SOFTWARE REQUIREMENTS SPECIFICATION

Room8

Version 0.0.0

Prepared by:
Mohammed Abed
Maged Armanios
Jinal Kasturiarachchi
Jane Klavir
Harshil Patel

McMaster University

October 11, 2024

Revision History

Name	Date	Reason For Changes	Version
Mohammed A. Maged A. Jinal K. Jane K. Harshil P.	2024-10-11	First Draft	0.0

This document follows the requirements documentation structure presented in the Handbook of requirements and business analysis, by Bertrand Meyer.

Contents

Goals

Contents

Comment: *Goals are "needs of the target organization, which the system will address". While the development team is the principal user of the other books, the Goals book addresses a wider audience: essentially, all stakeholders.*

G.1 Context and overall objective

Shared living environments often create tension between roommates. It is usually cumbersome to reach out to a roommate about messes they left behind in a shared space, concerns over bill payments, household upkeep, and more. Our project, Room8, aims to prevent these cumbersome interactions by providing an application and the necessary hardware components to help monitor the cleanliness of shared spaces, schedule tasks, track shared billing cycles, and alert roommates of any issues within the shared house.

Goal 1. Create a system that monitors the cleanliness of a shared living space, such as a kitchen or a living room.

Goal 2. Provide roommates with a centralized platform to manage, schedule, and task household tasks.

Goal 3. Provide roommates with a centralized platform to manage, schedule, and track bill payments.

Goal 4. Alert and remind roommates of issues regarding issues related to the shared living spaces, reducing tension between each other.

G.2 Current situation

Room8 is designed to reduce points of tension and reduce the need for frustrating dialogue commonplace with people in shared housing, especially students. Currently, in order to manage the many shared responsibilities, roommates have to utilize multiple techniques or tools such as physical calendars, notification reminders, and existing applications such as Splitwise which do not have any integrations between each other. Additionally, it is estimated that approximately 25% students experience conflicts with a roommate, harming academic performance and inducing stress [1]. Recognizing these factors, Room8 aims to create a centralized platform that not only simplifies communication but reduces stress and conflict to improve the lives of students.

G.3 Expected benefits

As stated in section G.2, conflicts between roommates create more than issues in the home and involve more than just misunderstandings in scheduling tasks or living standards. By creating a centralized suite that will help roommates monitor, schedule, and coordinate, Room8 will reduce conflicts between roommates over misunderstandings and miscommunications, and allow for greater flexibility in assigning responsibilities. Additionally, reduced conflicts will lead to mental health benefits for those living in the home by lowering conflict-related stress. Finally, Room8 is expected to improve academic performance in some students who are hindered academically by conflicts with roommates (approximately 17

Benefit 1. An improvement in response times amongst roommates over household concerns.

Benefit 2. Increased flexibility in task scheduling and management, due to the presence of a centralized application.

Benefit 3. Improved communication and transparency over household matters between roommates.

Benefit 4. A reduction of events in which a roommate forgets or neglects their responsibilities to the home.

Benefit 5. Improved household cleanliness and upkeep.

Benefit 6. A reduction in conflicts between roommates over household matters.

Benefit 7. A reduction in stress for roommates due to reduced conflicts amongst each other.

Benefit 8. Improved mental health for students who experience conflicts with roommates.

Benefit 9. An improvement in academic performance for students experience stress caused by issues with roommates.

G.4 Functionality overview

Comment: *Overview of the functions (behavior) of the system. Principal properties only (details are in the System book).*

User & House Management: The system allows students to register, create, and manage their accounts. Students are also able to create homes and invite other students.

Cleanliness Management: Students are able to set up cameras in their shared living spaces. Machine learning algorithms are used to detect messes and assign them to the students to increase accountability and to reduce communication friction.

Scheduler: The system allows users to create and manage chore and cleaning schedules. It will also send reminders to users about their assigned chores and cleaning tasks. Additionally, users are able to "book" common areas that are available within the shared space (i.e. party room, study room, etc.) to avoid conflicts

Bill Splitter: Students are able to add shared expenses to the house and keep track of who owes what. The system will automatically split the bills between the chosen students.

Chat: The system provides an exportable chatbot to SMS group chats to aid in home management and communication. The chatbot will be able to send remainders and notifications to the group chat about upcoming events, chores, and bills.

Compliance and Data Privacy: The system follows data privacy and compliance with Personal Information Protection and Electronic Documents Act (PIPEDA), Ontario's Freedom of Information and Protection of Privacy Act (FIPPA), and Anti-Spam Legislation (CASL) to ensure that student's data is secure and not shared with third parties.

Progressive Web App: The system is hosted as a progressive web app (PWA), so users can use it on their phones and computers.

G.5 High-level usage scenarios

Comment: *Fundamental usage paths through the system.*

Figure 1: High level use case diagram

- UC1: Create an account
  1. User accesses the Room8 app.
  2. User selects the "Create Account" option.
  3. User fills in personal information (name, email, password).
  4. User submits the form.
  5. System validates the information and creates the account.
  6. System authenticates the user.
  7. System redirects the user to their dashboard.
- UC2: Log in to the system
  1. User accesses the Room8 app.
  2. User selects the "Log In" option.
  3. User enters email and password.

4. User submits the form.
5. System authenticates the credentials and logs in the user.
6. System redirects the user to their dashboard.

- UC3: Log out of the system
  1. User accesses the main menu of the Room8 app.
  2. User selects the "Log Out" option.
  3. System prompts for confirmation.
  4. User confirms the action.
  5. System logs the user out and redirects to the login screen.
- UC4: Create a home
  1. User navigates to their dashboard.
  2. User selects the "Create Home" option.
  3. User enters home details (home name, address, etc.).
  4. User sends invitations to housemates by entering their email addresses.
  5. System sends email notifications to invited users.
  6. Invited users accept or decline the invitation.
- UC5: Join a home
  1. User receives an email invitation to join a home.
  2. User clicks on the provided link in the invitation.
  3. System redirects user to the Room8 app.
  4. User logs in or creates an account if not logged in.
  5. User confirms their decision to join the home.
  6. System adds the user to the home group.
- UC6: Leave a home
  1. User navigates to their Home settings.
  2. User chooses the "Leave Home" option.
  3. System prompts for confirmation.
  4. User confirms the action.
  5. System removes the user from the home group.
- UC7: Update user profile
  1. User navigates to the “Profile” section.
  2. User selects the "Edit Profile" option.

3. User modifies personal information (name, profile picture, etc.).
4. User saves changes.
5. System updates the user's profile information.

- UC8: Schedule a chore
  1. User navigates to the "Schedule" section.
  2. User selects "Add New Chore".
  3. User inputs chore details (name, description, time, frequency, assigned users, etc.).
  4. System saves the chore and sends reminders to the assigned users.
- UC9: Edit a chore
  1. User navigates to the "Schedule" section.
  2. User selects a scheduled chore.
  3. User chooses the "Edit" option.
  4. User modifies the chore details (name, description, time, frequency, assigned users, etc.).
  5. System updates the chore and notifies relevant housemates.
- UC10: Complete a chore
  1. User navigates to the "Schedule" section.
  2. User selects a chore assigned to them.
  3. User chooses the "Complete" option.
  4. System marks the chore as complete and updates the chore history.
- UC11: Add event to schedule
  1. User navigates to the "Schedule" section.
  2. User selects the "Add Event" option.
  3. User inputs event details (date, time, description).
  4. User sets reminders or notifications if needed.
  5. System adds the event to the users' schedule.
- UC12: View chore history
  1. User navigates to the "Schedule" section.
  2. User selects the "Chore History" option.
  3. System displays a list of completed and pending chores.
  4. User can filter or search through the history.

- UC13: Add an expense
    1. User navigates to the “Bill Splitter” section.
    2. User selects the ”Add Expense” option.
    3. User enters the total expense amount and description.
    4. User selects which housemates are responsible for the expense.
    5. System splits the expense and notifies the involved users.
- UC14: Edit an expense
    1. User navigates to the “Bill Splitter” section.
    2. User selects an expense to edit.
    3. User modifies the expense details (amount, description, responsible users).
    4. System updates the expense and notifies the involved users.
- UC15: Pay an expense
    1. User navigates to the ”Bill Splitter” section.
    2. User selects their profile’s bills.
    3. User selects an unpaid bill.
    4. User marks the bill as paid.
    5. System updates the payment status.
    6. System notifies other housemates about the payment status.
- UC16: View expense history
    1. User navigates to the ”Bill Splitter” section.
    2. User selects the ”Expense History” option.
    3. System displays a list of past payments and outstanding bills.
    4. User can filter the history by type or amount.
- UC17: View current cleanliness status
    1. User navigates to the “Cleanliness Management” section.
    2. System displays the current cleanliness score and detected messes.
    3. User selects another user to view their cleanliness score.
    4. System displays the selected user’s cleanliness score and pending tasks.
- UC18: View cleanliness history
    1. User navigates to the “Cleanliness Management” section.
    2. User selects the ”Cleanliness History” option.
    3. System displays a list of detected messes and cleanliness scores over time.

4. User can filter the history by location or date.

- UC19: Add chatbot to group chat
  1. User navigates to the “Chat Management” section.
  2. User selects the option to add a chatbot to their house group chat.
  3. User inputs notification and reminder settings.
  4. System generates an SMS numbers and instructions to add the chatbot to the group chat.
  5. The chatbot sends a welcome message to the group.
- UC20: Edit chatbot settings
  1. User navigates to the “Chat Management” section.
  2. User selects the ”Chatbot Settings” option.
  3. User configures the chatbot’s behavior (e.g., reminders, notifications).
  4. User saves the settings.
  5. System updates the chatbot based on the new settings.

G.6 Limitations and exclusions

Comment: *Aspects that the system need not address.*
Below is a list of limitations and exclusions the system will not address:

Limitation 1. System will not track activity completed by the user in the shared environment.

Exclusion 1. System will not request or send money directly to users in the bill splitting functionality.

Exclusion 2. System will not use images taken to train machine learning model.

G.7 Stakeholders and requirements sources

Comment: *Groups of people who can affect the project or be affected by it, and other places to consider for information about the project and system.* **Comment:** *This chapter should not be empty!*

G.7.1 Direct Stakeholders

Students
Students are the primary direct stakeholders for Room8. They are the main users of the mobile application who create houses within the app and set up the camera systems. These students seek to maintain cleanliness in their shared living spaces and establish

Stakeholder	Category
Students	Direct
Home Managers	Indirect
University Housing and Social Committee	Indirect

Table 1: Stakeholders and Categories

accountability when a mess is left behind by a roommate. The application addresses common challenges faced by students in shared living arrangements, such as maintaining cleanliness, splitting expenses, and scheduling activities.

G.7.2 Indirect Stakeholders

Home Managers
Home managers are indirect stakeholders to the project. Home managers includes landlords renting out their homes to students or residence assistants managing a room of students. The home managers look to maintain the clean condition of the shared space which is done by holding students accountable for messes that are made.

University Housing and Social Committee
The University Housing and Social Committee is another key indirect stakeholder in the project. These committees often seek to help students transition into living in shared spaces and provide guidance and support. Room8 offers a wide range of services that address common points of frustration faced by students, which are often brought up to these university committees. By facilitating better communication and organization, Room8 helps enhance the overall living experience for students.

Environment

Contents

Comment: *The Environment book describes the application domain and external context, physical or virtual (or a mix), in which the system will operate.*

E.1 Glossary

The glossary provides definitions for the key terms used throughout the project. It includes terminology related to the system's environment and functionality, ensuring that all parties involved have a clear understanding of the concepts and language specific to the project. This helps prevent miscommunication and fosters consistency in documentation and development.

E.2 Components

This section outlines the relevant external components that the system will interact with. These include existing systems or services, particularly software, that will provide or consume APIs. These interactions play a significant role in how the system operates within its broader ecosystem, impacting design and integration.

Motion-activated camera: The app will be connected to a camera that is activated by motion in the shared space. This component is used for taking a "before" and "after" picture that serves as the input for the cleanliness detection model.

Object detector API: The app will utilize an object detector that performs on a pair of images (i.e. the "before" and "after" picture) taken by the motion-activated camera. Object detection enables the algorithm to quantify the difference in room states as a function of objects and their transformations.

Household items dataset: The ML model will be pre-trained with a dataset containing common household and kitchen items. This will allow the object detector to classify the items it detects and is important for providing valuable output to students.

Google calendar API: The application will interface with Google calendar to facilitate scheduling chores and events. This is critical for Room8's chore scheduling.

Chatbot API:

E.3 Constraints

Constraints are non-negotiable limitations imposed on the system, stemming from external factors such as business rules or technical requirements. These restrictions must be strictly adhered to during development. Constraints can influence system design and implementation, and their fulfillment is critical to ensuring the system meets its operational requirements.

Camera resolution: The resolution of the motion-activated camera affects the quality of the input that is used by the object detector. Depending on the PPM of the image, there is a minimum viable object size for a given object to be classified.

Platform compatibility: The application must be compatible on Android and iOS to accommodate all users in the home.

Data privacy compliance: The application adheres to all applicable data privacy regulations for the region in which it operates. User data must be handled with the utmost care and cannot be shared or used without explicit consent. Prior to collecting, storing, or processing any personal data, the appropriate permissions must always be obtained to ensure compliance with privacy laws and to respect user rights.

E.4 Assumptions

Assumptions are conditions that are presumed to be true during development to simplify system design. These are not externally enforced but are accepted for the sake of

convenience or efficiency. While helpful, assumptions must be carefully evaluated, as changes in the assumed conditions could affect the system's success or performance.

Network connectivity: The application assumes that there is reliable internet connection. Operating under this assumption, the camera is able to transfer images to the detector and users have access to updated synchronized information when using the app.

Operating system support: The application is compatible with the versions of Android and iOS that are currently supported by mobile providers.

Common household items dataset: The classifier can assume that any object it comes across is a common household item that belongs to a class represented in its training data.

FoV of the camera: The camera is stationary and the FoV is identical in any pair of "before" and "after" images.

E.5 Effects

This section describes how the system's operations will influence its environment. These effects could be on processes, workflows, or other systems that interact with the system. Understanding these impacts is crucial to ensure that the system's operation aligns well with its intended environment and does not cause unintended disruptions.

Digital minimalism: Members of the home conduct all interactions on one platform. All common virtual services are provided by the Room8 application, hence eliminating the need for other applications that clutter the digital space.

Accountability: The application promotes social responsibility and user accountability by assessing the cleanliness state of the room before/after use and quantifying the difference in the 2 states. Students are given objective feedback on how their usage of the room contributed to cleanliness and all students in the group are held accountable for their usage.

Ease social tension: Students living in a home can feel tense about confronting their roommates when asking them to clean up, pay them back, or do the chores they agreed to. The chatbot sends reminders about these duties, taking the pressure off students to confront each other and risk compromising their relationships.

E.6 Invariants

Invariants are properties of the environment that the system must maintain throughout its operation. These are foundational conditions that must hold true at all times during system activities. Ensuring that these invariants remain intact is essential for the correct and safe functioning of the system within its environment.

Limited camera presence: The motion-activated camera is initially triggered by motion, and takes a "before" picture. When motion has not been detected for a prolonged and predefined amount of time, it takes an "after" picture. These 2 images aside, the camera does not take any pictures or footage and thus does not have any insight on the activities occurring in the living space.

Students' freedom to not comply: While the application sends reminders to students for them to pay their debts or clean up their messes, it cannot enforce these events. Room8 serves as a logistical coordinator but it cannot have a direct impact on the physical environment.

System

Contents

Comment: *The System book refines the Goal one by focusing on more detailed requirements about the system under development, mainly its constituents, behaviors and properties.*

S.1 Components

Comment: *Overall structure expressed by the list of major software and, if applicable, hardware parts.*
Comment: *This chapter should not be empty!*

Front-End Cluster

- **User Authentication and House Management:** The user interface which allows the user to log in and manage their account and/or house. It depends on a corresponding back-end component and allows the user to access all other functionality within the application once authenticated.
- **ChatBot Configuration:** The view which allows users to configure their ChatBot settings. This includes features such as opting in or out of certain personal and group notifications.
- **Cleanliness Manager:** Allows users to view their relevant details within the cleanliness detection section of the application. Includes details such as history of events within the shared space (i.e. time of entry, time of exit, changes made, etc.).

- **Schedule Configuration:** Users are able to configure details related to the scheduler within this view. Options to add/remove chores (including assigned users) and events (booked spaces) are found here.
- **Bill Splitter Configuration:** The area of the application where users can input and view details regarding shared expenses. Includes configuration to add/remove/edit bills and which users to split with, as well as outstanding balances and history of balances paid. Additionally, users can manually "settle" their outstanding payments by confirming that they've been paid within this interface.

Back-End Cluster

- **Auth and House Management Logic:** The logic for authentication and user/house details are housed here. Any updating of information (as per user request) will be carried out in this component. Authentication (through OAuth) will also be the responsibility of this software.
- **SMS Chatbot:** Responsible for sending out chatbot messages to home group chats including reminders to complete tasks, notification of cleanliness assessment, expenses due from bill splitter, or update to configuration of user/home. Has the most dependencies with other back-end components since updates in those components will drive the output of new messages from the bot.
- **Cleanliness Calculator:** Runs the algorithm to determine which changes have been made in the shared space by the identified user. Receives input from the hardware (dependenciy) and returns changes to the GUI after each event.
- **Schedule Generator:** Houses logic that deals with the scheduling aspect of the application. Generates events to be displayed to each users calendar based on whether or not they're included in a chore or event.
- **Bill Split Calculator:** Calculation of charges due for shared expenses is done within this component. It will also keep track of which expenses are due by each user and who they owe, along with a history of charges paid.

Hardware Cluster

- **Camera + Sensors:** All hardware used for the cleanliness calculation. Includes sensors to detect when a user enters or exits the shared space. Takes images which are sufficient quality for detecting the changes that a user makes within the space.

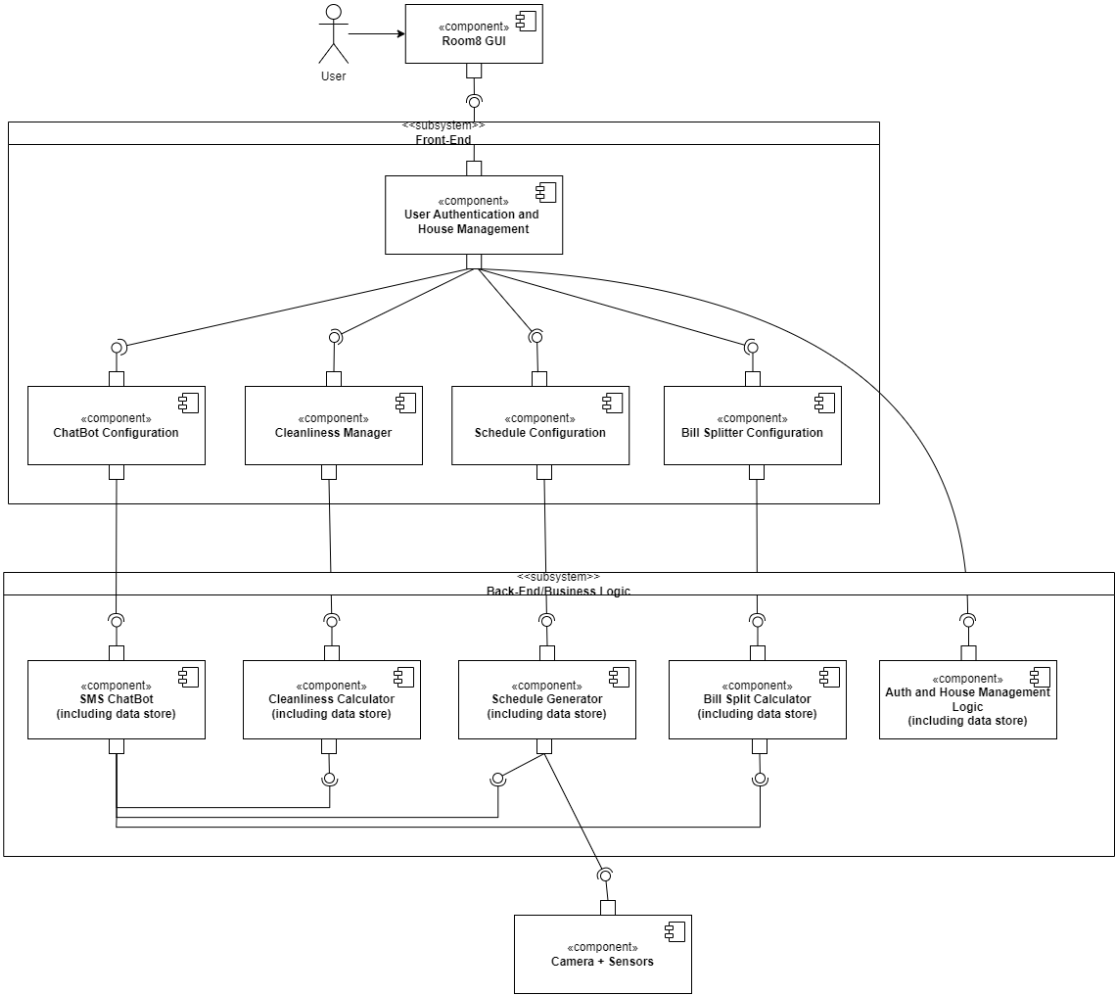

Figure 2: Component Diagram

S.2 Functionality

Comment: *One section, S.2.n, for each of the components identified in S.2, describing the corresponding behaviors (functional and non-functional properties).* **Comment:** *This chapter should not be empty!*

S.3 Interfaces

Comment: *How the system makes the functionality of S.2 available to the rest of the world, particularly user interfaces and program interfaces (APIs).*

S.4 Detailed usage scenarios

Comment: *Examples of interaction between the environment (or human users) and the system: use cases, user stories.*

S.5 Prioritization

Comment: *Classification of the behaviors, interfaces and scenarios (S.2, S.3 and S.4) by their degree of criticality.*

S.6 Verification and acceptance criteria

Comment: *Specification of the conditions under which an implementation will be deemed satisfactory.*

Project

Contents

Comment: *The Project book describes all the constraints and expectations not about the system itself, but about how to develop and produce it.*

P.1 Roles and personnel

Comment: *Main responsibilities in the project; required project staff and their needed qualifications.*

P.2 Imposed technical choices

Comment: *Any a priori choices binding the project to specific tools, hardware, languages or other technical parameters.*

P.3 Schedule and milestones

- **Milestone 1 (September 23rd) - Initial concept and development plan**
  - Deliverable: Problem Statement and Goals
  - Deliverable: Development Plan
- **Milestone 2 (October 11) - Requirements**
  - Deliverable: SRS
- **Milestone 3 (October 23) - Hazard Analysis**

  - Deliverable: Hazard Analysis Report

- **Milestone 4 (November 1st) - V&V Plan**
  - Deliverable: V&V Plan
- **Milestone 5 (Within November 11-22) - POC demonstration**
  - Event: Informal Project Demonstration
  - Goal: Core features functional
- **Milestone 6 (January 15) - Design Documents**
  - Deliverable: Software Architecture Document
  - Deliverable: Detailed Design Document
- **Milestone 7 (within February 3-14) - Revision 0 & Project completion**
  - Goal: Project is Complete and Functional
- **Milestone 8 (March 7th) - Verify & Validate**
  - Deliverable: V&V Report
  - Goal: Project meets all requirements outlined in the SRS and V&V Plan
- **Milestone 9 (within March 24-30) - Final demonstration**
  - Event: Final Project Demonstration
  - Goal: Unmet requirements from Milestone 8 are resolved
- **Milestone 10 (April) - Project EXPO demonstration**
  - Deliverable: Project Poster
  - Event: Project Expo
- **Milestone 11 (April 2nd) - Final documentation**
  - Deliverable: Final Problem Statement
  - Deliverable: Development Plan
  - Deliverable: POC Plan
  - Deliverable: Updated Requirements document
  - Deliverable: V&V Plan
  - Deliverable: V&V Report
  - Deliverable: User Guide
  - Deliverable: Project Source Code

P.4 Tasks and deliverables

Comment: *Details of individual tasks listed under P.3 and their expected outcomes.* **Comment:** *This chapter should not be empty!*

P.5 Required technology elements

Comment: *External systems, hardware and software, expected to be necessary for building the system.*

Camera
A camera is used for the systems cleanliness detection aspect. A camera takes pictures for comparing the state of the shared space.

Motion Sensor
A sensor is also used for the systems cleanliness detection aspect. The sensor detects movement which will work in unison with the camera to take pictures of the shared space.

Google Calendar API
Room8's scheduler requires a calendar component where user's can book and view events and chores in a calendar Google's Calendar API.

Secure Database
Users create accounts for Room8 where user and home information is stored on a secure database where sensitive information such as credentials are stored and individuals are able to change and delete accounts.

Gmail OAuth
Users will sign in securely without risk of data leaks and malicious connections.

SMS Message Service
Room8's Chatbot SMS will send messages to users using a third party messaging system that has connection to the Room8 application to send warnings and information of shared space.

P.6 Risks and mitigation analysis

Comment: *Potential obstacles to meeting the schedule of P.4, and measures for adapting the plan if they do arise.*

Machine Learning Model Training
Finding enough data to train the machine learning algorithm to detect when a mess has been made and what detect what is altered or added in the space. Sensitivity to slight changes such as lighting is another risk with the machine learning algorithm ap-

proach. An alternative to adapt this scenario would be to use image differencing to detect changes in environment.

User Identification
Being able to detect which user is using the shared space through the camera is a major risk. The program may not be supplied with enough information to identity which user is currently using the space. In order to work around this the program could get individuals to submit to the application that they are now using the space.

Scope Creep
Unclear requirements and adding too many elements to the scope of the project will result in not delivering on features that are listed to be part of the application and lower quality of core functionality due to time being strain. A revisal of requirements by making requirements and functionality of system clear and concise and developing a prioritization list would mitigate the risk of scope creep.

P.7 Requirements process and report

Comment: *Initially, description of what the requirements process will be; later, report on its steps.*

Appendix — Reflection

The information in this section will be used to evaluate the team members on the graduate attribute of Lifelong Learning.

1. What went well while writing this deliverable?
2. What pain points did you experience during this deliverable, and how did you resolve them?
3. How many of your requirements were inspired by speaking to your client(s) or their proxies (e.g. your peers, stakeholders, potential users)?
4. Which of the courses you have taken, or are currently taking, will help your team to be successful with your capstone project.
5. What knowledge and skills will the team collectively need to acquire to successfully complete this capstone project? Examples of possible knowledge to acquire include domain specific knowledge from the domain of your application, or software engineering knowledge, mechatronics knowledge or computer science knowledge. Skills may be related to technology, or writing, or presentation, or team management, etc. You should look to identify at least one item for each team member.
6. For each of the knowledge areas and skills identified in the previous question, what are at least two approaches to acquiring the knowledge or mastering the skill? Of the identified approaches, which will each team member pursue, and why did they make this choice?