes	Effects of Failure	Causes of Failure	Recommended Action	SR	Ref
Authentication	Unauthorized user access	Data breach, loss of data integrity	Weak security protocols, poor session management	Strengthen authentication mechanisms, enforce MFA	SR-1, SR-2	H1-1
	User unable to log in	Loss of productivity for affected users, potential lockout during critical operations	System downtime, credential errors, incorrect password policy	Ensure high system uptime, implement account recovery options	FR-14, SR-1, SR-4	H1-2
CSV Data Migration	Data not uploaded to the database	Loss of data availability, missing records in reports	Incorrect file format, server issue, unsupported encoding	Validate file format, encoding, and ensure server uptime	FR-3, FR-4, SR-9	H2-1
	Data partially uploaded	Incomplete data leads to incorrect reports, potential duplication on retry	Timeout during upload, corrupted file, network disconnect	Implement transaction rollback for partial uploads, retry logic for failures	FR-3, FR-2	H2-2
	Duplicate data entries	Conflicting results in analysis, redundant storage usage	No validation for duplicates, improper indexing	Add duplicate detection and rejection logic before insertion	SR-5, FR-4, SR-7	H2-3
Data Visualization	Incorrect graph rendering	Misleading insights, incorrect decision-making	Inaccurate parameter selection, incorrect data mapping	Improve input validation, enable real-time data verification	FR-8, FR-9	H3-1
	Slow graph rendering	Delayed analysis, poor user experience	Large dataset, inefficient plotting algorithm, memory leaks	Optimize graph rendering with efficient algorithms	PR-5, PR-2	H3-2
Query Functionality	Data not returned or delayed	Users unable to retrieve necessary data, system bottlenecks	Database connection issues, poorly optimized queries	Optimize indexing, caching, and error handling in query execution	FR-5, FR-6, PR-2	H4-1
	Incorrect data returned	Misleading results, incorrect decisions	Misconfigured query logic, incorrect joins or filters	Validate query logic, implement automated query testing	PR-6, FR-5	H4-2
	Query results outdated	Users work with old data, impacting real-time decisions	Lack of data refresh, stale cache	Implement scheduled data refresh and cache invalidation	PR-14, FR-6	H4-3
Data Export	CSV export generates corrupted files	Data cannot be processed by external systems	Incorrect formatting logic, encoding issues	Implement robust export validation, include file integrity checks	FR-15, PR-6, SR-3	H5-1
	Export missing data	Incomplete reports, misleading analysis	Timeout during export, data truncation, incorrect filtering	Ensure export process handles large data, implement chunked exports	FR-15, PR-8	H5-2

Component	Failure Modes	Effects of Failure	Causes of Failure	Recommended Action	SR	Ref
	Session timeout too short	Frequent logouts, user frustration	Misconfigured session settings, premature session expiry	Adjust session timeout dynamically based on user activity	FR-14, SR-16, SR-4	H5-3
Backend Database	Database connection lost	System downtime, inability to retrieve or update data	Network failure, server crash	Implement database redundancy, failover mechanisms	PR-9, PR-10	H6-1
	Data corruption during storage	Loss of data integrity, incorrect analysis	Faulty write operations, hardware failure	Implement checksums, enable automated backups	SR-6, SR-7	H6-2
	Insufficient storage space	System crashes, inability to store new data	Poor storage management, lack of monitoring	Increase storage capacity, implement monitoring alerts	PR-12, PR-13	H6-3
Frontend Interface	UI unresponsive	Users unable to complete tasks, frustration	JavaScript errors, resource overload	Debug UI, optimize resource management	FR-7, PR-3	H7-1
	Elements not displayed correctly	Confusion, incorrect user actions	Browser compatibility issues, incorrect CSS/HTML structure	Test compatibility across different browsers	LFR-1, OER-3	H7-2
	Data input fields allow invalid entries	Corrupts stored data, leading to incorrect operations	No input validation, improper form handling	Enforce strict input validation rules	FR-1, FR-4, SR-7	H7-3
Performance	Slow page load times	User frustration, reduced efficiency	Unoptimized frontend/backend code, large assets	Optimize assets, use lazy loading, improve database indexing	PR-3, PR-5	H8-1
	High CPU/memory usage	System instability, crashes	Inefficient data processing, memory leaks	Implement garbage collection, optimize memory usage	PR-4, PR-2	H8-2
Error Tracking	Errors not logged	Hard to diagnose issues, delayed troubleshooting	Lack of proper error handling in code	Implement centralized error logging	FR-12, FR-13	H9-1
	Logged errors not displayed to user	Users unaware of system issues, lack of feedback	Incomplete UI error handling	Display meaningful error messages to users	MSR-5, PR-9	H9-2
	Errors logged but not categorized	Troubleshooting complexity, inefficient debugging	Poor error categorization strategy	Implement structured error categorization	MSR-5, FR-12	H9-3
Machine Learning Analysis	Incorrect predictions	Misleading trends, faulty decisions	Poor training data, overfitting	Improve training data quality, implement cross-validation	FR-11	H10-1
	Model training incomplete	Model unable to make predictions	Insufficient training data, process failure	Ensure sufficient, clean training data, handle interruptions	FR-11, PR-8	H10-2
	Overfitting of training data	Model provides unreliable results	Training data too specific, lack of generalization	Implement regularization, cross-validation	FR-11	H10-3

6 Safety and Security Requirements

New safety and security requirements have been discovered and will be integrated within the SRS document. Note that the entire requirement codes have been changed with the new additions having expanded information, and the previous requirements displaying new codes.

6.1 Access Requirements

SR-1. Access to the application must be restricted to authorized personnel, with an authentication mechanism.

SR-2. Only authenticated users should have the ability to query or modify the data, and each user's access must be limited to their capabilities within the application.

SR-3. The application must restrict sensitive operations (e.g., data export) to authorized personnel only.

- *Rationale:* Prevents unauthorized users from exporting or sharing sensitive data, protecting data integrity.
- *Fit Criterion:* Only authorized users must be able to perform sensitive operations like data export.
- *Traceability:* FR-15, SR-2.

SR-4. The application must enforce session timeout and automatic logouts after a period of inactivity.

- *Rationale:* Protects the application from unauthorized access if users leave their session unattended.
- *Fit Criterion:* Sessions must time out and log users off automatically after a specified inactivity period.
- *Traceability:* FR-14, SR-1, SR-4.

6.2 Integrity Requirements

SR-5. The application must validate data inputs to ensure they conform to expected formats and values before they are processed.

SR-6. The application must not modify the data unnecessarily through its transfer process.

SR-7. The application must ensure that any data processed or transferred is free from duplication or inconsistencies.

SR-8. The application must have safeguards in place to maintain the accuracy of the transferred data.

SR-9. The application must validate CSV data thoroughly before upload to prevent corrupted or incomplete data entries.

- *Rationale:* Ensures that only valid, complete, and accurate data enters the application to prevent faulty analysis.
- *Fit Criterion:* The application shall reject any data that does not meet the validation criteria.
- *Traceability:* FR-3, FR-4.

6.3 Privacy Requirements

SR-10. All personal information related to experimental participants or stakeholders, if applicable, must be anonymized and handled in accordance with relevant privacy laws and regulations.

SR-11. The application must restrict data sharing with external parties unless expressly authorized by stakeholders, and users must be fully informed about the privacy policies.

SR-12. The application must monitor database storage capacity and alert administrators when thresholds are reached to prevent application crashes.

- *Rationale:* Ensures the application continues operating smoothly by addressing storage limits proactively.
- *Fit Criterion:* The application shall send alerts when storage capacity exceeds 80% usage.
- *Traceability:* PR-12, PR-13.

6.4 Audit Requirements

SR-13. The application must maintain a comprehensive audit trail, logging all access and modification events, including timestamps and identities of users performing actions.

SR-14. Audit logs must be securely stored and accessible only by authorized personnel.

SR-15. The application must display real-time error logs to users to enhance troubleshooting when applicable.

- *Rationale:* Ensures users are informed about application issues and can take corrective action promptly.
- *Fit Criterion:* All errors must be logged and displayed clearly to users in real-time.
- *Traceability:* FR-12, MSR-5.

6.5 Immunity Requirements

SR-16. The application must have proactive measures to detect and mitigate suspicious activities, such as repeated unauthorized access attempts, ensuring the application remains secure at all times.

SR-17. Real-time monitoring and optimization of application resources must be implemented to avoid crashes due to resource overload.

- *Rationale:* Prevents application downtime by ensuring efficient use of CPU and memory.
- *Fit Criterion:* The application must manage memory and CPU usage dynamically to avoid overloads.
- *Traceability:* PR-4, PR-9.

7 Roadmap

Table 3 outlines a proposed roadmap of when each safety requirement will be implemented within the capstone timeline and justifications.

Stage	**Req. Category**	**Req. ID(s)**	**Rationale**
PoC Demo (Nov 11)	Access	N/A	The PoC plan will not consider user access features at this time.
	Integrity	SR-5, SR-6, SR-7, SR-8, SR-9	The database must adhere to these requirements for a successful PoC.
	Privacy Audit Immunity	N/A N/A N/A	Since the PoC plan will only have the database, these requirements are not applicable.
Rev0 Demo (Feb 3)	Access	SR-1	User authentication should be implemented during front-end development.
	Integrity	N/A	The crucial integrity requirements will have already been implemented by the PoC demo.
	Privacy	SR-10, SR-11	At this point there will be user access, thus these requirements must be implemented.
	Audit Immunity	N/A N/A	These requirements are not high-priority.
Rev1 Final Demo (Mar 24)	Access	SR-2, SR-3	User access must be extended to permissions and capabilities prior to release.
	Integrity	N/A	The crucial integrity requirements will have already been implemented by the PoC demo.
	Privacy	SR-12	This requirement is necessary for system availability and robustness to extend past capstone.

Stage	Req. Category	Req. ID(s)	Rationale
	Audit	SR-15	Client and users must be informed about system issues and be able to troubleshoot even without the original development team.
	Immunity	SR-17	This requirement is necessary for system availability and robustness to extend past capstone.
Future considerations	Access	SR-4	This requirement is out of scope for the project timeline.
	Integrity	SR-11	This requirement is out of scope for the project timeline.
	Privacy	N/A	All privacy requirements have been covered.
	Audit	SR-13, SR-14	These requirements would be nice-to-haves but are not essential for the project.
	Immunity	SR-16	This requirement is out of scope for the project timeline.

Table 3: Roadmap of the implementation of the safety and security requirements.

Appendix — Reflection

1. **What went well while writing this deliverable?**

   Writing a comprehensive Software Requirements Specification (SRS) right before this milestone helped tremendously because as a team, we had been nudged into considering multiple aspects of the project we had not thought of earlier. All this research and decision-making ensured that coming into this milestone, we had most of the relevant things figured out prior to writing this documentation.

2. **What pain points did you experience during this deliverable, and how did you resolve them?**

   One major pain point we experienced was the lack of time we could dedicate to this milestone as a team. Since this milestone was due the week after reading week when most of us had 2 or more than 2 midterms along with multiple other assignment deadlines during the same week, it was harder to coordinate times to meet up and get a lot of the work done a couple of days before the deadline, which is what we ideally like to do. As an instance, due to the time constraints, we did not have enough time to get our Pull Requests (PRs) reviewed and approved by all the team members before merging.

   Additionally, since most of the sections are a lot more interwoven as compared to prior milestones (which although connected, could be written somewhat independently), it was harder to divide the sections. This ties back to the aforementioned time-contraints and we believe that if we had more time to meet up as a team, the division of tasks would have been easier to manage. We resolved this by effectively communicating with each other over text instead.

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?**

   Our team had thought of a majority of the risks before this deliverable. The ones that we had not considered include unresponsiveness of the frontend, performance related failures and error tracking related hazards. These came about after we had identified all the system boundaries and when we focused on the frontend, performance as well as error tracking components in particular and tried to consider all the potential ways these component can fail.

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?**

   Beyond the risk of physical harm, software products may have the following types of risks:

   (a) **Security Risks**: These involve vulnerabilities that can be exploited, leading to unauthorized access, data breaches or malicious attacks. For projects that are data centric like ours, these risks are critical because they can lead to the loss of sensitive information.

(b) **Operational Risks**: These risks arise from failures in the software's functionality, reliability, or performance under real-world conditions. Operational risks are important to consider because any failure in the software's expected behavior can disrupt business operations, cause customer dissatisfaction, lead to unexpected maintenance costs, or more.