Hazard Analysis
Room8

Mohammed Abed
Maged Armanios
Jinal Kasturiarachchi
Jane Klavir
Harshil Patel

2024-10-25

Table 1: Revision History

Date	**Developer(s)**	**Change**
2024-10-25	Mohammed Abed, Maged Armanios, Jinal Kasturiarachchi, Jinal Klavir, Harshil Patel	Document Created

Contents

1 Introduction **1**

2 Scope and Purpose of Hazard Analysis **1**

3 System Boundaries and Components **1**

3.1 User Device . . . 1
3.2 Camera . . . 1
3.3 Motion Sensor . . . 1
3.4 User-Facing Application . . . 1
3.5 Authentication . . . 1
3.6 SMS ChatBot . . . 1
3.7 Calendar Tool . . . 2
3.8 Cleanliness Manager . . . 2
3.9 Bill Splitter . . . 2
3.10 Database . . . 2

4 Critical Assumptions **2**

5 Failure Mode and Effect Analysis **3**

6 Safety and Security Requirements **10**

6.1 Safety and Security Requirements . . . 10
6.2 Access Requirements . . . 10
6.3 Integrity Requirements . . . 11

7 Roadmap **12**

1 Introduction

Room8 is a suite of tools aimed at reducing the occurrence of frustrating situations between roommates. Room8 is expected to be implemented as a mobile application and interact with the physical world using a camera and as a result, is expected to handle sensitive user data such as addresses, names, birthdays, images, and financial details. As a result, the application can pose potential hazards to users with a hazard defined as any property or state of the system that could cause our users harm. This document aims to outline the scope, critical assumptions, potential failures, and mitigation strategies for Room8 in order to allow the development to design mitigations for these hazards.

2 Scope and Purpose of Hazard Analysis

The purpose of this document is to identify and mitigate losses that can be incurred from system hazards. There are multiple ways to mitigate the losses, such as following appropriate regulations, implementing thorough testing, and informing users how to properly use the system. By examining as many scenarios as possible where the system can cause harm and recording it in this document, the development team aims to minimize the harm exposed to users, stakeholders, and the development team. Possible losses that can occur from hazards include financial loss, loss of reputation, and service disruptions.

3 System Boundaries and Components

This section goes over the components that the system can be divided into.

3.1 User Device

Smart phone the user is using with the supported version of Android or iOS.

3.2 Camera

Responsible for taking picture for cleanliness detection analysis when sensor sends information of user.

3.3 Motion Sensor

Detects movement in the shared space to determine if user has entered or exited shared space signaling the camera for a picture.

3.4 User-Facing Application

A mobile application installed on smart phones which have versions of Android and iOS that is currently being supported mobile providers. This includes front-end of the system where users can see details and change settings of various back-end components listed below.

3.5 Authentication

Authentication using OAuth of user credentials and house details are processed in this component including the update of information mentioned previously.

3.6 SMS ChatBot

ChatBot responsible for sending messages to group chat of home members for notifying them of cleanliness assessment, expenses from bill splitter, or reminders to complete tasks.

3.7 Calendar Tool

Allows users to add events to calendar and display to other housemates, if involved in event, in their respective calendars. Also houses logic for generating chore/cleaning schedule and adding in calendars of users.

3.8 Cleanliness Manager

Runs algorithm for detecting change in environment through input received from hardware and stores user's information for the user to view on application along with history of cleanliness.

3.9 Bill Splitter

Calculate charges due from a shared expense and keeps track of which expenses are due from each user and who they owe using the SMS ChatBot to notify users. Also stores history of expenses and charges paid for user to view.

3.10 Database

Used to securely store user and house information, calendar events, expense history, and pictures for cleanliness calculator.

4 Critical Assumptions

- **CA1**: Homes will have a consistent and uninterrupted supply of electricity available.
- **CA2**: Homes will have internet speeds capable of streaming video.
- **CA3**: Every resident of a shared home will have their own personal electronic device.
- **CA4**: Users have used other applications before and are familiar with common signifiers, mappings, and UI metaphores (ex. Heart implies like).
- **CA5**: External services, such as location services, map integrations, and calendar APIs will be available and reliable.
- **CA6**: Users' devices will have additional free storage beyond the what's required for the applications install.

5 Failure Mode and Effect Analysis

Table 2: **FMEA Table**

Design Functions	**Failure Modes**	**Effects of Failure**	**Causes of Failure**	**Recommended Action**	**SR**	**Ref**	**Severity**
Camera Takes Picture	Camera captures unclear image	Cleanliness detection algorithm does not produce good results due to bad input	a. Camera field of view being blocked b. Problem with lighting	a. Take hourly pictures (i.e. as long as motion not detected), use last picture taken b. Same as H1.1a	IR1, IR2	H1.1	Medium
Camera Uploads Image Pair To Cleanliness Management System	Camera fails to upload the images	Cleanliness management system does not detect or notify users of the changes to the space	a. Internet shuts down for an extended period of time	a. Retry to upload images multiple times b. Notify users if there has been no activity detected from the camera for an extended period of time	IR3, IR4	H2.1	Low

Design Functions	**Failure Modes**	**Effects of Failure**	**Causes of Failure**	**Recommended Action**	**SR**	**Ref**	**Severity**
Motion Sensor Detects Motion	Motion should not be detected (false positive)	Unnecessary computation and notifications	a. Motion sensor is overly sensitive b. Some brief/light movement that should not be classified as motion occurred (e.g., insect flew by) c. Continuous movement that should not be classified as motion is occurring (e.g., air conditioner causing curtain to move)	a. Include regular motion-calibration testing in system design b. Have a clause in the cleanliness detection algorithm to check for false positives (i.e., if the before/after pictures do not meet a threshold to be considered dissimilar) c. System does not classify minor continuous movement (i.e. that does not exceed some threshold) as motion	SR2	H3.1	Medium
	Motion is not detected although it is occurring (false negative)	Cleanliness detection does not occur	a. Motion sensor is underly sensitive b. Motion sensor is broken	a. Same as H3.1a b. Same as H3.1a	SR2	H3.2	Low

Design Functions	**Failure Modes**	**Effects of Failure**	**Causes of Failure**	**Recommended Action**	**SR**	**Ref**	**Severity**
System Authenticates User	Bad actor is able to log into user's account	Sensitive user information is revealed, such as full name and address. User privacy is violated.	a. Data breach b. Weak account password c. Lack of multi-factor authentication d. Brute force attacks e. Man-in-the-middle attacks f. Out-dated/insecure methods of storing user credentials	a. Check for unusual login patterns, such as different geolocations, IP addresses, and repeated failed attempts on same login b. Require all account passwords to satisfy a minimum password strength criterion c. Reccommend multi-factor authentication for logging in d. Impose rate limits on failed authentication attempts e. Salt and hash passwords	SR3, SR4, SR5	H4.1	High
	Bad actor alters account credentials	User cannot access their account	a. Lack of multi-factor authentication for changing account credentials	a. Require multi-factor authentication for changing account credentials b. Same as 4.1a c. Same as 4.1d	SR3, SR6	H4.2	High
	User forgot their password	User cannot access their account	a. No way to recover account	a. System has a "forgot password" clause where multi-factor authentication is used to create new password	SR6	H4.3	Low

Design Functions	**Failure Modes**	**Effects of Failure**	**Causes of Failure**	**Recommended Action**	**SR**	**Ref**	**Severity**
ChatBot Sends Notifaction	User is notified of deleted event	False information sent to user and user gets annoyed	a. ChatBot SMS did not get update from calendar that event was deleted and user does not need to be informed	a. Have test cases covering testing if ChatBot SMS is updated when calendar events update	NFR222	H5.1	Medium
	User is not notified of an event	Same as HR7.1	a. ChatBot SMS failed to get event from calendar	a. Make sure ChatBot SMS synchronizes with calendar.	NFR222	H5.2	Medium
Scheduling an Event in the Calendar	Event cannot be scheduled	User is frustrated, and important information is not being sent to roommates	a. Conflict occurred due to multiple users scheduling events simultaneously	a. Put a lock on the calendar resource	SR7	H6.1	Medium
Cleanliness Management System Detects Changes in the Cleanliness of a Room	System falsely concludes that a room has become more dirty	Conflict amongst roommates and trust in the system declines	a. Obstructions in the images captured by the camera b. Improper calibration and timing of the motion sensor c. Object detection algorithm has errors and classifies items incorrectly in an image	a. Create base case tests for the cleanliness management system, including no change, increase, decrease, and no room state change cases b. Alert and instruct users to clear camera obstructions before setting up the system	FR231, FR232, FR233	H7.1	Low
	System falsely concludes that a room has become more clean	Same as H7.1	Same as H7.1	Same as H7.1	SR6	H7.2	Low
	System concludes that there are no changes despite there being changes	Same as H7.1	Same as H7.1	Same as H7.1	SR6	H4.3	Low

Design Functions	Failure Modes	Effects of Failure	Causes of Failure	Recommended Action	SR	Ref	Severity
	The system concludes that there are changes when there are no changes	Confusion among users and a decline in trust of the system	Same as H7.1	Same as H7.1	SR6	H7.3	Low
Inputting into Bill Splitter	Amount owing not accurate	Users receiving false information	a. User made an error inputting bill b. The bill changed (i.e. amount or people owing) c. Bad actor creating false bills	a. Provide users with a mechanism to edit outstanding bills b. Same as 8.1a c. System has a way for roommates to delete bills and report misuse	FR252, FR253	Medium	
Data Storing User Data	Data is corrupted during storage	a. User unable to track actions b. User experiences unresponsive interface c. Database synchronization failure	a. Database hardware failure b. Power loss during write operations c. Software bugs during data writing or saving processes	a. Use redundant and fault-tolerant storage solutions (cloud storage) b. Perform regular backups c. Use transaction logs to ensure data integrity	SR8, IR5	H10.1	Medium
	Insufficient storage capacity for new data	a. System failure or user actions being blocked b. System slows down as disk capacity nears limit	a. Poor data capacity planning b. Lack of automated database scaling c. Failure to monitor storage space usage in real time	a. Implement real-time monitoring and alerts of storage space usage b. Use cloud storage with automatic scaling options c. Archive or delete obsolete data to free up space	IR6	H10.2	Medium

Design Functions	**Failure Modes**	**Effects of Failure**	**Causes of Failure**	**Recommended Action**	**SR**	**Ref**	**Severity**
Users Retrieving Stored Data	Slow response when retrieving data	a. Delays in system performance, leading to poor user experience b. Data query timeouts causing other systems to fail	a. Poorly optimized queries and lack of indexing b. High database traffic without load balancing	a. Optimize queries and implement indexing strategies b. Implement load balancing and database replication to handle high traffic	AR1	H11.1	Low
	Incorrect or missing data during retrieval	a. Users make incorrect decisions based on incomplete or incorrect data b. Loss of trust in the system if the information retrieved is unreliable	a. Incorrect query logic or data corruption b. Data deletion or loss during storage processes	a. Perform thorough testing of queries and retrieval functions b. Implement data validation during retrieval to catch incorrect or missing records	IR7	H11.2	Medium
	System becomes unavailable during data retrieval	a. Users cannot access data, resulting in downtime or service outages b. Lost productivity and user dissatisfaction	a. System crashes due to hardware or network failures b. Overload on the database server due to high traffic c. Software bugs causing server crashes	a. Use high-availability database solutions and clusters b. Implement real-time monitoring and alerts for system load c. Scale infrastructure based on traffic spikes	SR8	H11.3	Low

Design Functions	**Failure Modes**	**Effects of Failure**	**Causes of Failure**	**Recommended Action**	**SR**	**Ref**	**Severity**
Database Protecting User Information	Data exposed due to a security breach	a. Compromise of sensitive user information, leading to legal or regulatory consequences b. Loss of user trust and damage to application reputation c. Hesitancy to use camera for cleanliness monitoring	a. Weak encryption for sensitive data b. SQL injection attacks and cyber attacks c. Poor access control policies	a. Encrypt all sensitive data both in transit and at rest b. Parametrize queries and sanitize inputs to avoid SQL injection c. Implement strict access controls such as RBA	SR9	H12.1	High
	Unauthorized access to sensitive data	a. Malicious actors gain access to sensitive data, risking data misuse or theft b. Potential for data manipulation or malicious changes to critical information	a. Weak authentication mechanisms b. Lack of multi-factor authentication (MFA) c. Social engineering or phishing attacks on developers	a. Implement multi-factor authentication for database access b. Enforce strong password policies with regular expiration c. Educate developers on phishing and social engineering threats	AR2	H12.2	High

6 Safety and Security Requirements

The following lists new requirements which will be added to the Software Requirements Specification Document.

6.1 Safety and Security Requirements

SR1. The system will not show images of other users in the frame without proper sensoring.
Rationale: To maintain user privacy and comfort.

SR2. The system will perform regular hardware tests.
Rationale: To ensure hardware components are operating as expected.

SR3. The system deny suspicious login and password recovery attempts. This includes logins from drastically different geolocations or repeated login attempts.
Rationale: To maintain a users' account security.

SR4. The system will implement a minimum password strength criteria.
Rationale: To maintain a users' account security.

SR5. The system will hash and salt all confidential user data.
Rationale: To maintain users' data privacy.

SR6. The system will allow users to recover their password using their email.
Rationale: To allow users access to their account in case they forget their password.

SR7. The system will lock the calendar resource when a user is scheduling an event.
Rationale: To prevent scheduling conflicts with other users.

SR8. The system will use cloud storage with high-availability and auto-scaling capabilities to ensure continuous access to data.
Rationale: To minimize downtime and allow users uninterrupted access to information by using the redundancy and high availability support provided by cloud infrastructure. Cloud storage also provides many benefits such as automatic scaling and monitoring.

SR9. The system will encrypt all sensitive data at rest and in transit using industry-standard encryption protocols.
Rationale: To protect sensitive user information from unauthorized access and ensure compliance with data security standards.

6.2 Access Requirements

AR1. The system will implement optimal index using database index advisors to ensure data retrieval times do not exceed 3 seconds under normal load.
Rationale: To improve user experience and ensure timely access to stored information.

AR2. The system will enforce multi-factor authentication (MFA) for all users with access to sensitive information.
Rationale: To prevent unauthorized access by adding an extra layer of security to sensitive data access from developers.

6.3 Integrity Requirements

IR1. The system will send clear images of the shared space.
Rationale: It is important the images are not hindered by objects covering majority of the share space.

IR2. The system will take pictures of the space at regular intervals so long as motion is not being detected.
Rationale: This will allow the system to have a series of images over various lighting and atmospheric changes. Bad input into the cleanliness detection algorithm is mitigated, since an improper (e.g. noisy, obstructed) picture at time of movement can be swapped with a good picture that represents the same cleanliness state.

IR3. The system will retry to upload images when network connectivity is restored if it has failed has failed due to connectivity issues.
Rationale: The cleanliness management system is one of our core features, to make the system more robust to periods of time when the network shutdown it should attempt to retry so an event that happens when there's no internet is still captured.

IR4. The system will notify users if there has been no camera activity for an extended period of time.
Rationale: The system is expected to be set up in a high traffic area, prone to frequent use. A lack of activity in that area would be unusual and could imply the system isn't working properly.

IR5. The system will implement data integrity checks during data writing operations to verify successful storage.
Rationale: To ensure that user data remains uncorrupted and reliable for future retrieval. System will also be able to alert users if data is corrupted or if the action has failed, reducing confusion and frustration.

IR6. The system will monitor storage capacity in real-time and alert administrators when usage exceeds 80% of total storage capacity.
Rationale: To prevent system slowdowns and failure to store new data by ensuring storage capacity is sufficient before reaching limit thresholds.

IR7. The system will perform data validation checks during data retrieval to verify the completeness and correctness of requested information.
Rationale: To ensure users receive accurate and complete data, maintaining trust in the system. System will also be able to alert users if data is missing or incorrect, reducing confusion and frustration.

7 Roadmap

The hazard analysis has led to identifying essential safety, security, access, and integrity requirements to mitigate the possible risks in our system. The current focus will be on implementing core requirements which prevent major issues down the line as well as foundational tasks of the application that are harder to change in later stages of development. These include tasks such as data encryption, secure authentication, and high-availability and reliability cloud storage. In the next stage of development, feature component hazards will be tackled, such as safeguards for the calendar tool, cleanliness manager, and the bill splitter. Due to time constraints, certain non-critical requirements may not be fully implemented within the project timeline. For instance, more advanced cloud storage auto-scaling and redundancy configurations may be considered for future releases, whereas industry standard defaults will be used in the initial development stages. Towards the end of the project, the hazard analysis will be revisited to assess which risks have been effectively mitigated and identify any remaining vulnerabilities that require attention in future iterations.

Appendix — Reflection

1. While writing this deliverable, the team had a relatively difficult time constructing the FEMA table. Referring back to previous documents, such as the SRS and the Development Plan, made it easy to extract the core design functions of our application and start evaluating the associated hazards. Another thing that had gone well when writing this deliverable was the discovery of new requirements/tests for the sake of minimizing requirements. An example of this is the cleanliness management system, we determine that the recommended action for ensuring no false positives and negatives is to create a set of test cases. Finally, we think a good thing that came out of this deliverable was that we realized we had missed a lot of requirements from our SRS, and we're expecting to use what we learned from this document to revise it.

2. The most prominent pain point when writing this deliverable was that we had difficulty dividing the work and working in parallel. This document is a lot shorter than the SRS, and in this deliverable, each section was a lot more dependent on its previous sections. When it was time to work on the deliverable, the team got caught up with other obligations, leading to less time to work on this than we would have liked. Combining this fact with the fact that some people had to wait for others in order to finalize their sections led to a time crunch.
   Another pain point was the flexibility in formatting our document compared to the SRS. The SRS was more structured than this document, and we knew exactly how to format our deliverable, as there wasn't freedom to alter it. This deliverable gave the group a lot more freedom in formatting some sections, which led to inconsistencies when working on the deliverable. These inconsistencies required group discussions as we needed to agree on a style, and then we had to revise our document to ensure it was cohesive and consistent.

3. The risks related obstructions related to the camera as well as false positives, negatives, and misclassification of cleanliness changes had been discussed before. These risks were discovered by our supervisor Dr. Ratnasingham (Thamas) Tharmarasa, who challenged many of our ideas when we were initially pitching the project and had asked us questions which made us realize we had made too many assumptions about our environment. Some other risks, particularly involving bad actors accessing user accounts, had been discussed before this deliverable, but only in passing when some implementation ideas were discussed as a group in an informal setting. The other risks had all come about naturally by simply trying to ask ourselves, "What could happen?" and "If this did happen, what could go wrong?".

4. Other risks in software products include data privacy risks and security risks. Data privacy risks refer to risks that arise when a software product handles users' personal data poorly, leading to potential data leaks. Data privacy is arguably the most important requirement when designing a software system that accesses sensitive user data. If you mishandle a user's private information, the developer of the software is liable for the damage caused, which could be very severe depending on the type of data stolen. Not only will mishandling user data lead to a severe loss in user trust, but failing to comply with regulations can lead to legal consequences.
   Security risks involve vulnerabilities that could allow malicious actors to gain unauthorized access, manipulate data, or disrupt services. Security breaches are important as well because they can cause severe harm to users and lead to financial loss due to operational losses. Without the proper security measures, malicious actors could disrupt your service, gain access to large amounts of user data, and use that to produce harm outside the scope of your software service.