Hazard Analysis
Plutos

Team #10, Plutos
Payton Chan
Eric Chen
Fondson Lu
Jason Tan
Angela Wang

Table 1: Revision History

Date	**Developer(s)**	**Change**
10/23/2024	Angela	Initial draft
10/25/2024	Jason	Table for section 5
10/25/2024	Payton, Eric, Fondson, Jason, Angela	Final Rev 0
10/28/2024	Eric	Update section 3
01/29/2025	Payton	Revising sections 3 & 5
01/29/2025	Payton	Revising sections 3 & 5
04/01/2025	Angela	Rev0 feedback rewording
...	...	...

Contents

List of Tables

1 Introduction

A hazard in the context of this document is any property or condition that may lead to harm or damage to the Plutos system or its users. Potential losses due to these hazards may include loss of application functionality, performance, or accuracy, or breaches of user privacy or data. The following sections will identify hazards within the system and discuss the controls in place for their mitigation.

2 Scope and Purpose of Hazard Analysis

This document aims to provide a comprehensive hazard analysis of the Plutos system. It identifies hazards within the system, outlines measures to mitigate them, and specifies the safety and security requirements derived from this analysis. The analysis will follow the Failure Mode and Effect Analysis (FMEA) approach. The analysis aims to discover the potential failure modes within the system and develop a mitigation plan to reduce the risk of failure.

3 System Boundaries and Components

The system will be divided into the following components:

1. The Plutos application, which consists of:
   (a) **The database**: The database is where the user's receipts and profile data will be stored.
   - Hazards:
     i. SQL Injection - Attackers can manipulate SQL queries through user inputs, potentially accessing or altering the database.
   - Mitigation:
     i. Use parameterized queries or prepared statements to separate SQL code from data inputs, and validate and sanitize all user inputs.

   (b) **The backend server**: The backend server is responsible for handling and serving requests from the client. It will interact with all the other components listed here.
   - Hazards:
     i. Lack of Rate Limiting - Excessive requests from users or automated scripts can overload the app and lead to denial of service.
     ii. Dependency Vulnerabilities - Using third-party libraries and frameworks can introduce vulnerabilities if they are not regularly updated.
     iii. Inadequate Error Handling - Poor error handling can expose stack traces or sensitive information to users, aiding attackers.
     iv. Insecure API Endpoints - Unprotected or poorly secured API endpoints can expose sensitive data or allow unauthorized actions.
   - Mitigation:
     i. Implement rate limiting to control the number of requests a user can make in a given time frame and deploy web application firewalls (WAFs) to filter malicious traffic.

ii. Regularly review and update dependencies, and use tools to scan for known vulnerabilities in third-party packages.
iii. Implement generic error messages for users and log detailed errors securely for developers.
iv. Implement proper authentication and authorization mechanisms, and use HTTPS to secure API communications.

(c) **The frontend/user interface**: The frontend/user interface is responsible for displaying the appropriate views to the user and handling user interactions.

- Hazards:
  i. Authentication Vulnerabilities - Weak authentication mechanisms can allow unauthorized users to access sensitive features of the app.
  ii. Cross-Site Scripting (XSS) Attacks - Attackers may inject malicious scripts via images into the app, which can steal user data or compromise the app's functionality.
  iii. Poor Session Management - Users may remain logged in indefinitely, increasing the risk of unauthorized access on shared or public devices.
  iv. Client-Side Storage Vulnerabilities - Sensitive data stored in local storage or cookies can be easily accessed or manipulated by malicious scripts.
  v. Unprotected Routes - Sensitive pages may not have adequate access controls, allowing unauthorized users to access them.
- Mitigation:
  i. Implement an IAM service (OAuth 2.0) that handles authentication and authorization amongst login sessions.
  ii. Sanitize all user inputs and outputs to ensure that any HTML or JavaScript code is escaped or neutralized.
  iii. Implement session expiration policies and automatically log out inactive users and regularly rotate session tokens to minimize the risk of token theft.
  iv. Avoid storing sensitive information in client-side storage and use secure cookies for any necessary data.
  v. Implement role-based access control (RBAC) and ensure all routes are properly protected by authentication checks.

(d) **The machine learning (ML) model**: The ML model is responsible for parsing and categorizing items from a picture of an itemized receipt.

2. The user's mobile device and camera setup

3. Several links to external systems:

(a) Plutos will interact with various external APIs and services, which can impact its reliability and security.

- Hazards:
  i. Data exchange vulnerabilities with third-party services.
  ii. Dependency on external service availability, which may lead to system downtime.
  iii. Inadequate authentication mechanisms for external API calls.
- Mitigation:

i. Implement robust authentication and authorization protocols for all external API interactions.
ii. Regularly audit and monitor third-party services for security compliance and performance reliability.
iii. Establish fallback mechanisms to handle failures in external services, ensuring minimal disruption to the Plutos system.

4. Several links to external systems:

   (a) Plutos will interact with various external APIs and services, which can impact its reliability and security.

   - Hazards:
     i. Data exchange vulnerabilities with third-party services.
     ii. Dependency on external service availability, which may lead to system downtime.
     iii. Inadequate authentication mechanisms for external API calls.
   - Mitigation:
     i. Implement robust authentication and authorization protocols for all external API interactions.
     ii. Regularly audit and monitor third-party services for security compliance and performance reliability.
     iii. Establish fallback mechanisms to handle failures in external services, ensuring minimal disruption to the Plutos system.

5. Several links to external systems:

   (a) Plutos will interact with various external APIs and services, which can impact its reliability and security.

   - Hazards:
     i. Data exchange vulnerabilities with third-party services.
     ii. Dependency on external service availability, which may lead to system downtime.
     iii. Inadequate authentication mechanisms for external API calls.
   - Mitigation:
     i. Implement robust authentication and authorization protocols for all external API interactions.
     ii. Regularly audit and monitor third-party services for security compliance and performance reliability.
     iii. Establish fallback mechanisms to handle failures in external services, ensuring minimal disruption to the Plutos system.

4 Critical Assumptions

The project will be making the following critical assumptions:

1. The users will be using a mobile device running an up-to-date version of iOS or Android.

2. Users are not expected to repeatedly input invalid images into the system (i.e., images that do not contain a receipt). While it is anticipated that users may occasionally submit an invalid image, it is assumed to not be a significant concern.

3. The project assumes that users' cameras will function properly and produce images of adequate quality for analysis.

4. Issues such as low-quality images or invalid pictures may arise from the user's equipment (e.g., a malfunctioning camera) and are considered outside the project's scope.

5 Failure Mode and Effect Analysis

Table 2: Failure Mode and Effect Analysis Table

Design Function	Failure Modes	Effects of Failure	Causes of Failure	Recommended Action	SR	Ref
Authenticate user	Non-human accounts exist on the server	Increased traffic to the server, invalid inputs, performance deterioration	A bot account logs into the application	Implement an automated captcha verification system during account creation to prevent bot accounts.	FR3, SR1	H1-1
	User account information has been compromised due to unauthorized access	User can no longer access their account and their data might be compromised or lost	Unauthorized account access	Enforce strong user passwords, allow users to reset their password	FR6, SR7, SR12	H1-2
Accept image	Camera does not open	The user cannot input an image to be processed by the system.	(a) Application does not have access to user's camera (b) The application is bugged out	(a) Prompt the user to enable camera access or suggest selecting an image from their photo library instead. (b) Prompt the user to restart the application.	FR7, NFR13, SR2	H2-1
	Photo library doesn't open	Same as H2-1	Same as H2-1	Same as H2-1	FR8, NFR13, SR3	H2-2
Process receipt image	Image is not processable	Poor input into machine learning model; may cause incorrect results or cause model to run for a long time	(a) Poor quality receipt (e.g., paper is crumpled or wrinkled, light ink) (b) Poor photo quality (e.g., blurry, part of receipt cut out from frame, low lighting) (c) The image does not contain a receipt	(a) Incorporate an automated image quality assessment tool that provides real-time feedback and suggestions for improving photo quality before submission. (b) Same as H3-1a (c) Run image validation before passing the image through the ML model (e.g., check that there is a paper with text in the image)	FR16, NFR13, SR4, SR6	H3-1

Continued from previous page

Design Function	Failure Modes	Effects of Failure	Causes of Failure	Recommended Action	SR	Ref
	Optical character recognition (OCR) failing or incorrectly parsing text	Incorrectly parsed text, missing items, incorrect categorization or costs	(a) Receipt uses different font sizes or styles (b) Receipt quality causes some letters to be difficult to recognize (e.g., 0 and O, T and I)	(a) Develop an automated fallback mechanism that suggests alternative input methods or manual entry if OCR fails after multiple attempts.	FR16, NFR4	H3-2
	Incorrect item categorization	Items are miscategorized, resulting in incorrect insights	(a) OCR incorrectly parses text or fails to read some text (b) Ambiguous item name (c) Item has not been identified in the past	(a) Refer to H3-2. (b) Prompt the user to confirm the category of this object and give it an alias for helping in future recognition. (c) See H3-3b	FR17, NFR2	H3-3
	Processing image takes more than X seconds to complete	(a) Lower user satisfaction	(a) User's device is low on memory or has other tasks running in the background (b) Large image size/high image resolution (c) OCR model not optimized (especially for that specific use case) (d) Poor pre-processing	(a) Inform the user that the task is taking longer than expected and show troubleshooting tips. (b) See H3-2 (c) See H3-2 (d) See H3-2	NFR8, NFR16	H3-4
Save new receipt input (image details and model results)	Cannot connect to database	Receipt input cannot be saved after being processed.	(a) Poor network connection (b) Database server downtime	(a) Implement an automated network status checker that alerts users to connectivity issues and suggests troubleshooting steps. (b) Inform the user of the server error; system will try again at a later time (data will be stored locally and then backup up)	FR15, NFR13, SR5, SR8	H4-1

Continued from previous page

Design Function	Failure Modes	Effects of Failure	Causes of Failure	Recommended Action	SR	Ref
The system suggests user budget and goals	The system suggests a budget that is lower than the user's total monthly expenses (i.e. unattainable budget)	(a) Lower user satisfaction (b) User is unable to budget correspondingly to the suggested system budget	(a) Algorithm to calculate the budget fails	(a) Provide an option to the user to recalculate monthly budget with a given minimum monthly expense (user enters minimum monthly expense and system recalculates budget)	NFR5	H5-1
Data Synchronization	Network failure during data transmission	(a) Loss of data integrity (b) Inconsistent user experience (c) Potential data loss	(a) Poor network connectivity (b) Server downtime	(a) Establish an automated data synchronization protocol that ensures data integrity and consistency during network disruptions. (b) Establish real-time data backup protocols to maintain data integrity. (c) Develop detailed protocols for data validation and synchronization after network recovery to ensure consistency.	NFR20	H6-1
Image Processing Errors	Repeated image rejection due to poor quality or prolonged processing times	(a) User frustration leading to potential abandonment of the application (b) Decreased user satisfaction and retention	(a) Poor quality of input images (e.g., blurry, low lighting) (b) High processing times due to large image sizes or device limitations	(a) Integrate an automated error detection system that analyzes image quality and provides users with actionable tips to improve capture conditions. (b) Implement visual cues and real-time guidance to assist users in troubleshooting image quality issues. (c) Introduce a retry mechanism with tips for improving image capture (e.g., lighting, focus).	NFR16, NFR17	H7-1

6 Safety and Security Requirements

These are newly identified requirements that will be added to the Software Requirements Specification (SRS) document. They will be relabelled and added to appropriate functional and non-functional requirements sections in the SRS.

SR1 Account creation must include a captcha to verify that the account is being created by a human.
Rationale: Bot accounts can cause increased traffic to the server, invalid inputs, and performance deterioration.
Associated Hazards: H1-1

SR2 The application must prompt the user to allow camera access upon needing to open the camera. If camera access is disabled, the application must notify the user and show instructions on how to enable camera access.
Rationale: If the camera does not open, the user cannot input an image to be processed by the system.
Associated Hazards: H2-1

SR3 The application must prompt the user to allow photo library access upon needing to open the photo library. If photo library access is disabled, the application must notify the user and show instructions on how to enable photo library access.
Rationale: If the photo library does not open, the user cannot input an image to be processed by the system.
Associated Hazards: H2-2

SR4 The application will run image validation before passing the image through the ML model; the validation will check that there is a paper with text in the image.
Rationale: If the image is not processable, it may cause incorrect results or cause the model to run for a long time.
Associated Hazards: H3-1

SR5 The database will be store a local copy of receipts within the app until a connection to the server can be established.
Rationale: If the system cannot connect to the database, receipt input cannot be saved after being processed.
Associated Hazards: H4-1

SR6 The application will apply image sanitization techniques to ensure that malicious content cannot be injected into the app via uploaded images (e.g. XSS).
Rationale: Protecting against malicious content injection will help prevent attacks that could compromise user data or the system.
Associated Hazards: H3-1

SR7 User account passwords must be a combination of uppercase letters, lowercase letters, numbers, and symbols, and be at least 8 characters long.
Rationale: Weak passwords can lead to unauthorized access to user accounts.
Associated Hazards: H1-2

SR8 The application will ensure that images of receipts are securely stored using access controls and encryption and should be deleted when they are no longer needed or upon user request.
Rationale: Protecting receipt images will help prevent unauthorized access to sensitive financial data.
Associated Hazards: N/A

SR9 The application will be regularly updated to address newly discovered vulnerabilities, both in the application code along with any third-party libraries of frameworks. This will be done on an iterative basis with periodic patch releases (e.g. patch release every 3 weeks).
Rationale: Regular security updates will help protect against newly discovered vulnerabilities that could be exploited by attackers.
Associated Hazards: N/A

SR10 Before storing or processing any receipts that are uploaded, all sensitive information such as account numbers or personal details will be anonymized to protect user privacy.
Rationale: Anonymizing data will help protect user privacy and prevent unauthorized access to sensitive data. This is especially important when using data for AI training or analysis.
Associated Hazards: N/A

SR11 While the assumption is that users will be on a mobile device, the device must be secure (e.g. detecting if the device is jailbroken or rooted).
Rationale: This can help against local data theft or tampering.
Associated Hazards: N/A

SR12 The application will allow users to remotely wipe sensitive financial data and receipts from their devices in the case of loss or theft.
Rationale: This will help protect user data in the event of a lost or stolen device.
Associated Hazards: H1-2

7 Roadmap

Due to the time constraint of the capstone timeline, not all safety/security requirements identified in Section 6 of this hazards analysis document will be implemented; only a subset will be on the roadmap.

The requirements that will be implemented as part of the capstone time will be SR1, SR2, SR3, and SR7. The rest of the requirements will be implemented in the future.

As noted in the Section 9 of the SRS document, there are four phases for the development plan. The new requirements will be implemented in the following phases:

1. Phase 1: SR1, SR7
2. Phase 2: SR2, SR3

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 overall process while writing this deliverable was smooth and efficient as we were quickly able to identify the potential hazards related to our project. We brainstormed several ambiguous sections or things we thought were a bit unclear within this analysis document, and were able to get very clear answers from our helpful TA. The team worked well together as we all put in our best efforts and supported one another when completing this task.

   Using the FMEA (Failure Mode and Effect Analysis) approach helped streamline the hazard identification process. Breaking down the system into components allowed for a clear understanding of where risks might occur. Writing the deliverable helped the team clarify and solidify their understanding of how the receipt scanner, the AI model, and other system components interact, making it easier to identify hazards.

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

   At first, it was challenging to define the potential failure modes, especially for components like the machine learning model. The team resolved this by conducting additional research on common failure points in similar systems and reviewing how AI models typically behave with poor input data. Another challenge was balancing realistic assumptions about user behavior with potential risks. For example, while assuming users won't repeatedly input invalid images, we acknowledged this could still happen. We resolved this by planning mitigation strategies for those edge cases.

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?

   We had already considered risks related to image quality (e.g., blurry or incomplete receipt images) and network connectivity issues (e.g., users not being able to connect to the database).

   While working on the hazard analysis, we realized potential risks like Optical Character Recognition (OCR) misinterpretation due to varied receipt fonts and ML model processing time under different device conditions (e.g., low memory or poor network). These came up while brainstorming as a team and thinking about the specific steps in image processing and how the system handles diverse input.

4. Other than the risk of physical harm, list at least 2 other types of risk in software products. Why are they important to consider?

Two other risks that are apparent in software products are security and reliability risks.

Security vulnerabilities can lead to issues such as data breaches, unauthorized access or identity theft, as well as collateral damages, whether it be financial losses or reputational damage. This is considered a risk and is important to consider as it creates an opportunity for malicious users to exploit weaknesses in software systems, which can have a range of detrimental consequences. Examples include operational disruptions, intellectual property theft, ransomware attacks, etc.

As for reliability, it is mostly concerned with when software fails to function consistently, such as having frequent downtimes. This can affect the user's experience, leading to a loss of productivity or customer dissatisfaction. Both of these can lead to potential revenue loss. This is classified as a risk and is important to consider because unreliable software can lead to negative consequences, which affect not only the users but also the organization that provides the software. The damages can be both monetary and non-monetary, such as losing user trust/loyalty, reputational damage, and associated compliance and legal risks.