Department of Engineering and Computer Science

SOEN 390

The Condo Management Company

Compiled Report

Submitted To:

Professor Jinqiu Yang

Submitted By:

Camila-Paz Vejar-Rojas (ID: 40208489)

Andy Chhuon (ID: 40199798)

Gulnoor Kaur (ID: 40114998)

Amar Mahmoud (ID: 40168218)

Jonah Ball (ID: 40178421)

Mohammad Usman (ID: 40170784)

Nusrath Zaman (ID: 40123819)

Mark Tadros (ID: 40250850)

James Bitharas (ID: 26637175)

Submitted On:

May 1, 2024

Table of Contents

1. Product Vision Statement

1.1. Introduction

This vision document outlines the development of a comprehensive condo management system designed to streamline and enhance the operational efficiency of condo management, improve the living experience for condo residents, and foster stronger community engagement within condominium complexes. The system aims to address the needs of condo owners, tenants, and condo management companies by providing a unified platform for managing profiles, properties, financial transactions, facility reservations, and communication. By integrating key functionalities into a user-friendly interface accessible across multiple platforms, the project seeks to modernize condo management practices, ensuring that all stakeholders have easy access to critical information and services. The ultimate goal is to create a scalable, secure, and inclusive solution that meets the diverse needs of condominium communities.

1.2. Positioning

1.2.1. Problem Statement

The problem of	inefficient and fragmented condo management processes
affects	condo owners, tenants, and condo management companies
the impact of which is	increased administrative overhead, poor communication, and dissatisfaction among residents and property managers
a successful solution would be	a unified, intuitive platform that streamlines property management tasks, enhances communication and engagement within the condo community, simplifies financial management, and provides easy access to property-related information and services, thereby improving operational efficiency, resident satisfaction, and community well-being.

1.2.2. Product Position Statement

For	condo management companies, condo owners, and tenants
Who	need a comprehensive and integrated solution to manage their properties and enhance their living experience
The Condo Management System	is a property management platform
That	simplifies the complexities of condo administration, financial management, and community engagement
Unlike	fragmented and less user-friendly solutions currently dominating the market
Our product	provides a seamless, all-in-one platform designed to be accessible on multiple devices, offering intuitive user interfaces and features tailored specifically for the condo management ecosystem. Our product stands out by ensuring easy access to property information, financial data, and community resources, fostering a more connected and efficiently managed living environment

1.3. Stakeholder and User Descriptions

1.3.1. Stakeholder Summary

Name	Description	Responsibilities
Condo Management Companies	Organizations responsible for the day-to-day operations and management of condominium complexes.	● Ensure the condo management system meets the operational needs of property management, including maintenance, financial management, and communication with residents. ● Monitor the project's progress to ensure that the platform integrates seamlessly with existing processes and provides a return on investment.

Property Developers	Companies or individuals involved in the construction and sale of new condo developments.	● Interested in the system's ability to attract buyers by offering an advanced management solution, thereby increasing the marketability of their properties. ● Ensure that the system supports the transition from developer to owner-managed properties.
Investors	Individuals or entities that provide the capital necessary for the development of the condo management system.	● Ensure a return on investment through the successful adoption and operation of the system. ● Monitors financial performance and market penetration.
Regulatory Bodies	Government or industry organizations that set standards and regulations for property management and tenant rights.	● Ensures that the system complies with legal and regulatory requirements, including data protection and privacy laws.

1.3.2. User Summary

Name	**Description**	**Responsibilities**	**Stakeholder**
Condo Owners	Individuals who own condominium units.	● Use the system to manage their property information, financial transactions (e.g., condo fees), and access community features. ● Captures details related to their units and produces reports on financial status.	Represented by condo management companies and owners' associations to ensure the system meets their specific needs for property management and investment monitoring.
Renters	Individuals renting units within a condominium complex.	● Use the system for communication with management, access to amenities, and management of rental details. ● Provides feedback on the system's usability and functionality regarding tenant needs.	Condo owners and management companies represent tenants' interests, ensuring features like reservation systems and maintenance requests are user-friendly and efficient.
Employees	Employees or contractors	● Utilize the system for managing property	Condo management companies employ property

	responsible for the day-to-day operation of a condominium complex.	maintenance, tenant inquiries, and financial management. ● Captures details regarding maintenance requests, tenant interactions, and operational expenses.	managers, ensuring the system supports efficient property management practices.
Condo Management Companies	Organizations responsible for overseeing the management and operation of condominium complexes	● Oversee the entire management process, including financial management, property maintenance, and communication between all parties involved. ● Ensure compliance with laws and regulations. Implement and manage the condo management system to streamline operations.	Directly responsible for ensuring the system meets the operational needs and legal requirements of managing a condominium complex, serving as the primary administrators of the system.
Public Users	Individuals who are not condo owners, renters, or employees but are interested in the condominium complex or its amenities.	● May use the system to explore condominium amenities, availability for rent or purchase, and general information.	While not directly represented by any specific stakeholder group, their interests are considered by condo management companies to ensure the system is appealing and accessible to potential future owners or tenants.

1.3.3. User Environment

Condo Owners and Tenants

- Number of People Involved: Typically, tasks are completed individually, such as paying condo fees, booking amenities, or submitting maintenance requests. Family members or roommates may also be involved in decision-making or using the system's features.

- Task Cycle: The task cycle can range from a few minutes for booking amenities or submitting requests to longer periods for financial transactions or document review. The amount of time spent on each activity varies based on the complexity of the task.

- Environmental Constraints: Users may access the system in various environments, including at home, at work, or on the go, necessitating a mobile-responsive design that accommodates different devices and connectivity conditions.

- System Platforms: Currently, users might rely on a mix of emails, phone calls, and physical meetings for condo management. Future platforms include web and mobile applications accessible on Android, iOS, and web browsers.

- Integration Needs: The application may need to integrate with payment gateways for financial transactions, email systems for notifications, and possibly other community tools like forums or social networks.

Property Managers

- Number of People Involved: Property managers typically coordinate with a team, including condo management companies and service providers, to manage the property effectively.

- Task Cycle: Property managers engage in a variety of tasks ranging from quick administrative duties to longer-term planning and resident communication, requiring flexibility in time management and system support.

- Environmental Constraints: The work environment can include office settings, on-site locations within the condo complex, and mobile scenarios, necessitating access to the system through both mobile and desktop platforms.

- System Platforms: Existing systems might include specialized property management software and financial systems. The new system needs to support integration with these platforms and be accessible on future mobile and desktop devices.

- Integration Needs: Essential integrations include connecting with existing property management tools, financial software, and possibly CRM systems to ensure streamlined operations and accurate maintenance of records and communications.

1.3.4. Key Stakeholder or User Needs

1.3.5. Alternatives and Competition

Based on the information available and some knowledge of the condo management software market, here is a short list of alternatives and competitive choices, including their perceived strengths and weaknesses:

DoorLoop

- Strengths: Offers a comprehensive feature set including a central dashboard, work order management, built-in CRM and tenant portal, robust accounting, and community management tools. It's known for streamlining tasks and enhancing communication.
- Weaknesses: Pricing may require contacting the vendor for a demo, which could delay immediate access and comparison.

Wild Apricot

- Strengths: Great for small community associations with features for event management, committee management, and various payment processing options. It's also praised for its ease of use.
- Weaknesses: Lacks calendar management and work order management features, which may limit its utility for some associations.

PayHOA

- Strengths: Known for excellent customer support and a full range of HOA management tools. It also includes a comprehensive database for members and properties.
- Weaknesses: The absence of mobile app functionality might limit accessibility for users preferring mobile management.

Condo Control

- Strengths: Offers a mix of property management and unique HOA community management features, including security and concierge features, community management tools, and event management.
- Weaknesses: May require contacting the company for pricing and demo, which can be a barrier for quick evaluation.

Yardi Breeze

- Strengths: A versatile platform suitable for a wide range of property management needs, including condos and HOAs. It's designed to automate tedious tasks and simplify property management.
- Weaknesses: The platform might offer more features than needed for smaller associations, potentially complicating its use.

Buildium

- Strengths: Suitable for HOA boards with features for resident screening, online leasing, and tax record maintenance. It also supports property management companies in promoting their services.
- Weaknesses: Some users may find the platform too comprehensive, leading to a steeper learning curve.

Homegrown Solutions

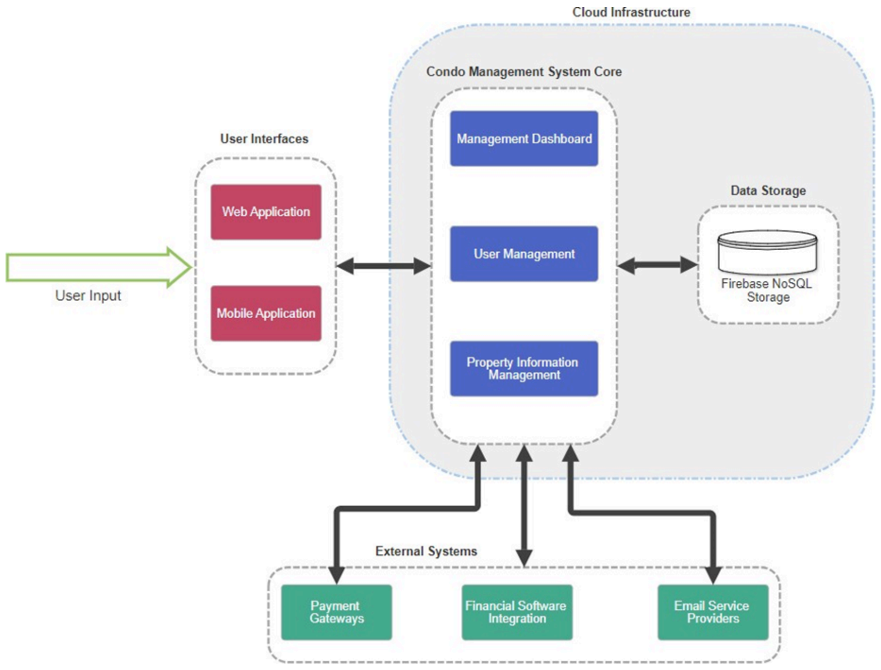
- Strengths: Customizable to specific needs and potentially lower upfront costs. Can be tailored to fit the unique workflows of an association.
- Weaknesses: Requires significant time and resources for development and maintenance. May lack scalability and robustness compared to commercial products.

Maintaining the Status Quo

- Strengths: Familiarity for users and no need to adapt to new systems. No immediate costs associated with transitioning to new software.
- Weaknesses: Inefficiency, increased risk of errors, and inability to meet modern expectations for management and resident engagement.

1.4. Product Overview

1.4.1. Product Perspective



1.4.2. Assumptions and Dependencies

- **Operating Systems Availability:** The assumption is that users will have access to modern operating systems on mobile and desktop platforms (iOS, Android, Windows, macOS, Linux) that can support web and potentially native applications of the condo management system.
- **Internet Connectivity:** The system assumes reliable internet connectivity for users to access cloud-based functionalities and for real-time data synchronization.
- **External System Integration:** It's assumed that third-party systems such as payment gateways, email service providers, and financial software offer stable APIs for integration and maintain compatibility with the system.
- **Compliance and Legal Requirements:** The development and deployment of the system are based on the assumption that it will comply with all relevant data protection and privacy laws (e.g., GDPR, CCPA) as of the launch. Changes in legal requirements could necessitate system updates.
- **Cloud Infrastructure:** The system assumes the availability and reliability of cloud infrastructure (AWS) for hosting the application, ensuring scalability, and managing data securely.
- **User Adoption:** There's an underlying assumption that users (condo owners, tenants, management companies) will be receptive to adopting a digital solution for condo management, transitioning from manual or less integrated systems.

Changes to any of these assumptions could significantly impact the vision, design, development, and deployment of the condo management system, necessitating a revisit to the vision document and potentially altering the product scope or functionality.

Need	**Priority**	**Concerns**	**Current Solution**	**Proposed Solutions**
Efficient Property Management	High	Current systems are fragmented, leading to inefficient management of condo properties, including maintenance requests, financial transactions, and communication with residents.	Manual processes, multiple software solutions with limited integration, and physical paperwork.	An integrated platform that consolidates property management features, including a dashboard for real-time updates on maintenance, finances, and resident communications.
Effective Communication Channels	High	Lack of effective and direct communication channels between condo	Emails, phone calls, and physical notices which are not	A built-in messaging system, notification features, and a

		owners, tenants, and property managers, leading to delays and misunderstandings.	always efficient or timely.	community forum within the platform to facilitate timely and clear communications.
Transparent Financial Management	Medium	Difficulty in tracking and managing financial transactions such as condo fees, maintenance costs, and budget planning.	Separate financial software, spreadsheets, and manual record-keeping.	Integrated financial management tools within the system that offer transparent tracking of fees, payments, and budgets with reports and alerts.
Convenient Booking of Amenities	Medium	Inefficient reservation system for amenities, leading to conflicts and underutilization.	Manual booking through property management, physical sign-up sheets, or separate booking systems.	A digital reservation system embedded in the platform, with a calendar view, availability checks, and reservation management.
Mobile Access and Usability	High	Current systems may not be accessible or optimized for mobile devices, limiting access for users on the go.	Desktop-based systems or non-optimized mobile websites.	A mobile-responsive design or dedicated mobile apps that ensure full functionality across devices and platforms.
Integration with Existing Systems	Medium	The need to maintain data consistency and workflow continuity with existing external systems.	Manual data transfer or limited API integrations.	Robust API support for seamless integration with popular payment gateways, financial software, and other property management tools.
User-Friendly Interface	High	Users often find existing systems complex and not intuitive, leading to underutilization and frustration.	Standard web interfaces with limited consideration for user experience design principles.	Implementing a modern, clean, and intuitive UI/UX design, with user feedback loops to continually refine the interface.
Real-Time Notifications and Updates	Medium	Delayed notifications about important updates, events, and maintenance schedules can lead to missed opportunities and dissatisfaction.	Email blasts and physical bulletin boards.	Push notifications through the platform for instant updates on events, maintenance, and financial matters.
Multi-	Medium	Condo communities are	Limited to the	Offering multi-language

Language Support		increasingly diverse, and language barriers can hinder access to information and services.	primary language of the region, with ad hoc translations as needed.	support within the platform to cater to a diverse resident population.
Secure Data Management	High	Concerns over the privacy and security of personal and financial data within the condo management system.	Basic security measures, often lacking comprehensive data protection strategies.	Implementing robust security protocols, including encryption, secure login mechanisms, and compliance with privacy laws.
Scalability	Medium	As condo complexes grow and more units are managed, the system must be able to scale without performance degradation	Static systems that require significant overhaul to support more users or functionalities.	Cloud-based architecture that allows for easy scaling and updates as the user base grows and needs evolve.
Customization Capabilities	Medium	Different condo management companies and communities have unique needs that may not be met by a one-size-fits-all solution.	Limited customization options, leading to workaround solutions or underutilization.	Allowing for modular customization of features and interfaces to meet specific community needs.

1.5. Product Features

Feature Name	Description	Priority	Justification
User Profile Management	Allows users to create and manage their profiles with essential details.	High	Fundamental for identification, personalization, and facilitating access to services.
Registration and Access Control	Secure registration process for users to access specific roles and functionalities.	High	Ensures secure access and appropriate user-role association.
Property and User Dashboard	Central dashboard displaying key information for condo owners and management.	High	Centralizes crucial information for easy access, enhancing user experience.
Property Profile Management	Creation and management of detailed property profiles by	High	Organizes and makes property information

	condo management companies.		readily accessible.
Document Management	Uploading, storing, and sharing of condo-related documents for access by condo owners.	High	Supports transparency and informed decision-making by ensuring document accessibility.
Financial Management System	Simplified module for managing finances, including condo fees and budgeting.	High	Critical for financial oversight and ensuring financial responsibilities are met.
Reservation System	System for booking and managing reservations for common facilities.	Medium	Improves the utilization of amenities and resident satisfaction.
Maintenance and Request Tracking	Allows users to submit and track requests, assigning them to appropriate staff.	Medium	Streamlines handling of requests and maintenance issues, ensuring prompt action.
Notification System	Updates and notifications for users regarding activities, requests, and information.	Medium	Enhances communication by keeping users informed of relevant updates.
Community Engagement Features	Forums, event tools, and sharing of discounts or offers to foster community interaction.	Low	Encourages community building, although not critical to core management functionalities.
Cross-Platform Accessibility	Ensures the app's accessibility across different devices and operating systems.	High	Vital for user accessibility and accommodating diverse user preferences.
Multilingual Support	Providing support in multiple languages to cater to a diverse user base.	Medium	Promotes inclusivity and increases accessibility for non-English speakers.
Single Sign-On (SSO) Integration	Facilitates easy and secure access through integration with existing accounts like Gmail.	Medium	Streamlines the login process, enhancing user convenience and security.

1.6. Other Product Requirements

Requirement Category	**Description**	**Priority**	**Rationale**

Standards and Compliance	Adherence to data protection, privacy laws, accessibility standards, and condominium management regulations.	High	Mandatory for legal use and user data protection.
Hardware/Platform Requirements	Accessibility on multiple platforms, including web browsers on PCs and mobile devices without high-end hardware requirements.	High	Ensures wide usability and accessibility.
Performance Requirements	Support for thousands of concurrent users with a response time not exceeding 2 seconds.	High	Key for user satisfaction and scalability.
Environmental Requirements	Operation in cloud environments for scalability and resilience, ensuring 24/7 availability.	Medium	Ensures availability and leverages cloud infrastructure advantages.
Usability and Accessibility	Intuitive UI/UX compliant with accessibility standards, including onboarding tutorials and clear navigation.	High	Reduces learning curve and enhances user engagement.
Fault Tolerance and Robustness	Error handling and recovery mechanisms for continuous operation.	High	Ensures system reliability and trustworthiness.
Documentation Requirements	Provision of comprehensive user manuals, online help, installation guides, and FAQs, available in multiple languages.	Medium	Supports user self-service and enhances satisfaction.
Security Requirements	Industry-standard security practices, including data encryption, regular audits, and user authentication (with 2FA options).	High	Critical for protecting user data and system integrity.
Integration Requirements	APIs for integration with external systems such as payment gateways and other property management tools.	Medium	Enhances system functionality and integration with existing digital ecosystems.
Environmental Sustainability	Design considerations for minimizing energy consumption and optimizing resource use, especially in	Low	Aligns with environmental goals and corporate social responsibility.

	cloud environments.		

2. Requirements and User Stories Backlog

1.User Profiles:
● Public users can create profiles with a profile picture, username, contact email, and phone number.
● Users must obtain a registration key from their condo management company to become condo owners or rental users.
2.Property Management:
● Condo management companies can create profiles for properties, including name, unit count, parking count, locker count, and address.
3.File Management:
● Management companies can upload and manage condo files accessible to all condo owners, such as declarations, budgets, and meeting minutes.
4.Unit Information Management:
● Management companies can enter detailed information for each condo unit, parking spot, and locker, including unit id, size, owner, occupant information, and associated condo fees.
5.Financial System:
● Condo management companies can set condo fees per square foot and parking spot, with automatic calculations for each unit.
● Recording operational budget, costs, and generating annual reports.
6.Reservation System:
● Set up common facilities for reservations.
● Calendar-like interface for booking facilities with availability display.
● First-come-first-serve system for reservations.
7.Role Management:
● Set up different roles for employees, such as managers and finance personnel.
8.Request Management:
● Owners can submit various requests such as moving in/out scheduling, intercom changes, access requests, violation reports, and maintenance issues.
● Requests are assigned to corresponding employees with a notification system for users.
9.Community Features:
● Forum for users to post and reply.
● Event organization and invitations.
● Discounts and offers listing for property occupants.
10.Cross-Platform Accessibility:
● The app should be accessible on Android, iOS, Linux, MacOS, and Windows platforms.

11.Multilingual Support:
● The app should be available in English and at least one additional language.
12.Single Sign-On (SSO):
● Users should have the option to log in using their Gmail accounts or other Single Sign-On methods.

1		User Registration	
As a public user, I want to create a unique profile with profile picture, user name, contact email, and phone number, because I need access to the condo management app and website.			
User Management			
M	L	L	2

2		Registration Key for Condo	
As a public user, I want to input a registration key from the condo management company to become a condo owner in the system, because I want accurate ownership representation.			
User Management			
M	M	L	2

3		Registration Key for Rental	
As a public user, I want to input a registration key from the condo management company to become a rental user in the system, so that I can access rental-specific features.			
User Management			
M	M	L	2

4		Access to Condo Owner Dashboard	
As a condo owner, I want to access a dashboard with general information about my properties, including personal profile details, condo information, and financial status (i.e.remaining balance in terms of monthly condo fee payments), status of the submitted request because I need to stay informed about my property's status and financial obligations.			
User Management			
M	L	M	3

5		Property Profile Creation	
As a condo management company, I want to create a profile for each property under my management, providing essential details such as property name, unit count, parking count, locker count, and address, to maintain organized property records within the system			
Property Management			
M	L	M	3

6		Uploading Condo Files	
As a condo management company, I want to upload condo files for each property, including condo declarations, annual budgets, and board meeting minutes, ensuring accessibility to all condo owners for transparency and communication purposes.			
Property Management			
S	M	M	3

7		Entering Detailed Information for Condo Units	
As a condo management company, I want to enter detailed information for each condo unit, parking spot, and locker in a building, including unit ID, size, owner details, occupant information, and associated condo fees, to maintain accurate records and facilitate efficient management of the property.			
Property Management			
M	L	M	3

8		Sending Registration Keys to Users	
As a condo management company, I want to send registration keys to unit owners or rental users for their dedicated units, because it links their condo units with their profiles within the system, ensuring accurate representation of ownership and occupancy status.			
User Management			
S	M	M	3

9		Entering Condo Fees in Financial System	

As a condo management company, I want to enter condo fees per square foot and per parking spot into the financial system, ensuring accurate calculation of fees for each unit.			
Financial Management			
M	L	M	3

10		Automatic Calculation of Condo Fees	
As a condo management company, I want the financial system to automatically calculate and present the condo fee for each unit, based on the entered fee per square foot and per parking spot, providing transparency to unit owners regarding their financial obligations.			
Financial Management			
M	L	M	5

11		Recording Operational Budget Details	
As a condo management company, I want to record operational budget details, including collected condo fees and costs for various operations, within the financial system, ensuring accurate financial tracking and management.			
Financial Management			
M	L	M	5

12		Entering Costs for Each Operation	
As a condo management company, I want to enter the cost for each operation conducted within the property into the financial system, because I want comprehensive tracking of expenses and budget allocation.			
Financial Management			
M	L	M	5

13		Generating Annual Financial Report	
As a condo management company, I want the financial system to generate an annual report summarizing all condo fees collected for a given year, providing stakeholders with a clear overview of financial performance and expenses.			
Financial Management			
M	L	M	5

14		Setting Up Common Facilities	
As a condo management company, I want to set up common facilities such as a sky lounge and spa fitness within the reservation system, ensuring that residents have access to these amenities for their enjoyment and convenience.			
Reservation System			
M	L	M	5

15		Providing Calendar-Like Interface	
As a condo management company, I want the reservation system to provide a calendar-like interface for users to view and select available time slots for common facilities, allowing residents to easily book their desired times.			
Reservation System			
M	L	M	5

16		Displaying Availability of Common Facilities	
As a condo management company, I want the reservation system to display the availability of common facilities in real-time, enabling residents to see which time slots are open for booking and avoiding confusion.			
Reservation System			
M	L	M	5

17		First-Come-First-Serve Booking	

As a condo management company, I want the reservation system to operate on a first-come-first-serve basis, ensuring fairness in booking common facilities for all residents.			
Reservation System			
M	L	M	5

18	Automatic Marking of Facilities as Booked		
As a condo management company, I want the reservation system to automatically mark a facility as unavailable once it has been booked by a user, preventing double bookings and ensuring exclusive use of the facility during the reserved time slot			
Reservation System			
M	L	M	5

19	Setting Up Roles for Employees		
As a condo management company, I want to set up different roles for employees responsible for the same property, such as managers for daily operations and finance personnel, to ensure efficient management of tasks and responsibilities.			
Role Management			
M	L	M	1

20	Submitting Requests as Condo Owners		
As a condo management company, I want to submit requests for various purposes such as moving in/out, intercom changes, access requests, reporting violations, reporting deficiencies, or asking questions, to address issues or concerns related to the property.			
Request Management			
M	L	M	2

21	Assignment of Requests to Corresponding Employees		
As a condo management company, I want each request submitted by a condo owner to be assigned to a corresponding employee based on the type of request, ensuring that it is addressed by the appropriate personnel in a timely manner.			
Request Management			
M	L	M	2

22		Notification Page for Users	
As a user, I want a notification page where I can see the latest activities in submitted or assigned requests, keeping me informed about the status of my requests and any updates or actions taken by the management team.			
Request Management			
M	L	L	2

23		Implementing Forum Feature	
As a condo management company, I want to implement a forum where users can post and reply, allowing for community engagement and communication among residents.			
Community Features			
C	M	M	5

24		Organizing Events Feature	
As a user, I want the ability to organize events and invite other occupants to attend, facilitating community-building and social interaction among residents.			
Community Features			
C	M	M	5

25		Listing Coupons and Offers	
As a condo management company, I want to list coupons and offers visible to all unit owners or rental users of a property, providing them with discounts and incentives to enhance resident satisfaction.			
Community Features			
C	M	M	5

3. Software Architecture Document

3.1. Introduction

Many condos have property managers who manage the community's daily operations manually. They are responsible for managing finances, responding to customer complaints, managing documents and records, liaising between residents and board, etc. Some of these tasks are still done on paper and may not have a proper storage for documenting records. Thus, the need arises for a Condo Management System to automate these tasks and a proper storage of records making it simpler for both the condo owners/renters and managers to communicate and view their required information online.

In this report, our team has designed a software architecture for a mobile app/website for the condos in Montreal, Quebec, Canada. The main goal of this platform is to simplify daily community operations by automating processes such as payments, storing data, providing analytics and insights.

3.2. Scope

Considering the available resources, duration of the course and available human resources of our team, we have decided to constrain the scope of the project to all the condos in Montreal. As we progress further, In the future we may consider expanding to the whole of Canada.

3.3. Stakeholders and Concerns

This section identifies the stakeholders, the individuals who will be accessing the system, and the concerns related to the system. The last subsection will be mapping the stakeholders to the concerns they have.

3.3.1. Stakeholders

The stakeholders we have identified are listed below with a brief description about them,

1. **Users**
   - **Condo Owners:** An individual who owns a condo in the system
   - **Condo Renters:** An individual who rents a condo in the system
   - **Condo Management Company:** An organization that has property's under its management
   - **Admin:** An individual responsible for responding to user requests
2. **Software Provider Organization**

    - **Project Manager:** An individual who is responsible for the planning and execution of the project
    - **Developer:** A group of individuals who develops software
    - **Tester:** An individual who tests the software built
3. **Property Developers:** Companies or individuals involved in the construction and sale of new condo developments.
4. **Investors:** An individual or entity who provides funds to the Project.
5. **Regulatory Bodies:** Government or industry organizations that set standards and regulations for property management and tenant rights.

3.3.2. Concerns

Some of the concerns considered fundamental to the architecture of Condo Management System have been listed below:

- **Concern 1:** Is all the information entered during registration validated?
- **Concern 2:** Is the system protected from unauthorized access?
- **Concern 3:** Is the system easy to operate and navigate?
- **Concern 4:** Is the correct information made visible to the correct user?
- **Concern 5:** Is it easy to get financial reports from the system?
- **Concern 6:** Is the analytics provided by the financial system accurate?
- **Concern 7:** Is it easy to navigate to and reserve a facility?
- **Concern 8:** Does the reservation system allow a single facility to be reserved at a point in time?
- **Concern 9:** Is the system quick and responsive?
- **Concern 10:** Will the software be delivered on time and within the budget?
- **Concern 11:** Is there enough skilled resources to develop the system?
- **Concern 12:** Which methodology is going to be used for the development? (Agile, waterfall, etc.)
- **Concern 13:** Is the system going to be easily modified?
- **Concern 14:** What is the environment for our system? (Hardware, Operating System)
- **Concern 15:** Are the features of the system able to interact with each other?
- **Concern 16:** Is the code easy to test?
- **Concern 17:** Are all the available resources being properly utilized?
- **Concern 18:** Cost of the System?
- **Concern 19:** Are the rules and regulation set being followed?
- **Concern 20:** Is it easy to verify information of the users?
- **Concern 21:** Is it easy to post requests on the system?

3.3.3. Concern-Stakeholder Traceability

	Condo Owner	Condo Renter	Condo Mgmt. Co.	Admin	Project Manage r	Dev.	Tester	Property Developer	Investor	Regulatory Bodies
Concern 1	X	X	X	X	-	-	-	-	-	-
Concern 2	X	X	X	X	-	-	-	-	X	-
Concern 3	X	X	X	X	-	-	-	-	-	-
Concern 4	X	X	X	X	-	-	X	-	-	-
Concern 5	X	-	X	-	-	-	-	-	-	-
Concern 6	X	-	X	-	-	-	X	-	-	-
Concern 7	X	X	-	-	-	-	-	-	-	-
Concern 8	-	-	X	-	-	-	X	-	-	-
Concern 9	X	X	X	X	-	-	-	-	-	-
Concern 10	-	-	-	-	X	-	-	-	X	-
Concern 11	-	-	-	-	X	-	-	-	-	-
Concern 12	-	-	-	-	X	-	-	-	-	-
Concern 13	-	-	-	-	-	X	-	-	-	-
Concern 14	-	-	-	-	-	X	-	-	-	-
Concern 15	-	-	-	-	-	X	-	-	-	-
Concern 16	-	-	-	-	-	-	X	-	-	-
Concern 17	-	-	-	-	-	-	-	-	X	-
Concern 18	-	-	-	-	X	-	-	-	X	-
Concern 19	-	-	-	-	-	-	-	-	-	X
Concern 20	-	-	-	-	-	-	-	-	-	X
Concern 21	X	X	-	-	-	-	-	-	-	-

Table 1: Concern-Stakeholder Traceability

3.4. Viewpoints and Views

We have used the Unified Modelling Language for describing our architecture design of the Condo Management System. The UML model helps system designers and developers visualize the structure of their system at a high-level and ensure that the application meets the user's needs.

3.4.1. Domain Model

This model is a real-world visualization of the system and it is derived from the project requirements. It is a visual representation of interconnected concepts of real-world objects that incorporates key concepts, behavior and relationships of all of its entities.

Based on the requirements, we found out the following users for our system: owners, renters, companies, employees. Each of them have been represented in the diagram below with their respective collections as we will store records of them. We also identified the need to have facilities and properties as components, as the companies can upload properties/facilities and owners/renters can buy/rent and reserve facilities. Thus in the diagram we have Facility and Property components with their collections.

Since we know from requirements that a owner/renter can buy/rent properties we showed this relationship with a directional arrow with the appropriate text above it. Similarly for facilities, the relationship has been shown with owner/renter using an arr directional arrow with appropriate text. The relationship between companies and employees is shown with the arrow named “sets” in the diagram. The relation between company and property is also shown as companies can add properties that are available to buy/rent.

From the requirements we know that the users have the ability to send requests to employees set by the company to respond to their request. This concept is shown between owner/renters and employees with the arrow labeled “messages” which is connected to the Messenger component. As we will also have notifications for users, the Notification component uses messenger and is connected to the respective users.

The system also has a financial system. This system includes a parent type financial calculator component that calculates the finances for users. This component depends on individual tax, condo fees and any extra expense calculator to calculate the total. This component is used by owner/renters and requires condo companies. This relation can also be seen in the diagram below.

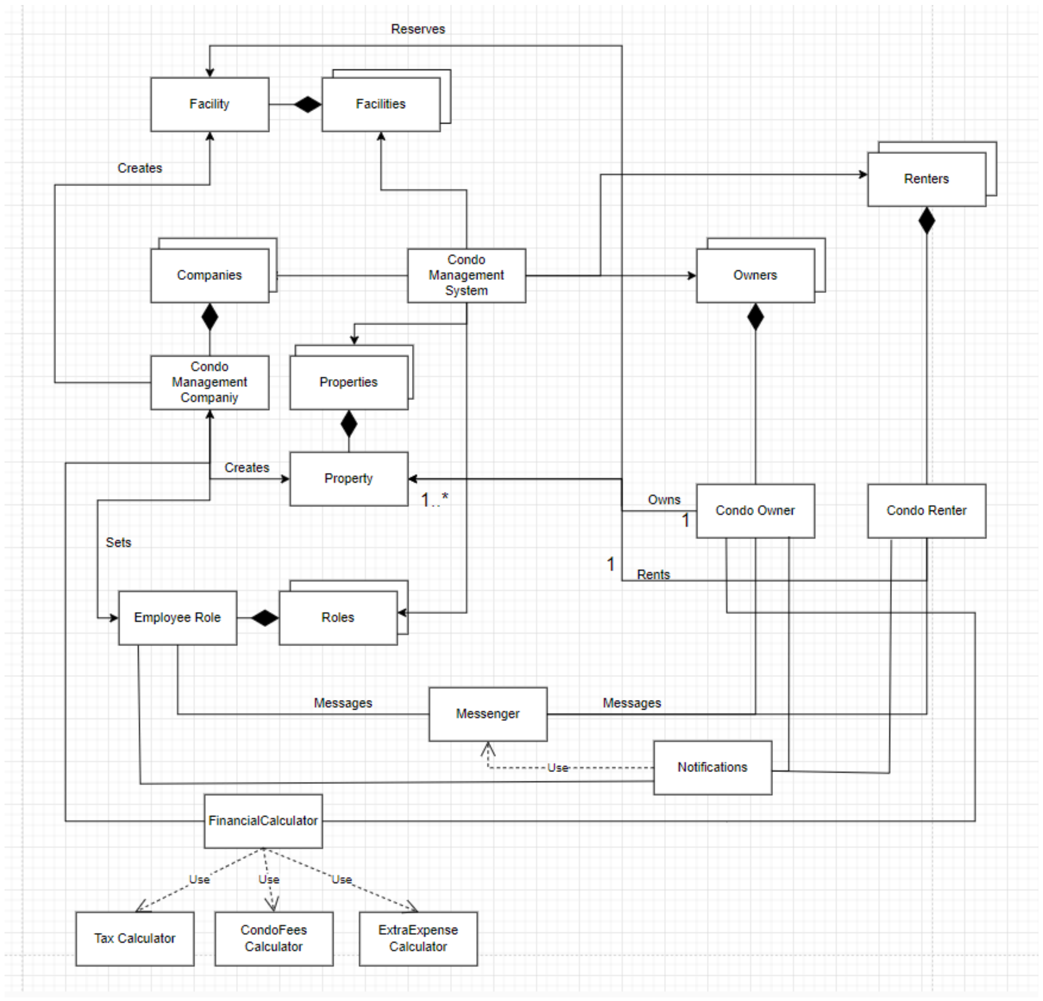


Figure 1: Domain Model of Condo Management System

3.4.2. Component Model

A component model, also known as a UML component diagram, describes the organization and wiring of the physical components in a system. It shows the structure of the software system, which describes the software components, their interfaces, and their dependencies.

The below component diagram for the Condo Management System is divided into four subsystems namely, the webInterface(UI), the messaging system, reservation system, financial system and the database.

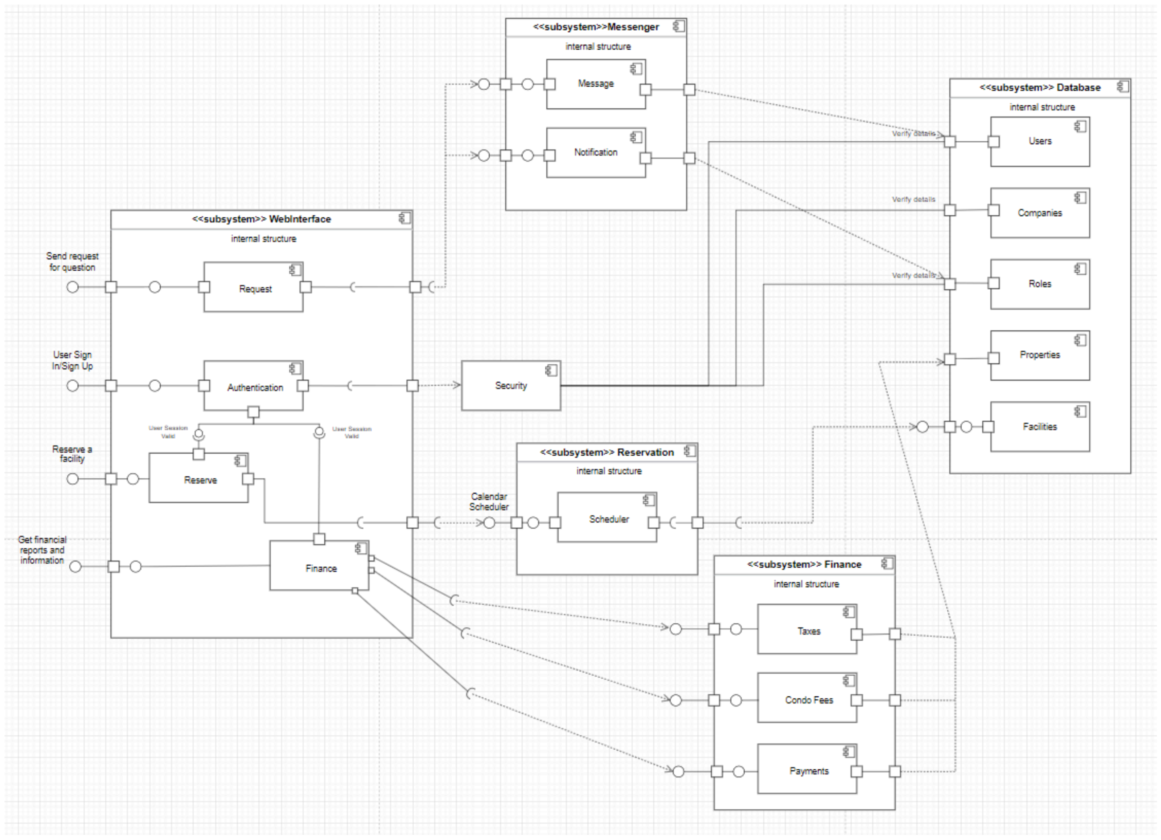


Figure 2: Component Model of Condo Management System

3.4.3. Class Diagram

Based on the domain model that was designed for our system, we have developed the following class diagram. It has all the classes identified in the domain diagram and the systems involved have been separated into classes to provide more details into its functioning. Using the GRASP principles we have identified the following patterns in our class diagram,

- The CondoManagementController class is the Controller Pattern. Since all the functionalities will be handled by this class. It also has the Information Expert pattern as it has access/copies of all users, companies, properties, facilities, etc.
- CondoManagementController class also has the Indirection pattern as it acts as an intermediary between UI and classes of the system.

- The FinanceCalculator class and Message class support the High Cohesion Pattern. We can have information regarding the messages received/sent and financial info of properties inside the CondoOwner/CondoRenter class, but then these classes will be focusing on different functions and will lead to low cohesion. Thus, these classes were created for them to focus on separate tasks.
- Owners, Renters, Companies, Facilities, Properties, Roles have the Creator pattern with regards to CondoOwner, CondoRenter, CondoManagementCompany, Facility, Property, Role classes respectively.

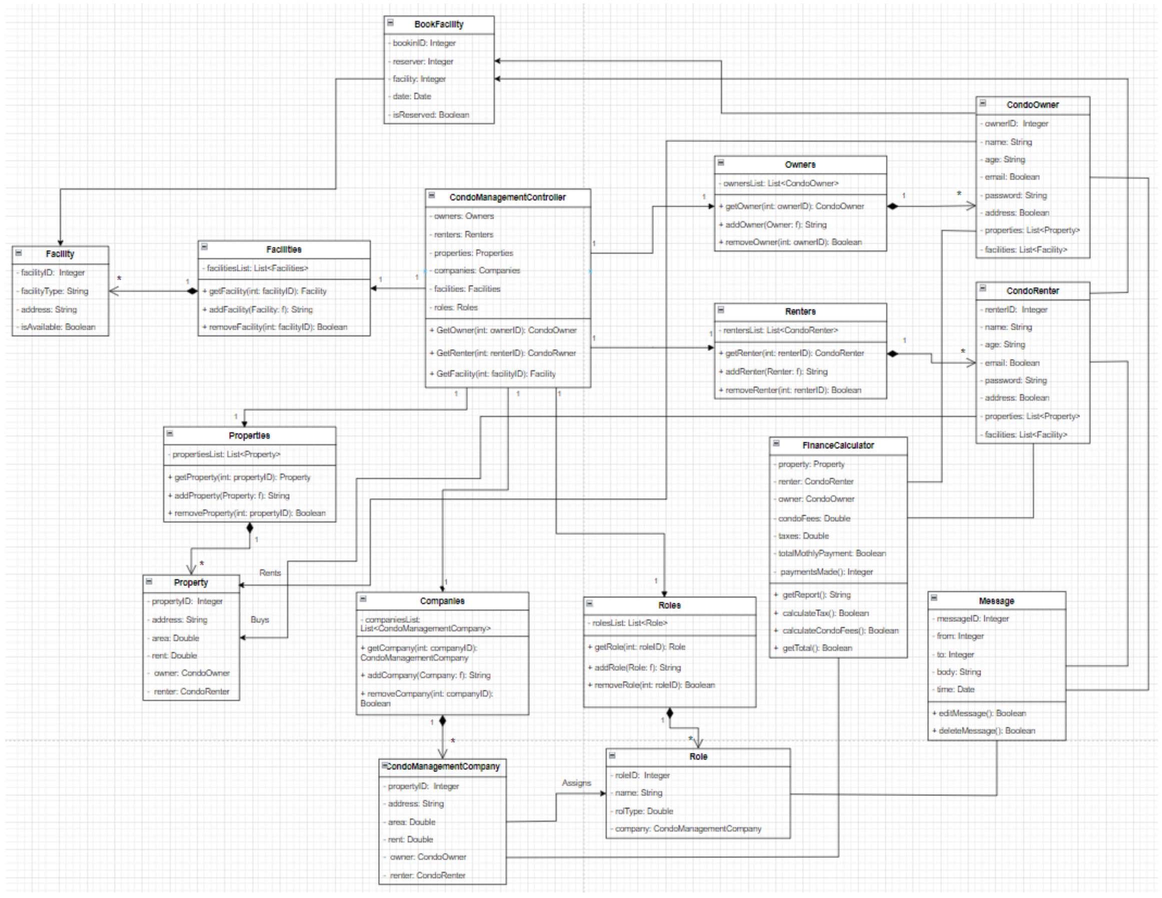


Figure 3: Class Model of Condo Management System

3.4.4. Deployment Diagram

The diagram below shows the deployment diagram for our system. We identified three nodes for our system. The first node is for our user i.e their devices on which they will access our application/system using a web browser/app which is shown as a sub-node.

The second node represents the web server for our application. Since for our backend we used node.js and express.js, these are shown as an executable environment node in our web server node. This node includes the logic for the different functions that our system will be performing like the reservation, financial, etc. logic. The user's browser will be able to connect to this server using the HTTP protocol.

The third node represents our database server. The database we used for this system is Firebase, a real-time database. We used the cylinder shape which is used in UML to represent the database. The individual tables are also represented within the database. The web server connects to the database with the help of an API provided. This connection is made by importing the sdk into the application using express.js.

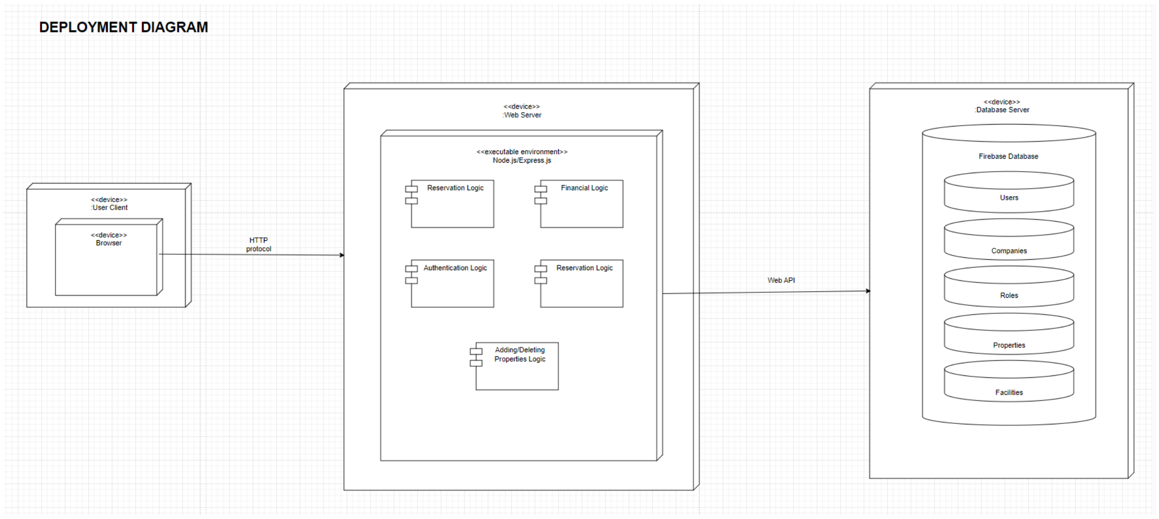


Figure 4: Deployment Diagram for Condo Management System

3.4.5. Use Case Diagram

In UML, Use Case diagrams are used to represent high-level functions and scope of a system. This diagram also identifies the interaction between the system and its actors. The use cases and the actors involved in the use cases describe what the system does and how the actors use it, without telling how the system works internally.

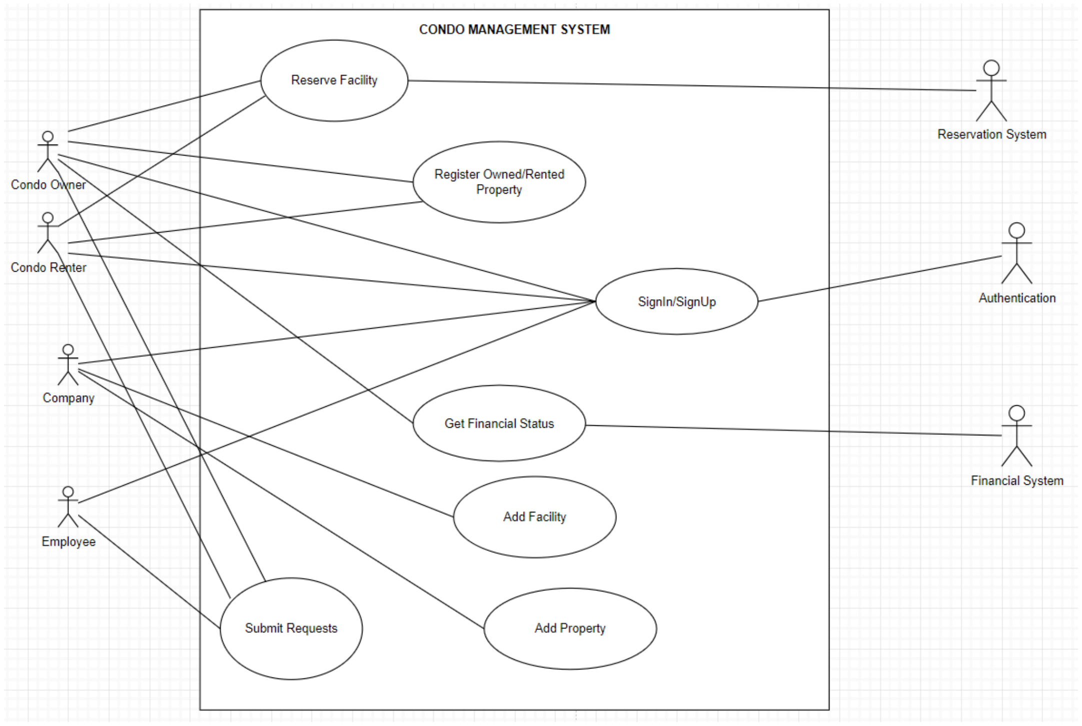


Figure 5: Use Case Diagram of Condo Management System

3.4.6. Activity Diagram

Activity Diagrams are a variation of the state diagram that show the workflow from a start point to the finish point detailing the many decision paths that exist in the progression of events contained in the activity. Below are some of the activity/function/use-case involved in the system.

3.4.6.1. AD1 - Sign In/Sign Up

This visualizes the activity of a user logging in or registering for the first time. The user starts at the login page. If the user is already registered, they enter their information to be validated and logged in. If they are a new user, they are taken to the registration page where they select their user category and enter the information to be registered.

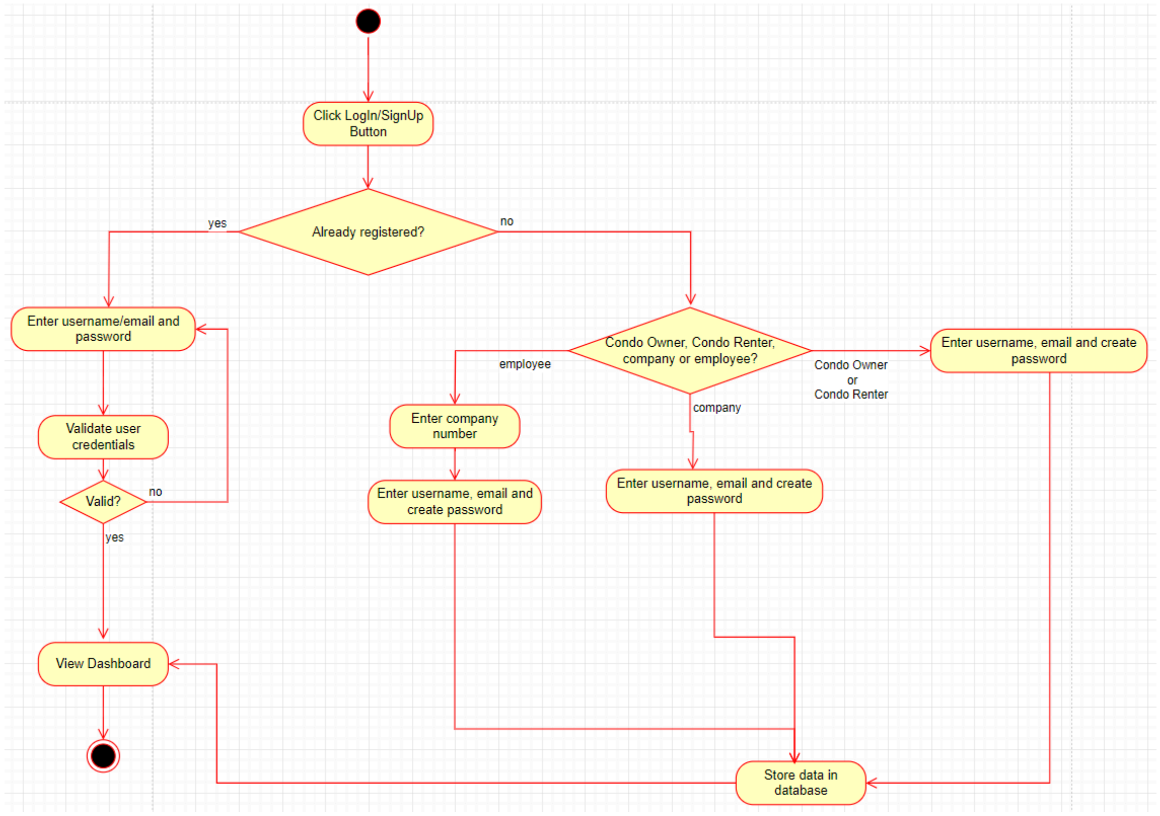
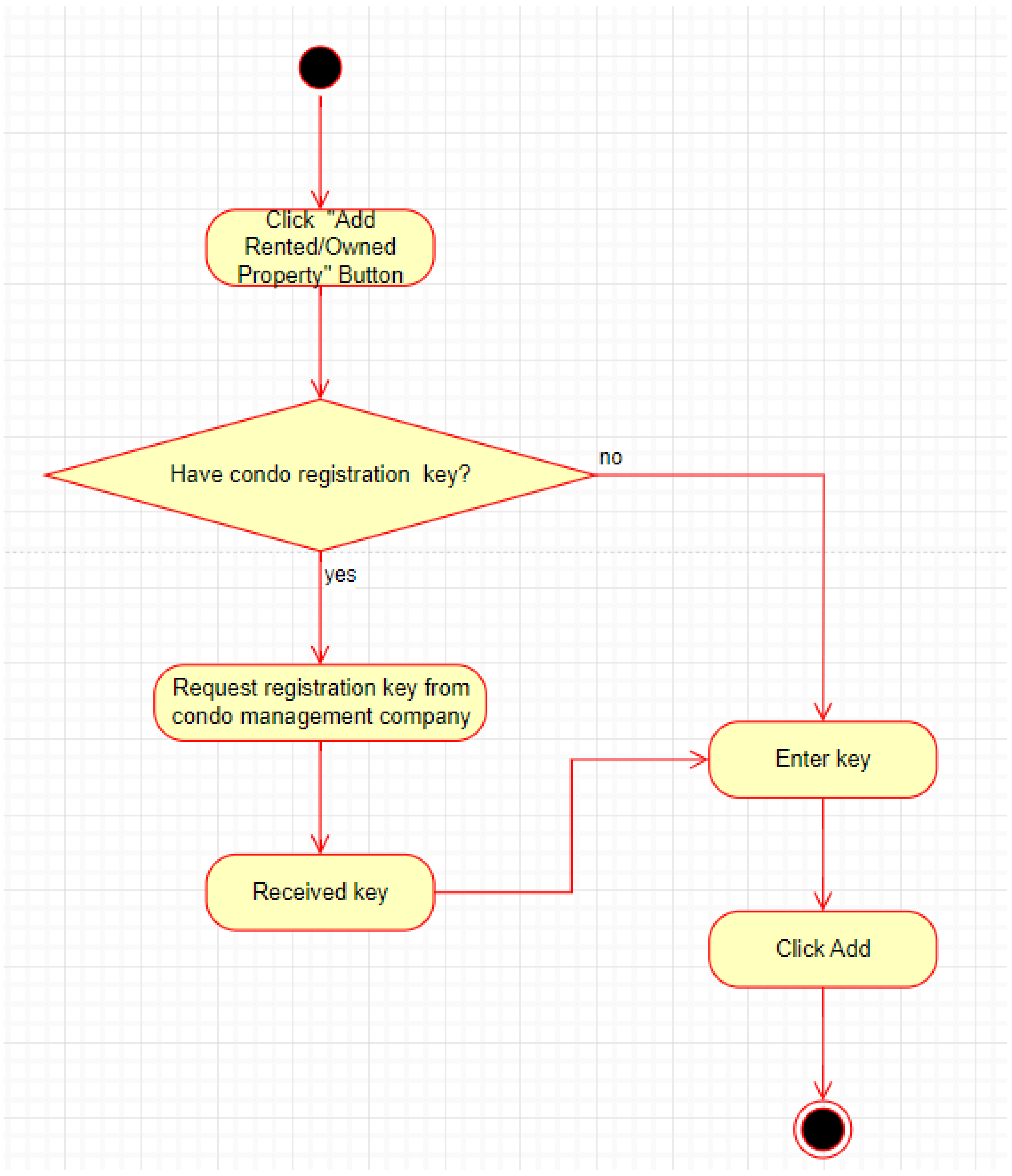

Click LogIn/SignUp Button
Already registered?
yes
no
Enter username/email and password
Validate user credentials
Valid?
no
yes
View Dashboard
Condo Owner, Condo Renter, company or employee?
employee
company
Condo Owner or Condo Renter
Enter company number
Enter username, email and create password
Enter username, email and create password
Enter username, email and create password
Store data in database

Figure 6: Activity Diagram for Sign In/Sign up

3.4.6.2. AD2 - Registering as Owner/Renter

This visualizes the activity of a user registering a property that they own or rent. From the profile dashboard the user clicks on the "Register Owner/Renter" button. On clicking, the user is displayed a field where they can enter the registration key they received from the condo management company. If they do not have a key, the user can request one from the company.



Click "Add Rented/Owned Property" Button
Have condo registration key?
no
yes
Request registration key from condo management company
Enter key
Received key
Click Add

Figure 7: Activity Diagram for Registering as Owner/Renter

3.4.6.3. AD3 - Reserving a Facility

This diagram visualizes the activity of reserving a facility in the system. From the dashboard the user clicks on the "Reserve Facility" button. The user is taken to the reserve facility page where the user is asked to select the facility type and select from and to date from the calendar to reserve and then click the submit button. If the facility is available it is reserved, else an error is shown that the facility is reserved and the user is asked to repeat the process.

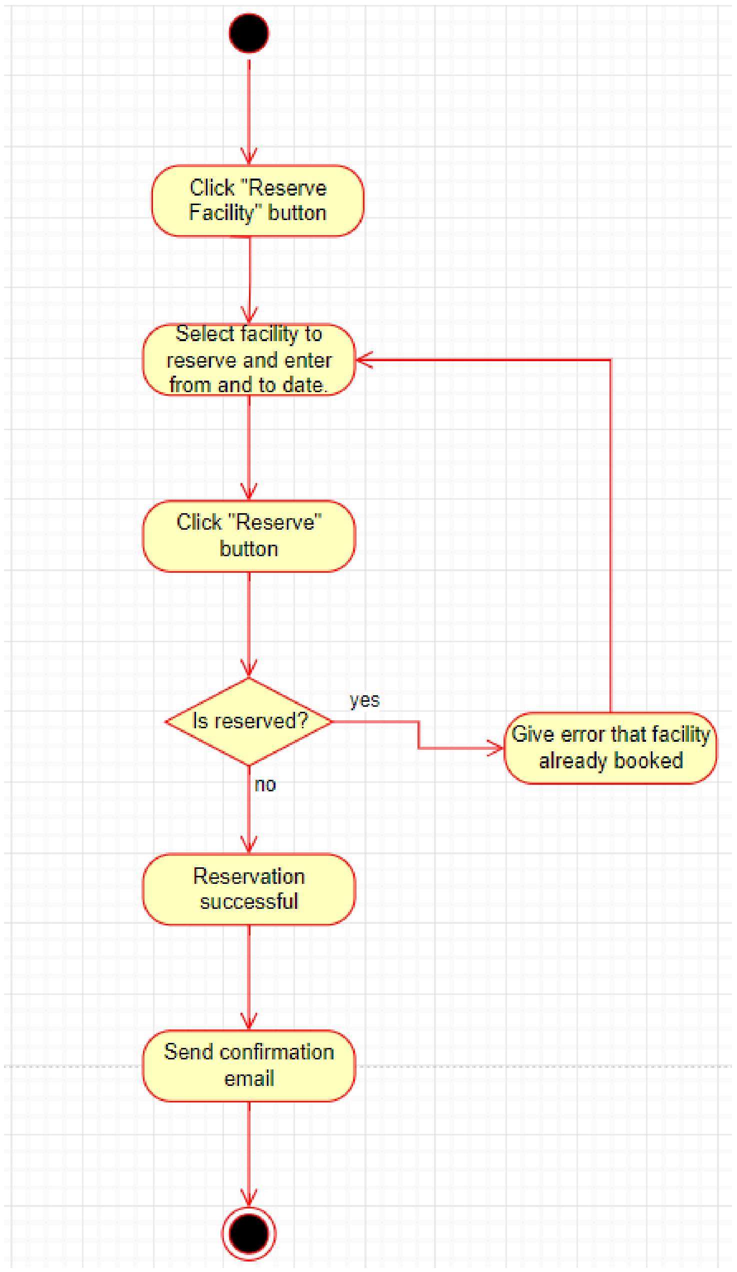

Click "Reserve Facility" button
Select facility to reserve and enter from and to date.
Click "Reserve" button
Is reserved?
yes
no
Give error that facility already booked
Reservation successful
Send confirmation email

Figure 8: Activity Diagram for Reserving a facility

3.4.6.4. AD4 - Submitting a Request

This diagram visualizes the activity to submit a request in the system. User clicks on submit a request button and is taken to the request page. Here the user is asked to select the type of request from the dropdown and then enter his contact information. After entering the details, the user clicks on the submit button to submit the request.

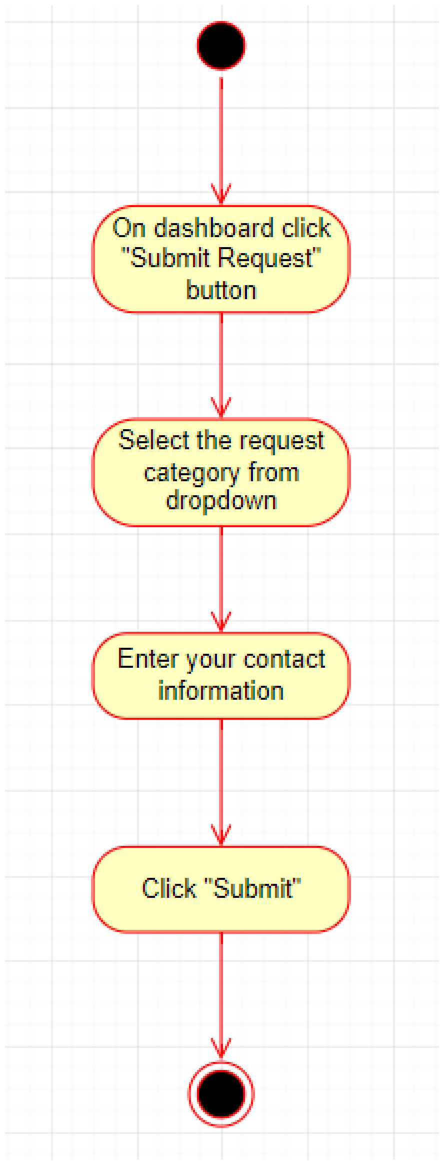

On dashboard click "Submit Request" button
Select the request category from dropdown
Enter your contact information
Click "Submit"

Figure 9: Activity Diagram for Submitting a request

3.4.6.5. AD5 - Access Financial Report

This diagram visualizes the activity of how a user gets a financial report of his total payments and payments made for a specific year or month. From the dashboard, the user clicks the “Financial Reports” button. The user then selects the year or month for which he wants to see the statistical analysis of. After selecting the user clicks “Get Report” and the information is displayed on the screen.

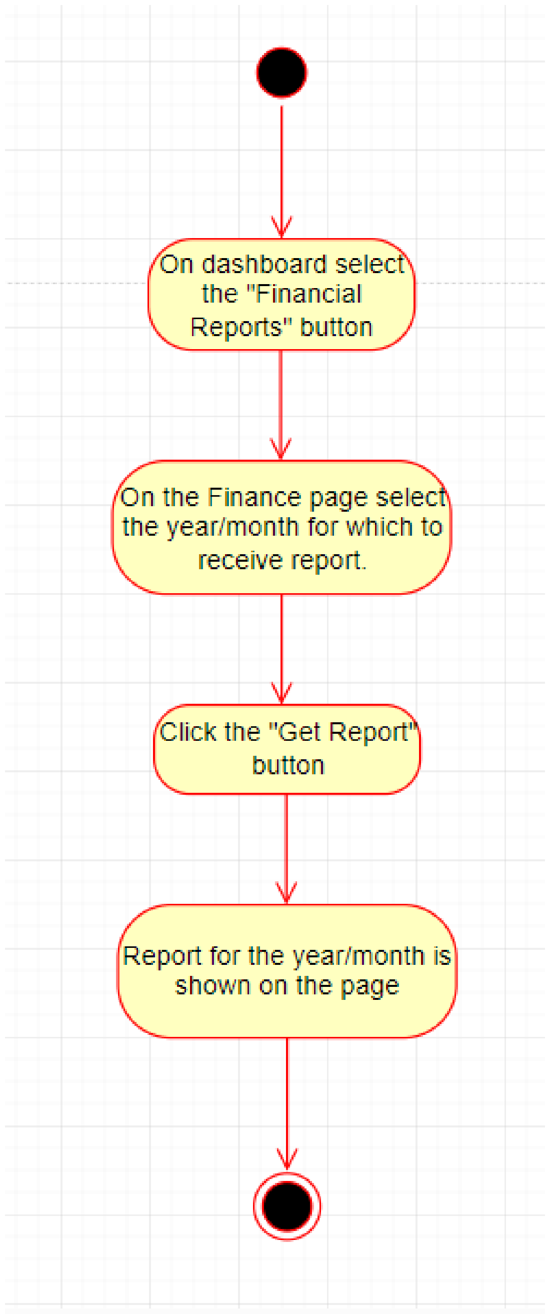

On dashboard select the "Financial Reports" button
On the Finance page select the year/month for which to receive report.
Click the "Get Report" button
Report for the year/month is shown on the page

Figure 10: Activity Diagram for Accessing Financial Reports

4. Risk Assessment and Risk Management Plan

The objective of the Risk Assessment and Management Plan is to identify the potential risks and outline the mitigation strategies for the identified risks that could affect the timely deployment of the Condo Management System. This plan provides a high-level view to manage the associated risks.

Risk Identification:

- The team conducted iterative brainstorming sessions post each sprint and utilized historical data to identify the potential risks.
- Using the expertise of each teammate the risks from different domains were included to have a well-rounded approach.
- The risks have been classified into the following categories: technical, management, external, documentation, design and budget-related.
- Using the risk management matrix, all the identified risks are categorized into different tiers based on their scores and hence prioritized accordingly.

Impact Probability	Low	Medium	High
Low	🟩	🟩	🟧
Medium	🟩	🟧	🟥
High	🟧	🟥	🟥

Figure 11 : Risk management guide

4.1. Risk management matrix

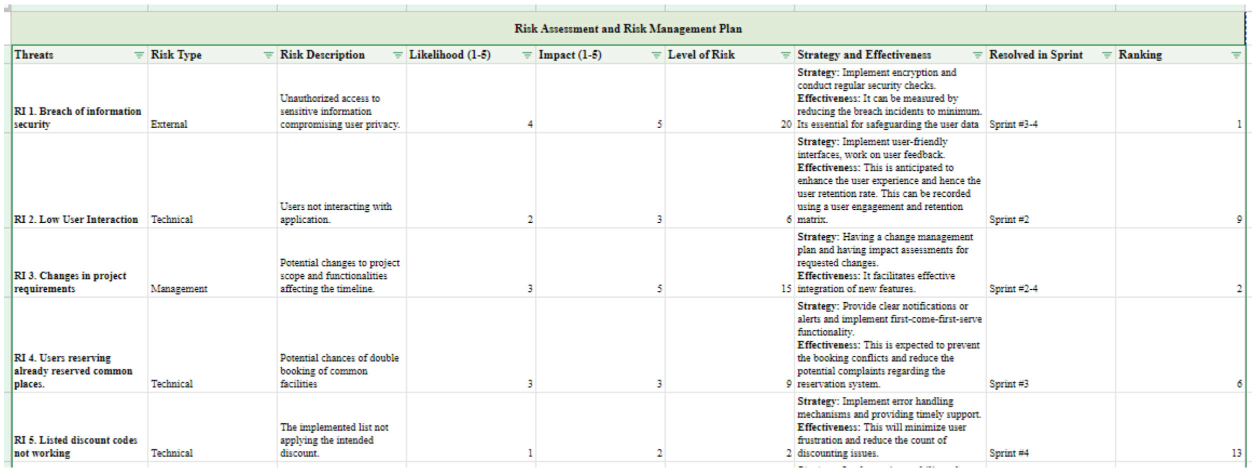
Risk ID	Risk Type and Description	Risk Score	Resolved in Sprint	Strategy and Effectiveness
RI 1. Breach of information security	- External - Unauthorized access to sensitive information compromising user privacy.	High with high impact	Sprint #3-4	**Strategy:** Implement encryption and conduct regular security checks. **Effectiveness:** It can be measured by reducing the breach incidents to minimum. It's essential for safeguarding the user data.
RI 2. Low User Interaction	- Technical - Users not interacting with applications.	Low with medium impact	Sprint #2	**Strategy:** Implement user-friendly interfaces, work on user feedback. **Effectiveness:** This is anticipated to enhance the user experience and hence the user retention rate. This can be recorded using a user engagement and retention matrix.
RI 3. Changes in project requirements	- Management - Potential changes to project scope and functionalities affecting the timeline.	High with high impact	Sprint #2-4	**Strategy:** Having a change management plan and having impact assessments for requested changes. **Effectiveness:** It facilitates effective integration of new features.
RI 4. Users reserving already reserved common places.	- Technical - Potential chances of double booking of common facilities	Medium with medium impact	Sprint #3	**Strategy:** Provide clear notifications or alerts and implement first-come-first-serve functionality. **Effectiveness:** This is expected to prevent the

				booking conflicts and reduce the potential complaints regarding the reservation system.
RI 5. Listed discount codes not working	- Technical - The implemented list does not apply the intended discount.	Low with low impact	Sprint #4	**Strategy:** Implement error handling mechanisms and provide timely support. **Effectiveness:** This will minimize user frustration and reduce the count of discounting issues.
RI 6. Language barrier	- External - Language differences hinder the user engagement with the system.	Low with low impact	Sprint #4	**Strategy:** Implementing multilingual support and testing with diverse user groups. **Effectiveness:** This helps in expanding the user base and increase inclusivity. It can be measured using an engagement matrix.
RI 7. Poor budget allocation	- Budget - Inadequate allocation of funds leading to financial constraints	Medium with high impact	Sprint #2-3	**Strategy:** Conduct a thorough budget analysis for the project and prioritizing budget based on a cost-benefit analysis. **Effectiveness:** Ensures that the financial resources are efficiently allocated and can be tracked by analyzing regularly updated budget reports.
RI 8. Poor communication with stakeholders	- Management - Lack of frequent and effective communication	Medium with high impact	Sprint #2	**Strategy:** Conduct regular and transparent meetings with the stakeholders. **Effectiveness:** This

				improves project transparency.
RI 9. Insufficient user training	- Management - Lack of proper training needed for the users	Medium with medium impact	Sprint #2	**Strategy:** Develop training modules and provide hands-on training if needed. Having a support team to assist the users with any issues. **Effectiveness:** This is the basis for user competency and helps to increase user performance and interaction.
RI 10. Data Corruption	- Technical - Caused due to software failures or possible human errors.	Medium with high impact	Sprint #2	**Strategy:** Regularly validate data integrity and recovery process. **Effectiveness:** It is critical for maintaining data integrity.
RI 11. Lack of Documentation	- Documentation - Incomplete documentation of software code, architecture and design.	Low with medium impact	Sprint #3	**Strategy:** Regularly maintain and update the Code Management Document. **Effectiveness:** This is essential for knowledge transfer.
RI 12. Scalability Constraints	- Technical - With scalability, the website will have higher user traffic which needs to be managed.	Low with medium impact	Sprint #4	**Strategy:** Having upgrade plans strategized and using proper cloud infrastructure that helps in scalability. **Effectiveness:** This ensures system longevity.
RI 13. Unexpected Bugs	- Technical - Unexpected software bugs can cause system	Medium with high impact	Sprint #3-4	**Strategy:** Implement thorough testing and constantly manage

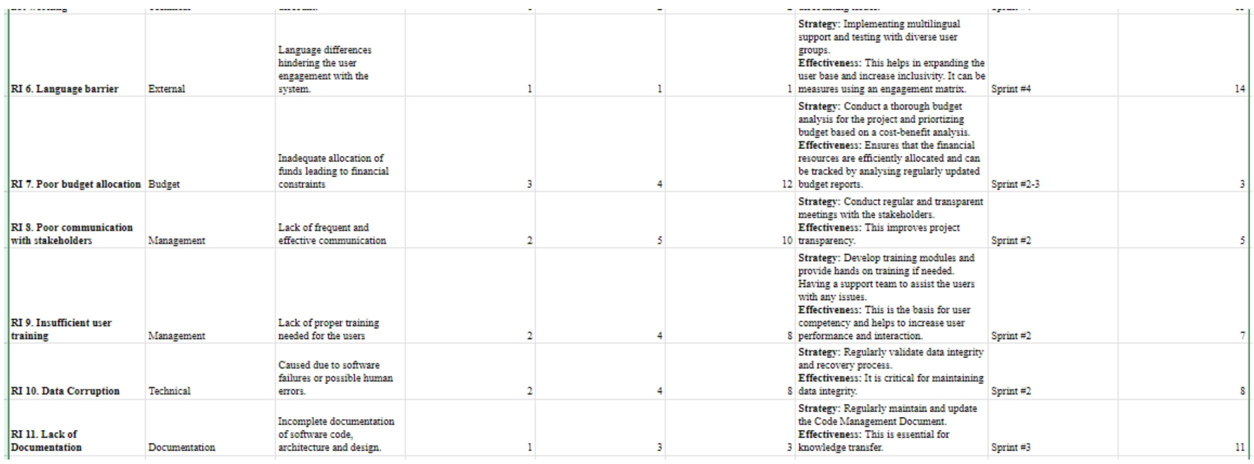
	malfunction and lead to system down time.			system bug testing reports. **Effectiveness:** This helps in system reliability.
RI 14. Regulatory Compliance	- Technical - System needs to comply with the industry regulations	Low with medium impact	Sprint #5	**Strategy:** Have a team managing and staying up to date with the latest industry regulations. **Effectiveness:** It prevents legal repercussions and reduces the compliance complaints to zero.

Table 2 : Risk management matrix

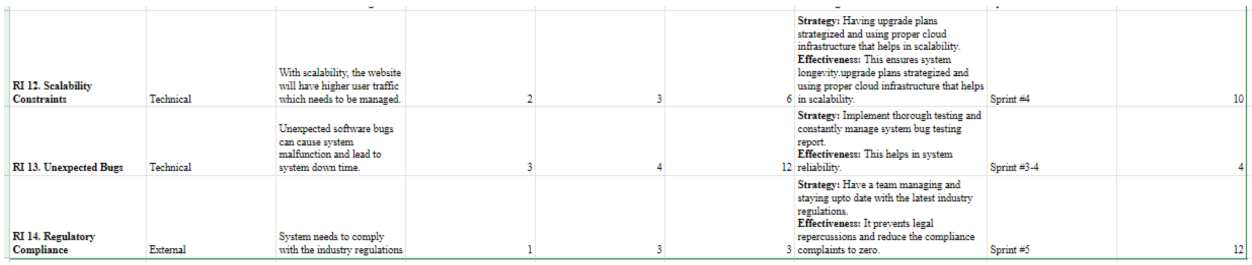


Risk Assessment and Risk Management Plan

Threats	Risk Type	Risk Description	Likelihood (1-5)	Impact (1-5)	Level of Risk	Strategy and Effectiveness	Resolved in Sprint	Ranking
RI 1. Breach of information security	External	Unauthorized access to sensitive information compromising user privacy.	4	5	20	**Strategy**: Implement encryption and conduct regular security checks. **Effectiveness**: It can be measured by reducing the breach incidents to minimum. Its essential for safeguarding the user data	Sprint #3-4	1
RI 2. Low User Interaction	Technical	Users not interacting with application.	2	3	6	**Strategy**: Implement user-friendly interfaces, work on user feedback. **Effectiveness**: This is anticipated to enhance the user experience and hence the user retention rate. This can be recorded using a user engagement and retention matrix.	Sprint #2	9
RI 3. Changes in project requirements	Management	Potential changes to project scope and functionalities affecting the timeline.	3	5	15	**Strategy**: Having a change management plan and having impact assessments for requested changes. **Effectiveness**: It facilitates effective integration of new features.	Sprint #2-4	2
RI 4. Users reserving already reserved common places.	Technical	Potential chances of double booking of common facilities	3	3	9	**Strategy**: Provide clear notifications or alerts and implement first-come-first-serve functionality. **Effectiveness**: This is expected to prevent the booking conflicts and reduce the potential complaints regarding the reservation system.	Sprint #3	6
RI 5. Listed discount codes not working	Technical	The implemented list not applying the intended discount.	1	2	2	**Strategy**: Implement error handling mechanisms and providing timely support. **Effectiveness**: This will minimize user frustration and reduce the count of discounting issues.	Sprint #4	13



RI 6. Language barrier	External	Language differences hindering the user engagement with the system.	1	1	1	**Strategy**: Implementing multilingual support and testing with diverse user groups. **Effectiveness**: This helps in expanding the user base and increase inclusivity. It can be measures using an engagement matrix.	Sprint #4	14
RI 7. Poor budget allocation	Budget	Inadequate allocation of funds leading to financial constraints	3	4	12	**Strategy**: Conduct a thorough budget analysis for the project and priortizing budget based on a cost-benefit analysis. **Effectiveness**: Ensures that the financial resources are efficiently allocated and can be tracked by analysing regularly updated budget reports.	Sprint #2-3	3
RI 8. Poor communication with stakeholders	Management	Lack of frequent and effective communication	2	5	10	**Strategy**: Conduct regular and transparent meetings with the stakeholders. **Effectiveness**: This improves project transparency.	Sprint #2	5
RI 9. Insufficient user training	Management	Lack of proper training needed for the users	2	4	8	**Strategy**: Develop training modules and provide hands on training if needed. Having a support team to assist the users with any issues. **Effectiveness**: This is the basis for user competency and helps to increase user performance and interaction.	Sprint #2	7
RI 10. Data Corruption	Technical	Caused due to software failures or possible human errors.	2	4	8	**Strategy**: Regularly validate data integrity and recovery process. **Effectiveness**: It is critical for maintaining data integrity.	Sprint #2	8
RI 11. Lack of Documentation	Documentation	Incomplete documentation of software code, architecture and design.	1	3	3	**Strategy**: Regularly maintain and update the Code Management Document. **Effectiveness**: This is essential for knowledge transfer.	Sprint #3	11



RI 12. Scalability Constraints	Technical	With scalability, the website will have higher user traffic which needs to be managed.	2	3	6	**Strategy:** Having upgrade plans strategized and using proper cloud infrastructure that helps in scalability. **Effectiveness:** This ensures system longevity.upgrade plans strategized and using proper cloud infrastructure that helps in scalability.	Sprint #4	10
RI 13. Unexpected Bugs	Technical	Unexpected software bugs can cause system malfunction and lead to system down time.	3	4	12	**Strategy:** Implement thorough testing and constantly manage system bug testing report. **Effectiveness:** This helps in system reliability.	Sprint #3-4	4
RI 14. Regulatory Compliance	External	System needs to comply with the industry regulations	1	3	3	**Strategy:** Have a team managing and staying upto date with the latest industry regulations. **Effectiveness:** It prevents legal repercussions and reduce the compliance complaints to zero.	Sprint #5	12

Figure 12: Risk Management Sheet

5. Release Plans

5.1. Release Plan Matrix

User Story ID	Sub-User Stories	User Story Points (USP)	Priority	Status	Expected Sprint
1. User Profile Authentication	1.1. Validate unique profile	2	High	Done	Sprint 1
2. Condo Owner's Dashboard	2.1. Display profile information	3	High	Done	Sprint 2
	2.2. Overview of owned properties				
	2.3. Financial status				
	2.4. Requests submitted				
3. Condo Management Company Profile	3.1. Property Profiles	3	High	Done	Sprint 2
	3.2. Uploading condo files				
	3.3. Include detailed information for condo units				
	3.4. Arrange exchange of registration keys				
4. Financial System	4.1. Setting condo fees	5	High	Done	Sprint 2

	4.2. Condo fee calculation				
	4.3. Record operational budget				
	4.4. Generate annual financial reports				
5. Reservation System	5.1. Setup common facilities	5	Medium	Todo	Sprint 3
	5.2. Interface for reservations				
	5.3. Availabilities for the listed facilities				
	5.4. Implement first-come-first-serve functionality				
6. Employee Role Management System	6.1. Define employee roles	1	Medium	Todo	Sprint 3
	6.2. Assigning roles to employees				
7. Request Management System	7.1. Implement different request types	2	Medium/Low	Todo	Sprint 3
	7.2. Assign requests to employees				
8. Notification Page	8.1. Create a notification page.	2	Medium	Todo	Sprint 4
	8.2. Display the latest activities and requests				
9. Additional Features	9.1. Develop user forum functionality	5	Low	Todo	Sprint 5
	9.2. Create an events page to organize events and invite users.				
	9.3. Develop a functional discount and offer list				
10. Bonus Features	10.1. Cross-platform accessibility	5	Low	Todo	Sprint 5
	10.2. Multilingual				

	10.3. User login using various accounts like Gmail, SSO etc.				
Total USP		33 points			

Table 3: Release Plan

5.2. User Story Points covered across each sprint:

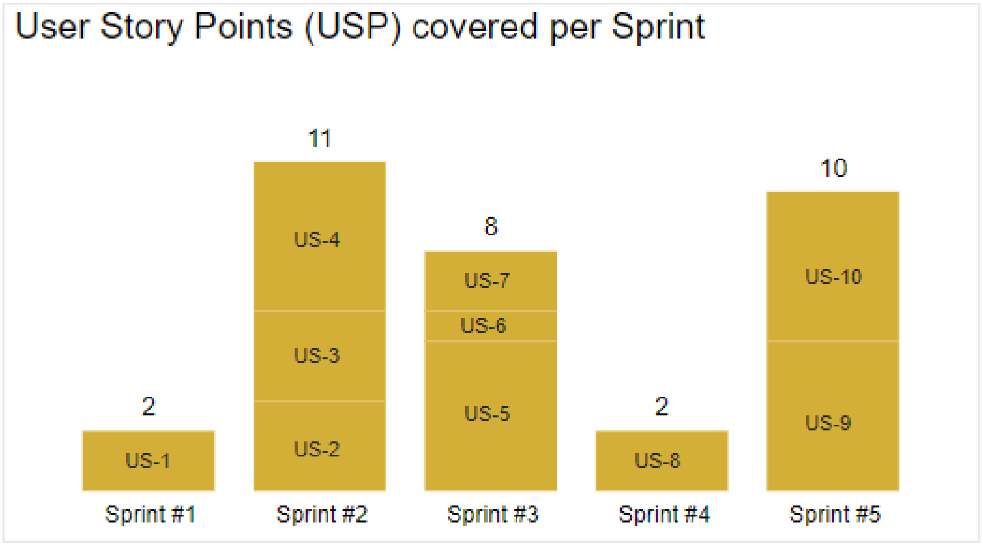
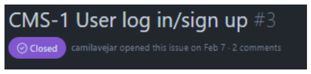
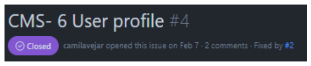
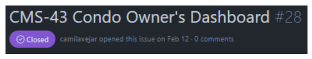
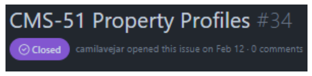
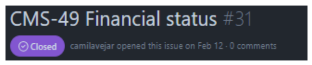
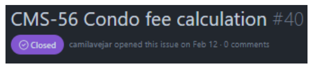
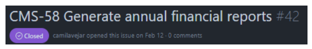
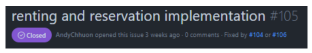


Figure 13: User Stories covered per Sprint

6. Traceability matrix with requirements

Req ID	Requirement	Description	Related Github Issue	Related Testing
1	User Registration	Users would like to sign in and create a profile with their picture and information.	Issue #3 and #4  CMS-1 User log in/sign up #3 Closed camilavejar opened this issue on Feb 7 · 2 comments &  CMS- 6 User profile #4 Closed camilavejar opened this issue on Feb 7 · 2 comments · Fixed by #2	In Sprint 2 Testing report, Test #1 and #2 show the successful testing of the sign in functionality.

2	Condo Owner Dashboard	As condo owner, access to dashboard with information about my properties, profile and financial status	Issue #28  CMS-43 Condo Owner's Dashboard #28 Closed camilavejar opened this issue on Feb 12 · 0 comments	In Sprint 3 Testing report, Test #3 to #5 show successful testing of this requirement
3	Property Profile Creation	As a condo management company I want to create a profile for all properties under my management and show their essential information	Issue # 34  CMS-51 Property Profiles #34 Closed camilavejar opened this issue on Feb 12 · 0 comments	In Sprint 3 Testing report, Test #11 verifies this requirement
4	Uploading Condo Files	As a condo management company, want to upload condo files for each property	Issue #35	In Sprint 3 Testing report, Test #11 verifies this requirement
5	Financial System	As a user, I would like to see financial status like due fees, expenses and financial reports.	Issue #31, #40 and #42  CMS-49 Financial status #31 Closed camilavejar opened this issue on Feb 12 · 0 comments  CMS-56 Condo fee calculation #40 Closed camilavejar opened this issue on Feb 12 · 0 comments  CMS-58 Generate annual financial reports #42 Closed camilavejar opened this issue on Feb 12 · 0 comments	In Sprint 3 Testing report, Test #12 to #15 verify successful working of this requirement
6	Reservation System	As a condo owner/renter, want to be able to reserve a facility available through a calendar-like interface	Issue #105  renting and reservation implementation #105 Closed AndyChhuon opened this issue 3 weeks ago · 0 comments · Fixed by #104 or #106	In Sprint 4 Testing report, Test #6 to #8 are used to test this requirement. The test passed resulting in successful execution
7	Request/ Messaging System	As condo owner/renter, want to be able to make requests related to condo	Issue #80 and #81	In Sprint 4 Testing report, Test #4 and #5, successfully test this

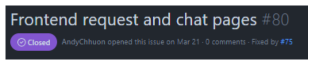
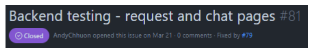
		matters and communicate with the relevant person	 Frontend request and chat pages #80 Closed AndyChhuon opened this issue on Mar 21 · 0 comments · Fixed by #75  Backend testing - request and chat pages #81 Closed AndyChhuon opened this issue on Mar 21 · 0 comments · Fixed by #79	requirement

Table 4: Traceability Matrix

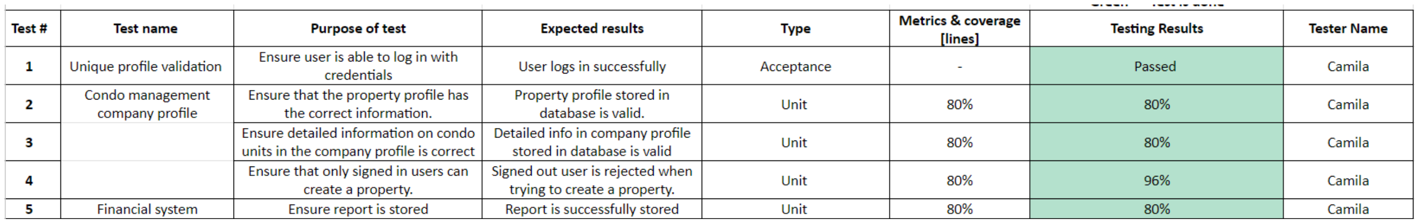
7. Complete Testing Report

The tools used for testing were Jest for unit testing and Cypress for integration testing. We aimed for 80% coverage throughout the sprints.

7.1. Sprint 2

7.1.1. Overview

5 tests were completed this sprint.

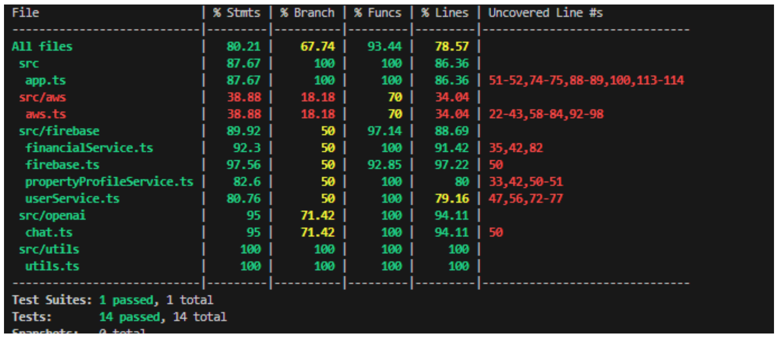


Test #	Test name	Purpose of test	Expected results	Type	Metrics & coverage [lines]	Testing Results	Tester Name
1	Unique profile validation	Ensure user is able to log in with credentials	User logs in successfully	Acceptance	-	Passed	Camila
2	Condo management company profile	Ensure that the property profile has the correct information.	Property profile stored in database is valid.	Unit	80%	80%	Camila
3		Ensure detailed information on condo units in the company profile is correct	Detailed info in company profile stored in database is valid	Unit	80%	80%	Camila
4		Ensure that only signed in users can create a property.	Signed out user is rejected when trying to create a property.	Unit	80%	96%	Camila
5	Financial system	Ensure report is stored	Report is successfully stored	Unit	80%	80%	Camila

Figure 14

7.1.2. Unit testing

Unit testing achieved an overall statement coverage of 83% with all of our tests passing. We have two items that do not pass the tests but we are not taking them into account. We are working with AWS and that element does not work well with Jest. It does not represent the testing and so we are not taking it into account in our coverage.



```
File                        | % Stmts | % Branch | % Funcs | % Lines | Uncovered Line #s
----------------------------|---------|----------|---------|---------|------------------------------
All files                   |   80.21 |    67.74 |   93.44 |   78.57 |
 src                        |   87.67 |      100 |     100 |   86.36 |
  app.ts                    |   87.67 |      100 |     100 |   86.36 | 51-52,74-75,88-89,100,113-114
 src/aws                    |   38.88 |    18.18 |      70 |   34.04 |
  aws.ts                    |   38.88 |    18.18 |      70 |   34.04 | 22-43,58-84,92-98
 src/firebase               |   89.92 |       50 |   97.14 |   88.69 |
  financialService.ts       |    92.3 |       50 |     100 |   91.42 | 35,42,82
  firebase.ts               |   97.56 |       50 |   92.85 |   97.22 | 50
  propertyProfileService.ts |    82.6 |       50 |     100 |      80 | 33,42,50-51
  userService.ts            |   80.76 |       50 |     100 |   79.16 | 47,56,72-77
 src/openai                 |      95 |    71.42 |     100 |   94.11 |
  chat.ts                   |      95 |    71.42 |     100 |   94.11 | 50
 src/utils                  |     100 |      100 |     100 |     100 |
  utils.ts                  |     100 |      100 |     100 |     100 |
----------------------------|---------|----------|---------|---------|------------------------------
Test Suites: 1 passed, 1 total
Tests:       14 passed, 14 total
```

Figure 15

7.1.3. Acceptance test

UAT1- User Login with credentials

Description: The user has an account and can log in into the system.
Test Steps:
1- User clicks on “Already have an account” hyperlink
2- User enters email address and password into text fields.
3- User clicks on login button

Expected result:
User is logged in successfully and has now access to the site.

Actual result:
Same as expected

Status:
Pass [X]
Fail [ ]

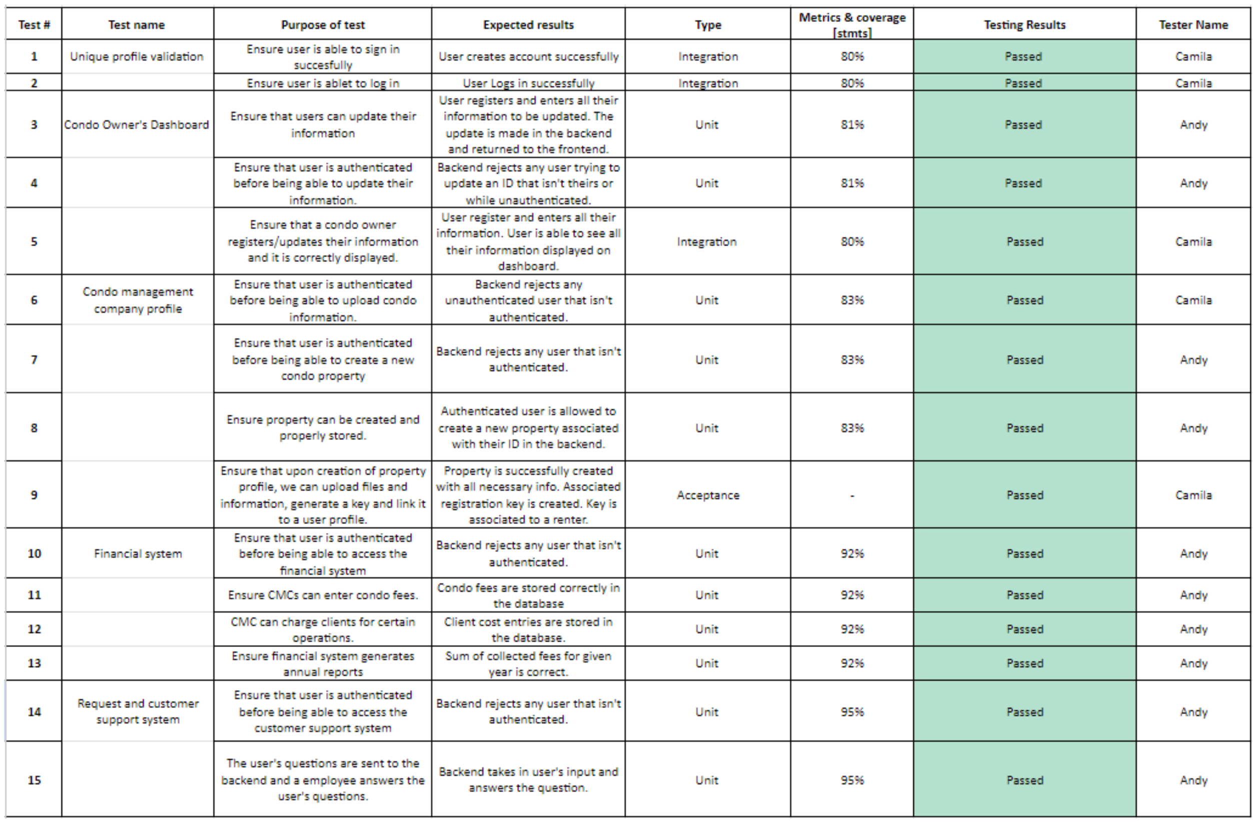
7.1.3. Improvements For The Next Sprint

In this first sprint of testing, we underestimated the amount of people needed for testing. We plan to add more testers and get team members more familiar with Jest. We also plan to try to finish implementation earlier to test more.

7.2. Sprint 3

7.2.1. Overview

15 tests were done this sprint. Cypress was introduced to use for integration testing.

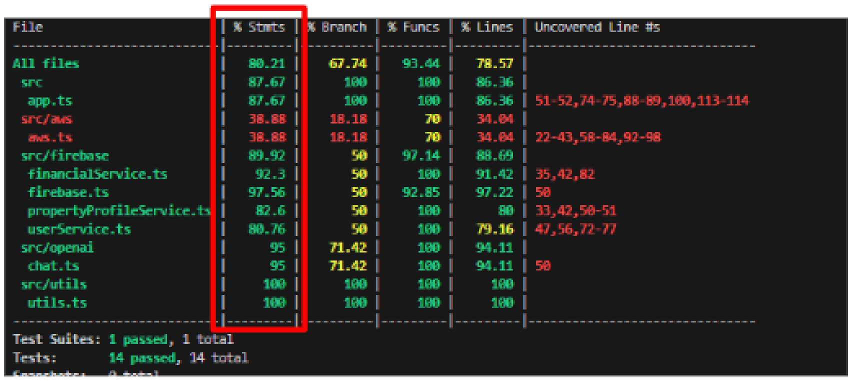


Test #	Test name	Purpose of test	Expected results	Type	Metrics & coverage [stmts]	Testing Results	Tester Name
1	Unique profile validation	Ensure user is able to sign in succesfully	User creates account successfully	Integration	80%	Passed	Camila
2		Ensure user is ablet to log in	User Logs in successfully	Integration	80%	Passed	Camila
3	Condo Owner's Dashboard	Ensure that users can update their information	User registers and enters all their information to be updated. The update is made in the backend and returned to the frontend.	Unit	81%	Passed	Andy
4		Ensure that user is authenticated before being able to update their information.	Backend rejects any user trying to update an ID that isn't theirs or while unauthenticated.	Unit	81%	Passed	Andy
5		Ensure that a condo owner registers/updates their information and it is correctly displayed.	User register and enters all their information. User is able to see all their information displayed on dashboard.	Integration	80%	Passed	Camila
6	Condo management company profile	Ensure that user is authenticated before being able to upload condo information.	Backend rejects any unauthenticated user that isn't authenticated.	Unit	83%	Passed	Camila
7		Ensure that user is authenticated before being able to create a new condo property	Backend rejects any user that isn't authenticated.	Unit	83%	Passed	Andy
8		Ensure property can be created and properly stored.	Authenticated user is allowed to create a new property associated with their ID in the backend.	Unit	83%	Passed	Andy
9		Ensure that upon creation of property profile, we can upload files and information, generate a key and link it to a user profile.	Property is successfully created with all necessary info. Associated registration key is created. Key is associated to a renter.	Acceptance	-	Passed	Camila
10	Financial system	Ensure that user is authenticated before being able to access the financial system	Backend rejects any user that isn't authenticated.	Unit	92%	Passed	Andy
11		Ensure CMCs can enter condo fees.	Condo fees are stored correctly in the database	Unit	92%	Passed	Andy
12		CMC can charge clients for certain operations.	Client cost entries are stored in the database.	Unit	92%	Passed	Andy
13		Ensure financial system generates annual reports	Sum of collected fees for given year is correct.	Unit	92%	Passed	Andy
14	Request and customer support system	Ensure that user is authenticated before being able to access the customer support system	Backend rejects any user that isn't authenticated.	Unit	95%	Passed	Andy
15		The user's questions are sent to the backend and a employee answers the user's questions.	Backend takes in user's input and answers the question.	Unit	95%	Passed	Andy

Figure 16

7.2.2. Unit Testing

Unit testing achieved 80-92% statement coverage. All tests passed.



```
File                         | % Stmts | % Branch | % Funcs | % Lines | Uncovered Line #s
-----------------------------|---------|----------|---------|---------|-----------------------------------
All files                    |   80.21 |    67.74 |   93.44 |   78.57 |
 src                         |   87.67 |      100 |     100 |   86.36 |
  app.ts                     |   87.67 |      100 |     100 |   86.36 | 51-52,74-75,88-89,100,113-114
 src/aws                     |   38.88 |    18.18 |      70 |   34.04 |
  aws.ts                     |   38.88 |    18.18 |      70 |   34.04 | 22-43,58-84,92-98
 src/firebase                |   89.92 |       50 |   97.14 |   88.69 |
  financialService.ts        |    92.3 |       50 |     100 |   91.42 | 35,42,82
  firebase.ts                |   97.56 |       50 |   92.85 |   97.22 | 50
  propertyProfileService.ts  |    82.6 |       50 |     100 |      80 | 33,42,50-51
  userService.ts             |   80.76 |       50 |     100 |   79.16 | 47,56,72-77
 src/openai                  |      95 |    71.42 |     100 |   94.11 |
  chat.ts                    |      95 |    71.42 |     100 |   94.11 | 50
 src/utils                   |     100 |      100 |     100 |     100 |
  utils.ts                   |     100 |      100 |     100 |     100 |
-----------------------------|---------|----------|---------|---------|-----------------------------------
Test Suites: 1 passed, 1 total
Tests:       14 passed, 14 total
Snapshots:   0 total
```

Figure 17

7.2.3. Integration Testing

Included here are the links to the video demos of these tests.

1- User Sign up (https://youtu.be/VWmMSfB_YSI)

2- User login (https://youtu.be/HEAGNi6yUpE)

3- User profile update (https://www.youtube.com/watch?v=CiJrLS-7Tj8)

Coverage is not included because there are issues in enabling it. We plan to fix it for the next sprint.

7.2.4. Acceptance Testing

UAT1- Create Property

Description: User is logged in and can create a property under their name successfully.

Test Steps:

1- User clicks on "Property Profiles" tab button.

2- User clicks on "add property" icon

3- User enters all necessary property information in the text boxes.

4- User clicks on "add property" button

Expected result:

Property is created and shown to the user.

Actual result:

Same as expected

Status:

Pass [X]

Fail [ ]

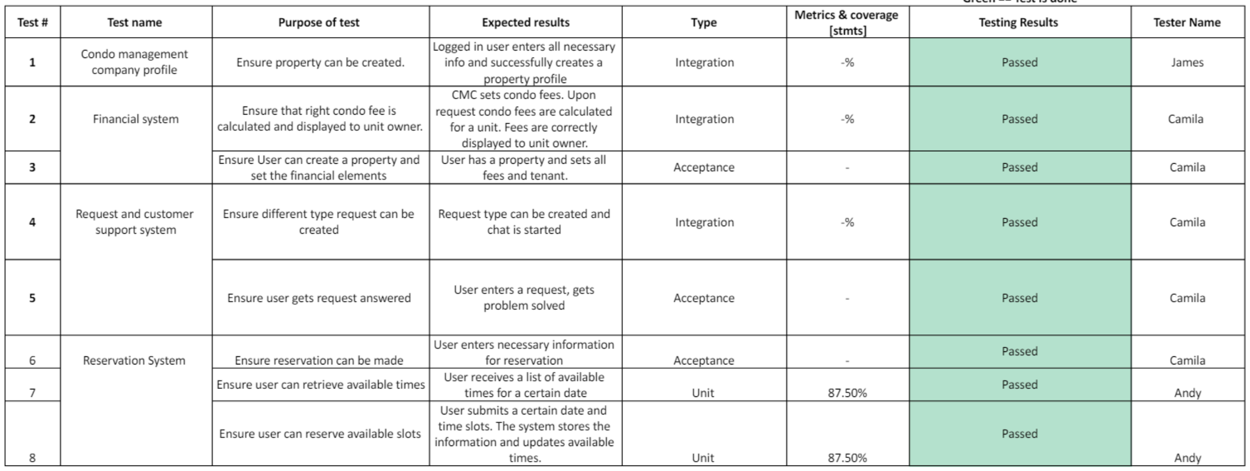
7.2.5. Improvements For The Next Sprint

As mentioned above, we will be fixing code coverage for Cypress. We will also work to simplify tests to make them more doable for more team members. We are still looking to add more members to the testing aspect of the project.

7.3. Sprint 4

7.3.1. Overview

In this sprint, we wrote 8 tests. We did unit testing and continued with end-to-end testing. We were able to enable code coverage with Cypress by instrumenting the code but there are still fixes to make. The coverage does not currently cover the right files but we will still include a sample of what it looks like. We will improve this code coverage feature for the next sprint.



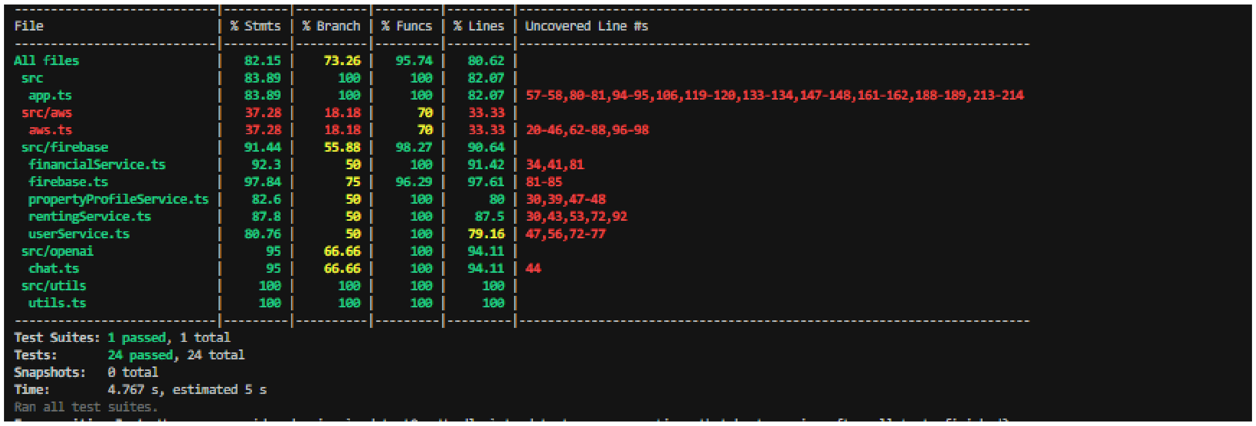
Test #	Test name	Purpose of test	Expected results	Type	Metrics & coverage [stmts]	Testing Results	Tester Name
1	Condo management company profile	Ensure property can be created.	Logged in user enters all necessary info and successfully creates a property profile	Integration	-%	Passed	James
2	Financial system	Ensure that right condo fee is calculated and displayed to unit owner.	CMC sets condo fees. Upon request condo fees are calculated for a unit. Fees are correctly displayed to unit owner.	Integration	-%	Passed	Camila
3		Ensure User can create a property and set the financial elements	User has a property and sets all fees and tenant.	Acceptance	-	Passed	Camila
4	Request and customer support system	Ensure different type request can be created	Request type can be created and chat is started	Integration	-%	Passed	Camila
5		Ensure user gets request answered	User enters a request, gets problem solved	Acceptance	-	Passed	Camila
6	Reservation System	Ensure reservation can be made	User enters necessary information for reservation	Acceptance	-	Passed	Camila
7		Ensure user can retrieve available times	User receives a list of available times for a certain date	Unit	87.50%	Passed	Andy
8		Ensure user can reserve available slots	User submits a certain date and time slots. The system stores the information and updates available times.	Unit	87.50%	Passed	Andy

Figure 18

7.3.2. Unit Testing

As shown above, the unit testing was focused on the reservation system story.

From the snapshot included below, we can see that the tests achieved 80%+ coverage.



```
----------------------------|---------|----------|---------|---------|-------------------------------------------------------------------
File                        | % Stmts | % Branch | % Funcs | % Lines | Uncovered Line #s
----------------------------|---------|----------|---------|---------|-------------------------------------------------------------------
All files                   |   82.15 |    73.26 |   95.74 |   80.62 |
 src                        |   83.89 |      100 |     100 |   82.07 |
  app.ts                    |   83.89 |      100 |     100 |   82.07 | 57-58,80-81,94-95,106,119-120,133-134,147-148,161-162,188-189,213-214
 src/aws                    |   37.28 |    18.18 |      70 |   33.33 |
  aws.ts                    |   37.28 |    18.18 |      70 |   33.33 | 20-46,62-88,96-98
 src/firebase               |   91.44 |    55.88 |   98.27 |   90.64 |
  financialService.ts       |    92.3 |       50 |     100 |   91.42 | 34,41,81
  firebase.ts               |   97.84 |       75 |   96.29 |   97.61 | 81-85
  propertyProfileService.ts |    82.6 |       50 |     100 |      80 | 30,39,47-48
  rentingService.ts         |    87.8 |       50 |     100 |    87.5 | 30,43,53,72,92
  userService.ts            |   80.76 |       50 |     100 |   79.16 | 47,56,72-77
 src/openai                 |      95 |    66.66 |     100 |   94.11 |
  chat.ts                   |      95 |    66.66 |     100 |   94.11 | 44
 src/utils                  |     100 |      100 |     100 |     100 |
  utils.ts                  |     100 |      100 |     100 |     100 |
----------------------------|---------|----------|---------|---------|-------------------------------------------------------------------
Test Suites: 1 passed, 1 total
Tests:       24 passed, 24 total
Snapshots:   0 total
Time:        4.767 s, estimated 5 s
Ran all test suites.
```

Figure 19

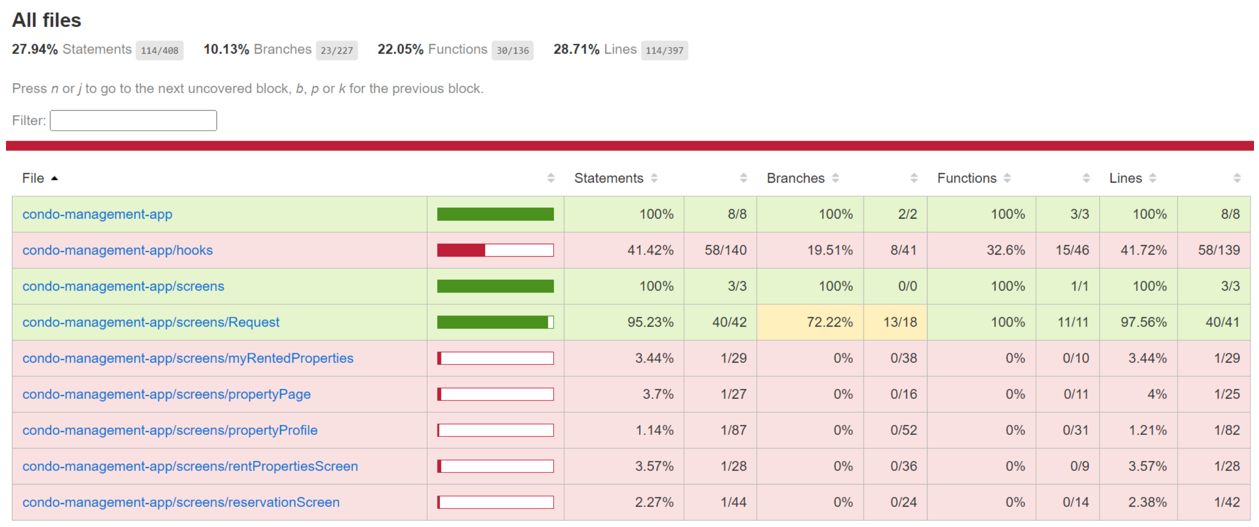
* The 37% coverage seen is from a database issue, it does not represent the tests and so we still consider all of our tests to have successfully passed.

7.3.3. Integration Testing

The following integration tests were completed:

- Adding a property (https://youtu.be/vKL7TgvR8Ww)
- Editing existing property (https://youtu.be/o_r3t_to1Zw)
- Creating a Request (https://youtu.be/ZoNRqN6HVEE)

Here we are including a sample of the coverage generated by Cypress. It still needs to be fixed, but it is at least working for this sprint.



All files

27.94% Statements 114/408 **10.13%** Branches 23/227 **22.05%** Functions 30/136 **28.71%** Lines 114/397

Press *n* or *j* to go to the next uncovered block, *b*, *p* or *k* for the previous block.

Filter:

File ▲		Statements		Branches		Functions		Lines	
condo-management-app	🟩	100%	8/8	100%	2/2	100%	3/3	100%	8/8
condo-management-app/hooks	🟥	41.42%	58/140	19.51%	8/41	32.6%	15/46	41.72%	58/139
condo-management-app/screens	🟩	100%	3/3	100%	0/0	100%	1/1	100%	3/3
condo-management-app/screens/Request	🟩	95.23%	40/42	72.22%	13/18	100%	11/11	97.56%	40/41
condo-management-app/screens/myRentedProperties	🟥	3.44%	1/29	0%	0/38	0%	0/10	3.44%	1/29
condo-management-app/screens/propertyPage	🟥	3.7%	1/27	0%	0/16	0%	0/11	4%	1/25
condo-management-app/screens/propertyProfile	🟥	1.14%	1/87	0%	0/52	0%	0/31	1.21%	1/82
condo-management-app/screens/rentPropertiesScreen	🟥	3.57%	1/28	0%	0/36	0%	0/9	3.57%	1/28
condo-management-app/screens/reservationScreen	🟥	2.27%	1/44	0%	0/24	0%	0/14	2.38%	1/42

Figure 20

As seen in the image above, the coverage numbers are very low, but with the testing done, this does not seem to be representative of what we have done. We plan to fix and improve this for the last sprint.

7.3.4. Acceptance Testing

UAT1- Creating Property and setting fees

Description: User can create a property profile and set the corresponding fees

Prerequisites: Users must have an account and be signed in.

Test Steps:

1- User clicks on “Property profiles” tab

2- User click on “+” to add property

3- User enters the property information on the text fields.

4- User clicks on “Add Property”

Expected result:

Property created is added at the bottom of the page.

Actual result:

Same as expected

Status:

Pass [X]

Fail [ ]

UAT2- Make a request
Description: User can make a request for maintenance or a complaint.
Prerequisites: Users must have an account and be signed in.
Test Steps:
1- User clicks on the “Make a Request” tab.
2- User selects the request type from the dropdown.
3- User enters the issue in the Message textbox
4- User clicks on “Send”
5- User provides more information to chat with employees.
6- User clicks on “End Chat” when the issue is resolved.

Expected result:
Chat is opened upon request creation and the issue is resolved by the employee.

Actual result:
Same as expected

Status:
Pass [X]
Fail [ ]

UAT3- Make a reservation
Description: User can make a reservation on a rented property
Prerequisites: Users must have an account and be signed in.
Test Steps:
1- User clicks on “Rent a Property” tab
2- User chooses property and clicks on “Start Renting” button
3- User clicks on “My Rented Properties” tab
4- User clicks on the “Make a Reservation” button for the desired service.
5- User selects date for reservation and clicks on “save” button
6- User selects time slot for reservation.
7- User clicks “Reserve” button

Expected result:
Reserved activity is displayed under “My displayed activities.

Actual result:
Same as Expected.

Status:
Pass [X]
Fail [ ]

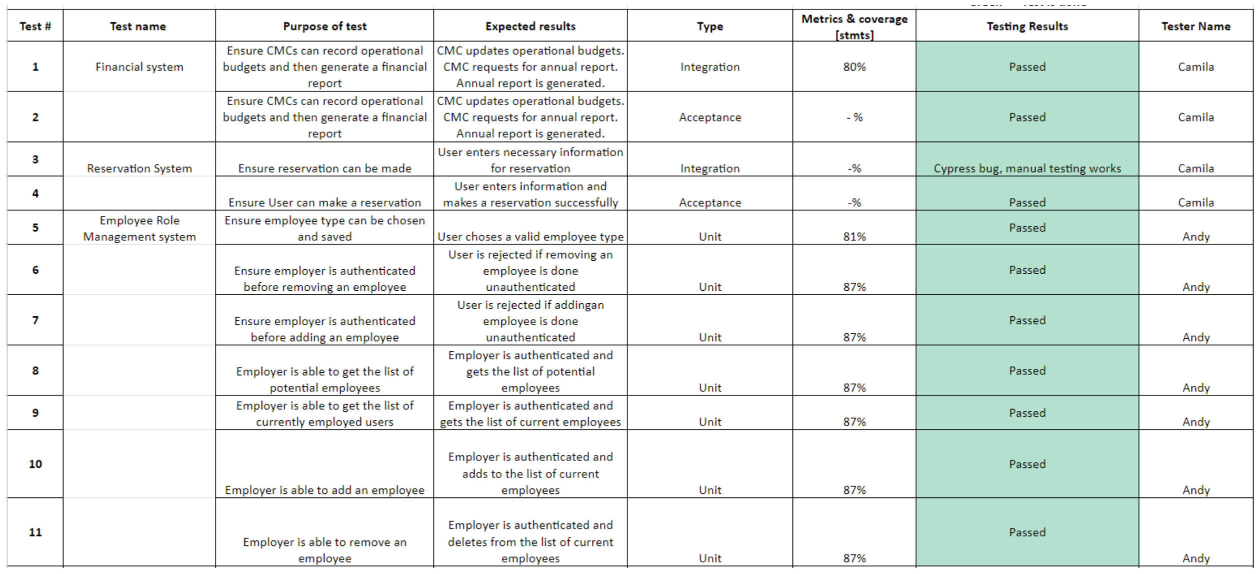
7.3.5. Improvements For The Next Sprint

For the next sprint, we plan to fix and improve the code coverage on Cypress, as well as finalize the testing for our main features. We were also planning to have an integration test done to show the reservation system, but we found that Cypress does not recognize the calendar widget we used. This made it so that we could not finish that test even though we knew it passed the acceptance test. Our plan for the next sprint is to fix this or change this widget to correctly test these features.

7.4. Sprint 5

7.4.1. Overview

11 tests were done this sprint. Focused on testing the main features we were working on last sprint.

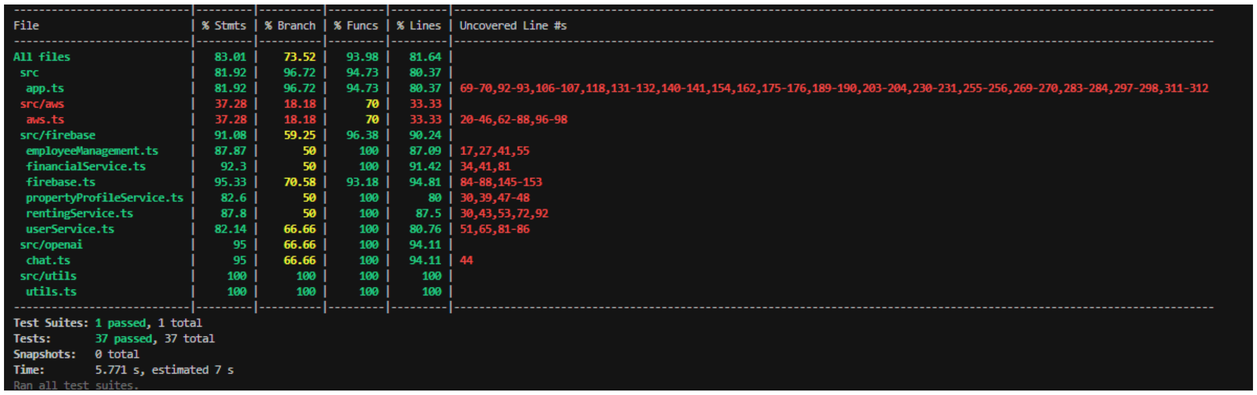


Test #	Test name	Purpose of test	Expected results	Type	Metrics & coverage [stmts]	Testing Results	Tester Name
1	Financial system	Ensure CMCs can record operational budgets and then generate a financial report	CMC updates operational budgets. CMC requests for annual report. Annual report is generated.	Integration	80%	Passed	Camila
2		Ensure CMCs can record operational budgets and then generate a financial report	CMC updates operational budgets. CMC requests for annual report. Annual report is generated.	Acceptance	- %	Passed	Camila
3	Reservation System	Ensure reservation can be made	User enters necessary information for reservation	Integration	-%	Cypress bug, manual testing works	Camila
4		Ensure User can make a reservation	User enters information and makes a reservation successfully	Acceptance	-%	Passed	Camila
5	Employee Role Management system	Ensure employee type can be chosen and saved	User choses a valid employee type	Unit	81%	Passed	Andy
6		Ensure employer is authenticated before removing an employee	User is rejected if removing an employee is done unauthenticated	Unit	87%	Passed	Andy
7		Ensure employer is authenticated before adding an employee	User is rejected if addingan employee is done unauthenticated	Unit	87%	Passed	Andy
8		Employer is able to get the list of potential employees	Employer is authenticated and gets the list of potential employees	Unit	87%	Passed	Andy
9		Employer is able to get the list of currently employed users	Employer is authenticated and gets the list of current employees	Unit	87%	Passed	Andy
10		Employer is able to add an employee	Employer is authenticated and adds to the list of current employees	Unit	87%	Passed	Andy
11		Employer is able to remove an employee	Employer is authenticated and deletes from the list of current employees	Unit	87%	Passed	Andy

Figure 21

7.4.2. Unit Testing

Code coverage was 81% for this sprint. Unit tests focused on the newly finished feature of employee roles.



```
----------------------------|---------|----------|---------|---------|-------------------------------------------------------------------------------------------------------------------
File                        | % Stmts | % Branch | % Funcs | % Lines | Uncovered Line #s
----------------------------|---------|----------|---------|---------|-------------------------------------------------------------------------------------------------------------------
All files                   |   83.01 |    73.52 |   93.98 |   81.64 |
 src                        |   81.92 |    96.72 |   94.73 |   80.37 |
  app.ts                    |   81.92 |    96.72 |   94.73 |   80.37 | 69-70,92-93,106-107,118,131-132,140-141,154,162,175-176,189-190,203-204,230-231,255-256,269-270,283-284,297-298,311-312
 src/aws                    |   37.28 |    18.18 |      70 |   33.33 |
  aws.ts                    |   37.28 |    18.18 |      70 |   33.33 | 20-46,62-88,96-98
 src/firebase               |   91.08 |    59.25 |   96.38 |   90.24 |
  employeeManagement.ts     |   87.87 |       50 |     100 |   87.09 | 17,27,41,55
  financialService.ts       |    92.3 |       50 |     100 |   91.42 | 34,41,81
  firebase.ts               |   95.33 |    70.58 |   93.18 |   94.81 | 84-88,145-153
  propertyProfileService.ts |    82.6 |       50 |     100 |      80 | 30,39,47-48
  rentingService.ts         |    87.8 |       50 |     100 |    87.5 | 30,43,53,72,92
  userService.ts            |   82.14 |    66.66 |     100 |   80.76 | 51,65,81-86
 src/openai                 |      95 |    66.66 |     100 |   94.11 |
  chat.ts                   |      95 |    66.66 |     100 |   94.11 | 44
 src/utils                  |     100 |      100 |     100 |     100 |
  utils.ts                  |     100 |      100 |     100 |     100 |
----------------------------|---------|----------|---------|---------|-------------------------------------------------------------------------------------------------------------------
Test Suites: 1 passed, 1 total
Tests:       37 passed, 37 total
Snapshots:   0 total
Time:        5.771 s, estimated 7 s
Ran all test suites.
```

Figure 22

7.4.3. Integration Testing

The following integration tests were completed:

- Add Cost Entry and Generate Report (https://youtu.be/OE76DAoZ1aU)

We also made an integration test to test the reservation process but we encountered a bug between Cypress and the calendar component we used to set the date (datepicker modal). It was tested manually and we are also including an acceptance test for it.

7.4.4. Acceptance Testing

UAT1- Add cost entry and generate report

Description: User is logged in and enters a cost for a specific tenant. Users can then generate a pdf report of these costs.

Test Steps:

1- User clicks on “Property Profiles” tab button

2- User clicks on the “Financials” button of a specific property

3- User enters the tenant ID, the cost description and cost into the text inputs.

4- User clicks on “Add Cost Entry” button

5- User clicks on “Generate report” button

Expected result:

Cost entry is added and pdf report is generated with the costs added.

Actual result:

Same as expected

Status:

Pass [X]

Fail [ ]

UAT2- Make a reservation
Description: User is logged in and can reserve an activity in a rented property.
Test Steps:
1- User clicks on “My Rented Properties” tab button
2- User clicks on the “Reserve” button for the activity wanted in the specific property.
3- User selects date in calendar and clicks on the “save” button
4- User selects the time period for the reservation
5- User clicks on “Reserve” button

Expected result:
Reservation info appears in “My reserved activities” below

Actual result:
Same as expected

Status:
Pass [X]
Fail [ ]

7.4.5. Improvements

For future sprints, we would work on the cypress coverage and maybe consider other tools. We would also consider doing system tests now that we have a functional system. There are still more tests to do with the new features, but that is for the future.

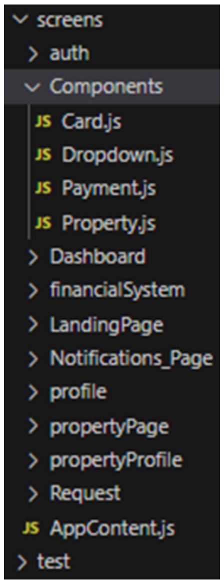
8. Code management

8.1. Repository structure

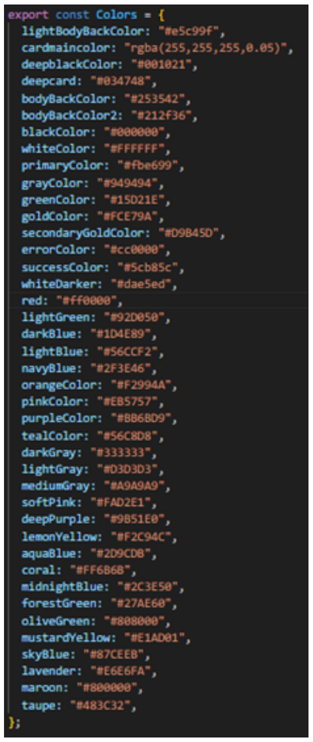
We are implementing a monorepo structure for this project. The backend is separated in its own folder (condo-backend), the frontend into another (condo-management-app), and the pipeline yml files into another (.github).

8.2. Frontend coding styles

Since we are coding the frontend in react, everything is divided into pages and components. Components are reusable pieces of code that are used all throughout the app in all pages. An example is a card component, which simply applies a specific style to the component's children.



All components associated with the same screen are grouped together, allowing for readability and maintainability of the code base. Each component has a specific purpose, allowing for separation of concerns.



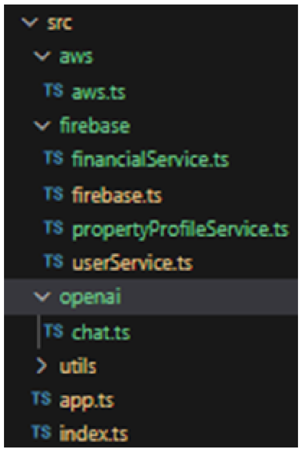
```
export const Colors = {
  lightBodyBackColor: "#e5c99f",
  cardmaincolor: "rgba(255,255,255,0.05)",
  deepblackColor: "#001021",
  deepcard: "#034748",
  bodyBackColor: "#253542",
  bodyBackColor2: "#212f36",
  blackColor: "#000000",
  whiteColor: "#FFFFFF",
  primaryColor: "#fbe699",
  grayColor: "#949494",
  greenColor: "#15D21E",
  goldColor: "#FCE79A",
  secondaryGoldColor: "#D9B45D",
  errorColor: "#cc0000",
  successColor: "#5cb85c",
  whiteDarker: "#dae5ed",
  red: "#ff0000",
  lightGreen: "#92D050",
  darkBlue: "#1D4E89",
  lightBlue: "#56CCF2",
  navyBlue: "#2F3E46",
  orangeColor: "#F2994A",
  pinkColor: "#EB5757",
  purpleColor: "#BB6BD9",
  tealColor: "#56C8D8",
  darkGray: "#333333",
  lightGray: "#D3D3D3",
  mediumGray: "#A9A9A9",
  softPink: "#FAD2E1",
  deepPurple: "#9B51E0",
  lemonYellow: "#F2C94C",
  aquaBlue: "#2D9CDB",
  coral: "#FF6B6B",
  midnightBlue: "#2C3E50",
  forestGreen: "#27AE60",
  oliveGreen: "#808000",
  mustardYellow: "#E1AD01",
  skyBlue: "#87CEEB",
  lavender: "#E6E6FA",
  maroon: "#800000",
  taupe: "#483C32",
};
```

In order to ensure consistency across all pages, we have a color palette and set of fonts used throughout the app, which can be found in the styles.js file.

Component, function and variable names follow CamelCase/camelCase styling to help with readability.

8.3. Backend coding styles

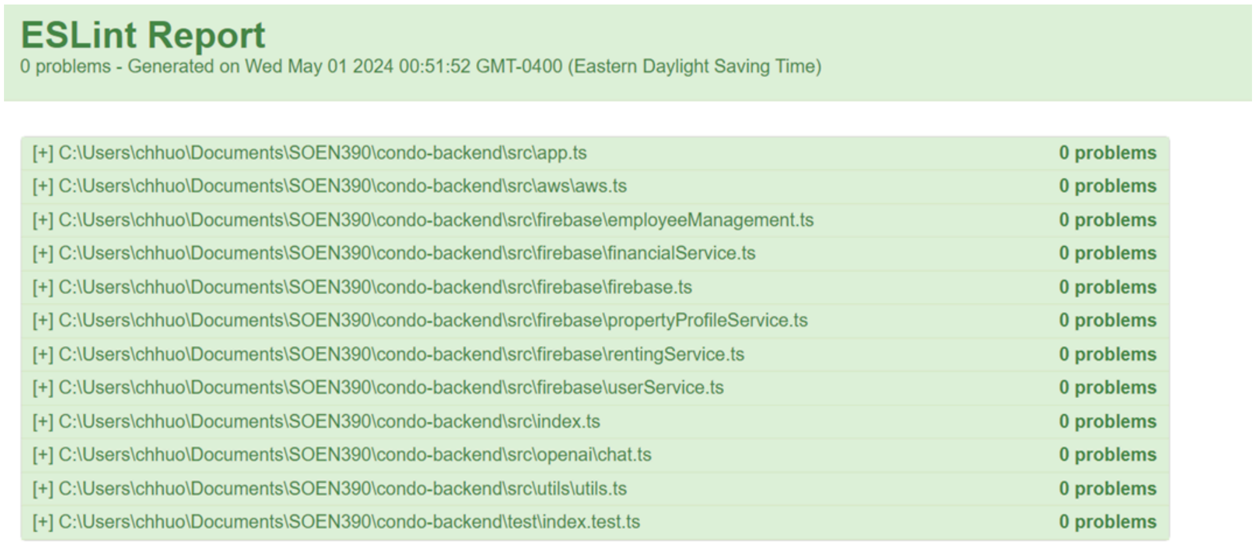
The backend follows a controller/service structure, where the service files are responsible for the bulk of the backend logic and the controllers (in app.ts) simply expose the functions as an endpoint.


```
src
  aws
    aws.ts
  firebase
    financialService.ts
    firebase.ts
    propertyProfileService.ts
    userService.ts
  openai
    chat.ts
  utils
  app.ts
  index.ts
```

8.4. Prettier and ESLint

We use prettier and eslint during the development to maintain a certain code style and for static analysis. We use the *standard-with-typescript* configuration, which allows us to avoid typical antipatterns and maintain a consistent style.

Here is our ESLint report, showing 0 issues:

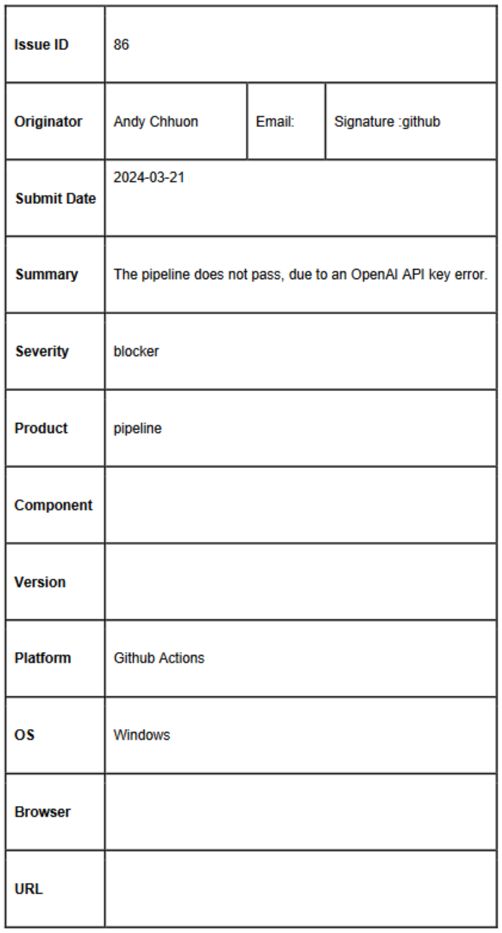

```
ESLint Report
0 problems - Generated on Wed May 01 2024 00:51:52 GMT-0400 (Eastern Daylight Saving Time)

[+] C:\Users\chhuo\Documents\SOEN390\condo-backend\src\app.ts                                  0 problems
[+] C:\Users\chhuo\Documents\SOEN390\condo-backend\src\aws\aws.ts                              0 problems
[+] C:\Users\chhuo\Documents\SOEN390\condo-backend\src\firebase\employeeManagement.ts          0 problems
[+] C:\Users\chhuo\Documents\SOEN390\condo-backend\src\firebase\financialService.ts            0 problems
[+] C:\Users\chhuo\Documents\SOEN390\condo-backend\src\firebase\firebase.ts                    0 problems
[+] C:\Users\chhuo\Documents\SOEN390\condo-backend\src\firebase\propertyProfileService.ts      0 problems
[+] C:\Users\chhuo\Documents\SOEN390\condo-backend\src\firebase\rentingService.ts              0 problems
[+] C:\Users\chhuo\Documents\SOEN390\condo-backend\src\firebase\userService.ts                 0 problems
[+] C:\Users\chhuo\Documents\SOEN390\condo-backend\src\index.ts                                0 problems
[+] C:\Users\chhuo\Documents\SOEN390\condo-backend\src\openai\chat.ts                          0 problems
[+] C:\Users\chhuo\Documents\SOEN390\condo-backend\src\utils\utils.ts                          0 problems
[+] C:\Users\chhuo\Documents\SOEN390\condo-backend\test\index.test.ts                          0 problems
```

9. Bug Reports

Sprint 3

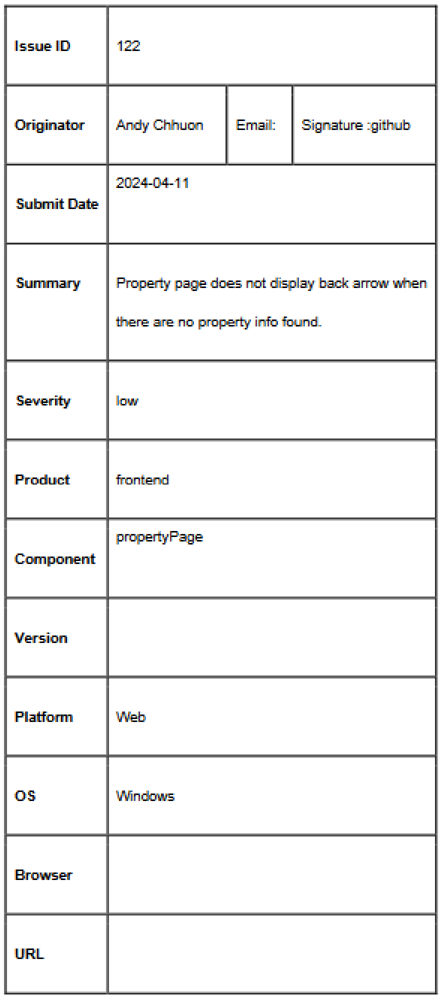
One bug was found and addressed this sprint:



Issue ID	86		
Originator	Andy Chhuon	Email:	Signature :github
Submit Date	2024-03-21		
Summary	The pipeline does not pass, due to an OpenAI API key error.		
Severity	blocker		
Product	pipeline		
Component			
Version			
Platform	Github Actions		
OS	Windows		
Browser			
URL			

Sprint 4

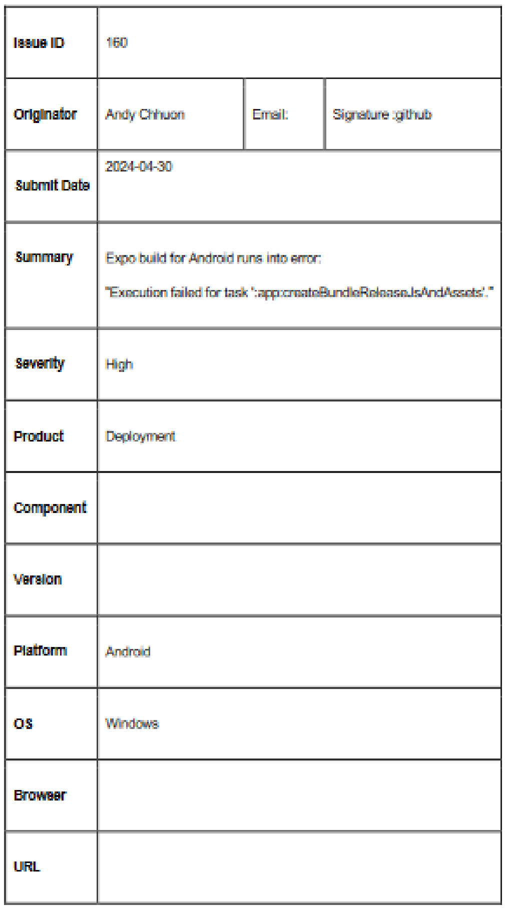
One bug was found and addressed this sprint:



Issue ID	122		
Originator	Andy Chhuon	Email:	Signature :github
Submit Date	2024-04-11		
Summary	Property page does not display back arrow when there are no property info found.		
Severity	low		
Product	frontend		
Component	propertyPage		
Version			
Platform	Web		
OS	Windows		
Browser			
URL			

Sprint 5

One bug was found and addressed this sprint:



Issue ID	160		
Originator	Andy Chhuon	Email:	Signature :github
Submit Date	2024-04-30		
Summary	Expo build for Android runs into error: "Execution failed for task ':app:createBundleReleaseJsAndAssets'."		
Severity	High		
Product	Deployment		
Component			
Version			
Platform	Android		
OS	Windows		
Browser			
URL			

10. Complete Retrospective

10.1. Sprint 1 Retrospective

Introduction

This first retrospective takes place at the end of the first sprint for the development of a condo management system (CMS). During the sprint, we produced mostly all the necessary documentation and planning for the project. We also developed a log in/sign up page for users as well as the corresponding profile pages. We are still in the early development stage and so we mostly focused on documenting, planning and developing UI prototypes for future sprints.

The expectations when the sprint began were high as we thought we could finish all our tasks and possibly advance more in the project because we knew that it is a heavy project. However, upon starting to work on the project, we realized it was very heavy on documentation. We did spend time on some of the technical aspects of the project by developing some pages as well as developing UI prototypes for the next sprint, however, most of our resources were spent on documentation and planning for the coming sprints.

As for the results, we managed to complete all of our tasks and so the results are satisfactory. As a team, we have also become more familiar with the project and between team members and so we are confident to be more productive in the next sprint.

For the development of the CMS, we have been using GitHub to collaborate as well as Jira to manage all the tasks to do. As mentioned previously, we mainly focused on documentation and so we did not use that many tools, but we do have plans to implement pipelines for automatic testing. We have also decided to use firebase to store our data because it allows us to abstract away the authentication process, making it faster to implement

What went wrong

1 - Time management of the team

The team struggled with time management because we started working on the project later than what would have been ideal. This happened because the channels were not very active and the team members had other priorities at the time. It was the beginning of the semester and everyone was still adjusting to their courses and workloads.

The impact of this was that we had a lot less time to produce all the deliverables required for sprint 1. We addressed this issue by communicating more on slack and setting up meetings to determine what to do as well as just using the text channels. As for the future, we know now to meet early in the sprint to make sure everyone is on the same page and knows what to do.

2 - Late Task distribution

Another problem we had was that we did not know what to do initially. We waited to meet with our TA and did not discuss or distribute the tasks immediately after that. This slowed the production of our deliverables. This happened because we did not initially take the initiative to plan.

The impact of this was that we were slowed down. We addressed this by starting to use a management software where each person assigned themselves to a task. In the future sprints, we know we can do better by planning before the sprint starts. We have already done that for sprint 2 and we have high hopes it will go smoother.

3 - Availability constraints within team

The team also struggled with availabilities. Our team is comprised of 10 members with different schedules and different workloads, we found it initially very hard to find times to meet where everyone was available which caused us to have our first meeting very late.

The impact of these availability constraints was that we could not start working and it created confusion as to what to do. We addressed this by having to make compromises with our schedules in order to meet as well as using our slack channel more. We could have been better at planning for these meetings, but now we know and we will be communicating through slack, smaller team meetings as well as setting team meetings in advance.

What went right

1 - Using Jira (management software)

As a team, we initially struggled with the distribution of tasks. We decided to start using Jira because it is a tool very popular and commonly used in professional settings, and it is very good at displaying what we need to do, what is being done and what is done.

The addition of this tool had a positive impact with the team because it provided them with a clear overview of what was to be done. It was also very easy to assign oneself to a task as well as see who was doing what. Most of the team was able to easily use this tool. As for what could have been done better, we could possibly write the tasks in more details for future sprints.

2 - Team initiative in little time to finish work

Another positive aspect was the team initiative when we realized we did not have much time left in the sprint. Most of the team really put their all and worked hard to get the deliverables ready by the deadline. This happened because we started discussing things more and the project was already being set up through Jira and GitHub to facilitate collaboration.

The impact of this initiative is that it allowed us to communicate more between team members, helping each other out and clarifying any confusions because we all became more involved. We do think that we could have done this before, but we are glad to have found this rhythm and have higher expectations for the next sprint.

3 - Setting up codebase to facilitate the work between team members

We were able to setup the codebase on GitHub, with all the configuration for backend and frontend done. The pages were setup in a way where we can develop in parallel without impeding on one another, which has made development a lot easier and faster. We could work on further separating the code into components to increase cohesion and our ability to work in parallel.

Conclusion

By the time we finished Sprint 1, we learned about all the preparations needed to develop an application. We did some development during this sprint, but we mostly focused on the documentation, diagrams and planning for the whole project. It was a very documentation-heavy sprint, but all team members have a better understanding of what the system has to do as well as all the tasks we will have to do in order to complete development.

We also learned how to work in a big team. Although we struggled in the beginning because of schedule conflict and poor time management, we were eventually able to get a good momentum going and we learned to work efficiently with each other. It is not often that we get to work with so many people in a school setting and it brings a lot of challenges, but it is something similar to what a real life job would be like and it is providing good practice for our futures. We were able to work well together through better communication as well as management and collaboration softwares.

As the main takeaway, we learned that communication within the team is essential. With all the different people in the team, we have different strengths and it has been working very well to ask in the slack channel and receive help from the people who can help in the team. We aim to continue this type of communication as well as prioritizing it in future sprints to ensure a smooth development.

10.2. Sprint 2 Retrospective

Introduction

As Sprint 2 is ending, many things became more clear. The team has a lot of strong qualities like being able to deliver high-quality work in a short period of time and helping each other in times of need, which are qualities that must be preserved. On the other hand, the team lacks some small behaviors that would highly increase efficiency if addressed correctly for the remaining sprints.

What went wrong?

All 3 of the following sections are intertwined together, since if the task distribution is done at the very beginning of the sprint's timeline, there would be an overall better time management. Also, if every member would attend the team meeting corresponding to the task distribution, it would be clear from the beginning what is expected from each member.

1 - Late task distribution

This section was also a problem seen in the last sprint. The team meeting specifically created for the task distribution was done a bit late in the timeline of this sprint. This could be due to leaving a couple of days of rest before starting to think again about the project. One way of tackling this issue is to have a meeting within 24 hours of the previous sprint deliverance. In that meeting, the team would go over all the necessary tasks, make sure no tasks are missing and divide all the tasks within members. Then, there would be a few general meetings to see how each task section is going and to enhance the work environment.

2 - Time management

There is a slight problem with the time management of the overall team, but this is mainly due to the late task distribution. Even though it went a bit better than the previous sprint, it is because of the reading week that helped taking a break from the rest of the work and focus more on getting the work done as early as possible. Another problem was the confusion of the due date, at the end it turned out to give us more time to get the tasks done. A solution to this problem is to set personal due dates to tasks which would make sure tasks are done a couple of days before the sprint's deliverance date. Also communicating more between members of a small task section to make sure no one is delaying the other.

3 - Showing up to team meetings

Not all meetings required all the members to attend, some meetings were held between the documentation team or between the backend team or the frontend team. In these meetings, only the members of the specific section are required to meet and discuss, but when it comes to the general team's meetings, every member of the team is required to attend and participate. This was an issue since we would have a consensus on the date and time of the meeting, but not all members would show up. We would end up postponing the meeting and come up with another date for the meeting.

What went right

1 - Using Jira

Jira is a great way of communicating for teams. Everything takes place in Jira except physical meetings are done on Zoom. The task division feature helps a lot to divide the tasks and know who is doing what without the need of wasting time asking each other all the time.

2 - Availability constraints

For this section, there were a lot of improvements compared to the previous sprint. When asking about each one's availability to have a meeting, someone would always create a poll on Jira, and we could always find 1 or 2 slots in which every member was available.

3 - Good communication between members

In general, there was good communication between task sections to understand what needs to be done between each other and between members of the same tasks section. It seems that members are interacting well with each other, making sure everyone knows what they are doing and helping in times of need.

Conclusion

For the next sprint, the team should work on assigning tasks at the beginning, develop the timeline of sprint 3 and analyze how the rest of the issues develop since it seems to be the root cause of most issues.

10.3. Sprint 3 Retrospective

Introduction

This retrospective seals the third sprint for the development of a condo management system. During this sprint, we advanced in the front-end, back-end and more importantly in the testing. We focused on developing a reservation system which would allow users to reserve a facility like the gym or the pool, on an employee role management system which would organize the two types of employees and assess their roles and responsibilities. Lastly, we developed a request management system which would allow the communication between the users and the employee.

With those, testing was the most important section that needed to be completed since our progress for testing was slow and behind the planned timeline. As for the results of this sprint, we successfully completed many tests all the while advancing as planned for the implementation of the new features. This made us gain back our confidence and assured us that we will be able to finish the project, with all its planned features, in time.

What went wrong

1- Time Management

This is the only problem that persists until now. The team in general struggles with time management, unlike previous sprints. This time it was due to lack of knowledge towards the due date and that there was a lot to do in a relatively short time. In the future, we will assign someone to remind the whole team of the due date every here and there, so that it doesn't come as a surprise a few days before the deadline.

2- Implementation

In this sprint User Stories 4 and 7 were implemented while all user stories until 7 (included) should have been implemented by the end of the third sprint. This is due to time management and underestimating the load of work for the implementation. For the next sprint, we will need to implement user stories 5,6 and 8, so that in the last sprint will be less of a load than the coming because realistically speaking we will have much less time during the fifth sprint due to finals.

What went Right

1 - Using Jira

Nothing changed in that section. Jira is a great platform for task divisions.

2 - Good communication between members

For the most part, there was good communication between the team, whenever someone would request a meeting, most members would answer right away. There was always a team member that would answer right away when someone asks a question. At each conversation, we made sure that everyone understood what was said, no one would come out not understanding what was said.

3 - Task Division and Meetings

As stated in the previous retrospective, this section was the main cause for everything that went wrong last sprint, but it was fixed right away. The team had a meeting right after the sprint 2 demo and talked about the vision for the future sprints and the tasks that needed to be completed for this sprint. Then, the tasks were divided among members and a summary was written on the team’s chat for those who could not attend the meeting.

4 - Tests and Code Management

As tests were very lacking in the previous sprints, more members were assigned to write code tests in this sprint. This allowed an increase in tests compared to the previous sprints. Also, being aware of the mistakes we made previously, all reports, whether for the documents or for the codes, were clearly written. This includes testing, code style and quality, and bug reports. This allows for any reader that was not part of the code development to understand what the code is, and what's good/bad about it.

Conclusion

In this sprint, we saw the benefits of having a meeting at the beginning of the sprint, filling in everyone about the tasks that need to be completed by the end of this sprint and dividing the tasks in that moment. That gave everyone the full number of days to complete their part without being confused or left without a task. On the other hand, we misjudged the load of work that was ahead of us. For the future, we would consider the amount of work a task takes and increase its planned time by a couple of hours/days.

10.4. Sprint 4 Retrospective

Introduction

This retrospective seals the fourth sprint for the development of a condo management system. As the semester is coming to an end, so is our project and the final submission of our semester long product. This marks the end of the fourth sprint and the start of the fifth and final sprint.

By now, we should have implemented all our features and hosted the website according to the initial release plan. As the sprints went by, we were falling short on some work and ended up with having to implement 3 features in this sprint instead of 2. Thankfully, we were able to implement all 3 features which are the development of a reservation system which would allow users to reserve certain products like the pool or the game room, an employee role management system which would allow to organize employees into 2 categories customer service and financial, and a Notification page which would store all recent notifications received. Above these, we progressed in our testing and in the organization of the documents.

What went Wrong

1 - Time Management

By now, this has been the only section that is always lacking. Unlike the previous sprint, where it had gone a bit better, we did not set up a team meeting right away after demonstrating the previous sprint, we let close to a week pass before scheduling our first team meeting for the fourth sprint. This, therefore, gave us the opportunity to only meet one other time for this sprint. In the last sprint we should do as we did in the previous sprint and meet right away, in the following days of the sprint 4 deadline.

2 - Tasks Planning and Divisions

This section goes in hand with the previous sprint, but it is more important in the upcoming sprint since some tasks have been removed and new tasks have been added. Task planning and division is usually done in the first team meeting, but even though every team member has a position, we tend to overpower one team section while leaving another more important section with very few workers. This issue should be addressed in the following sprint to complete the heavier tasks ahead of time instead of stressing to submit some work before the deadline.

What went Right

1 - Updated with the documents' submissions

Due to a misunderstanding, in the first 2 sprints, we were missing some documents or in more precise terms, our documents were not well organized. We started organizing our documents in sprint 3 and had to do some modifications and improve the writing skills for these documents which leads us to the submitted documents for sprint 4 in which all documentations are placed in well-labeled documents/files and no required information is missing.

2 - Tests and Code Management

This section has improved since the last sprint, we increased test cases and widened our test coverage, all reports, including the code management document and the bug report, have been updated with the work of this sprint. This allows anyone who wants to be informed with our work to skim through the code and read thoroughly our documents and they should understand the coding with ease.

3 - App Hosting

We achieved a very important event in this sprint which was hosting our app/website over the internet. This success would not have been realized without working well on the frontend to have a presentable, pleasing to the eye, product to the users. Also, would not have been possible without a tailored backend which is linked to the frontend and allows the user to perform certain activities. The hosting is an important achievement in our project, and it is even more important that we were able to do so without tardiness to the planned release date.

4 - Compromising the lost time

Even though we were late in the meetings and task divisions for this sprint, we were able to work efficiently to complete all the required works for this sprint and the missing works from the previous sprint. This leads us to be on track with the overall timeline of the project which is also another great achievement for the team.

Conclusion

Even though we repeated some of our mistakes in this sprint after dealing with them in the past, we were still able to get the job done and developed the planned features and the complementary tasks. The plan for the next sprint is to meet at the beginning of it, decide on the plan for the upcoming sprint and set fake deadlines to guide us not to leave everything to the end. We will mainly focus on updating and enhancing all the existing features and if time permits, we will implement some of the additional and bonus features. Overall, we matured in comparison to the beginning of this project, each learned to develop in their own way to bring more to the table, yet no one is perfect and therefore we still commit the same errors sometimes. We learn and move on; we will focus on those errors in the last sprint and hopefully it will make us more resilient in the future.

10.5.Sprint 5 Retrospective

Introduction

At the writing of this retrospective, the team has completed the work for the fifth and final sprint of this project. By now, all demanded features have been implemented and are functioning. Above that, the test coverage is either close or above the 80% mark.

What Went Wrong

1 - Time Management

As previously mentioned, this section seems to be the main problem in our group. Between scheduling team meetings, discussing the matters and coming to a conclusion, we never end up having a lot of time in our hands to progress at a comfortable rate. This forces us to stress over the matter and cram everything at the last few days before the submission deadline. Add to it the finals season that each student was going through, it really did not help fix this problem, as a matter of fact it amplified this issue.

2 - Communication

This section was always one of the group's best characteristics. Most members of the group used to be very active on the group chat and always stayed updated with the latest decisions. Unfortunately, Sprint 5 was in the middle of the finals season which caused our communication to weaken since everyone was more focused on their final exams and would not be consistently checking the group chat. This had a negative impact on our time management and the delay of the completion of tasks.

What Went Right

1 - Task Division

The division of tasks was done right away on our first team meeting for this sprint, everyone had a task and we focused on completing the mandatory features rather than dividing ourselves and trying to complete the additional and bonus features too. This was definitely a good idea since we were able to complete all implementations, all documentations, and all testing as it was demanded.

2 - Implementation

For this sprint we had planned to complete the US 8, 9 and 10, but we were also behind on some features, mainly for US 5 and 6. Taking the advice from our teacher and TA, we decided to focus on perfecting the mandatory US so 5, 6 and 8. While working on those, it happens that we were able to implement a small part of the bonus/additional features. So, maybe we did not reach our goal of implementing all the additional features, but we were able to complete all the mandatory ones.

3 - Testing

The testing section went very well this sprint, just like last sprint. One difficulty we did face is that the tool we used to calculate the percentage of coverage was making mistakes and it would not count some of the implemented tests, therefore the percentage was lower than the actual coverage. We ended up being able to overcome that obstacle and developed some many tests since we were also a bit behind on the testing, but now it shows all the test coverages and we have excelled the 80% coverage mark as demanded.

4 - Documentation

All documentation was completed by our previous sprint. Up until sprint 3, we were missing some documents which we were unaware of, but after reading the comments given by the teacher and understanding which documents we were missing and which documents needed to change format, we worked on those on sprint 3 and 4 and we were able to complete them all.

For this sprint, it was a matter of adding the sections for the end of sprint 5 and the end of this project. By now, all documents have been updated to include the sections of sprint 5.

Conclusion

In conclusion, this sprint went very well in terms of tasks and completion of features. We were able to complete all development concerning the mandatory features, partially implement one or two of the additional/bonus features, attained the 80% overall test coverage and completed all required documentations. On the other hand, the biggest problem was, surprisingly, communicating with each other, which we proved to be able to overcome any obstacle.

10.6.Overall Retrospective

Intro

The Condo Management System project consisted of 5 deliverables (sprints) in which we had to plan accordingly which features will be implemented in order that by the end of the fifth sprint, we would have at least completed the mandatory tasks. In each of the sprints, we will quickly go over the obstacles and what we learned.

Summary

In sprint 1, it was all about planification and taking the first step into the project, implementing very basic functions. The obstacles faced were 1- getting to know each other and 2- creating a health dynamic between each other.

In sprint 2, we started implementing some features and started working strongly on documentations. The tasks were not yet officially confirmed on that point, but we had an idea about the strengths and weaknesses of each team member. The obstacle we faced in this sprint was late task division which we recovered from in the following sprint.

In sprint 3, each person had their official post on the team, discussions in place had to do with planification and how to divide the work within each subset in the team. In case, a certain area needed more help, teammates from other subsets would join to fill the lack. The obstacles we faced in this sprint was the clarification of our documents and the lack of test coverage.

In sprint 4, we had our biggest development phase. In this phase, we implemented most of the features and enhanced some of the existing features, while also enhancing the documentation and adding the missing documents and improved how test coverage for some features. It is also in this sprint where we solved part of our time management problem by meeting right away at the beginning of the sprint timeline. Overall, it was a very strong sprint.

In sprint 5, we completed all the mandatory and remaining features while continuing the tests and reaching a test coverage above 80% as assigned and completed all the documentation each document according to its kind and fully written without any missing parts. The obstacle of this sprint was keeping a consistent communication with each other while each going through the struggle of the final exams season. Even though it did affect our communication, our performance remained the same and we were able to deliver the project as it was expected from us.

Conclusion

In conclusion, this project was very beneficial on many levels, not only academically but also on a personal level. We truly believe that we have learned a lot coming out of this project. We are not the same people as in the beginning. We learned to cooperate with each other without knowing anything about each other beforehand. We learned to discuss and understood the importance of meetings, especially the long ones. We learned and experienced the different phases of a software project, from having a vague guideline to writing documents to finally releasing our final project. We came out to become more resilient and that on its own is enough as an advantage to our future projects.

References

[1] ‘Domain Modeling: What you need to know before coding’, *Thoughtworks*. Available: https://www.thoughtworks.com/en-ca/insights/blog/agile-project-management/domain-modeling-what-you-need-to-know-before-coding. [Accessed: Feb. 07, 2024]

[2] ‘IBM Documentation’, Jul. 18, 2023. Available: https://www.ibm.com/docs/en/dma?topic=diagrams-component. [Accessed: Feb. 07, 2024]

[3] ‘IBM Documentation’, Mar. 04, 2021. Available: https://www.ibm.com/docs/en/rational-soft-arch/9.6.1?topic=diagrams-use-case. [Accessed: Feb. 07, 2024]

[4]‘What is Activity Diagram?’ Available: https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-activity-diagram/. [Accessed: Feb. 07, 2024]

[5] ‘Example: Software Architecture Document’. Available: https://www.ecs.csun.edu/~rlingard/COMP684/Example2SoftArch.htm. [Accessed: Feb. 07, 2024]

[6] ‘GRASP (object-oriented design)’, *Wikipedia*. Jan. 08, 2024. Accessed: Mar. 03, 2024. [Online]. Available: https://en.wikipedia.org/w/index.php?title=GRASP_(object-oriented_design)&oldid=1194382900

[7] ‘Deployment Diagram Tutorial’, *Lucidchart*. Available: https://www.lucidchart.com/pages/uml-deployment-diagram. [Accessed: Mar. 21, 2024]

[8] T. A. Gamage, ‘Answer to “Having trouble with deployment diagram”’, *Stack Overflow*, Mar. 18, 2017. Available: https://stackoverflow.com/a/42874052. [Accessed: Mar. 21, 2024]

[9] ‘Scaling the Practice of Architecture, Conversationally’, *martinfowler.com*. Available: https://martinfowler.com/articles/scaling-architecture-conversationally.html. [Accessed: Mar. 22, 2024]

[10] DoorLoop - "10 Best HOA and Condo Property Management Software (2024 Guide)" from DoorLoop. Available: https://www.doorloop.com/blog/best-hoa-condo-property-management-software

[11] Software Advice - "Condo Manager Software Reviews, Demo & Pricing - 2024" from Software Advice. Available: https://www.softwareadvice.com/property/condo-manager-profile/

[12] Yardi Breeze - "Best Association Software For Condo & HOA Managers (2023 Update)" from Yardi Breeze. Available: https://www.yardibreeze.com/

[13] ButterflyMX - "Top 8 Condo Management Software You Must Try This Year" from ButterflyMX. Available: https://www.butterflymx.com/blog/condo-management-software/

Archive-Google drive link

Here is the link to the archived files in the google drive folder:
https://drive.google.com/drive/folders/1rvRaEksFKITsEVbeRDuqCnohiBFtMS8r?usp=sharing