Software Design Document

CoRec Occupancy Tracker

By Team 7
Ethan Lee
Daniel Shi
Matthew Zhang
Shen Wei Leong

Table of Contents

Purpose

Many customers of the corec want to know how occupied the corec is at certain times of the day so they can plan their workouts at times where the corec is at low occupancy and less busy than usual. However, it's currently very hard to gauge how occupied the corec will be at certain times. The purpose of this project is to utilize Computer Vision and human detection to accurately keep track of the occupancy rate of each of the different rooms of the corec and analyze the equipment usage/availability in real time while also using those statistics/data to both graph past corec occupancy over different times of the day and predict corec occupancy at future times of the day.

Design Outline

- Design Decisions
    - Client-server model
    - REST API architecture (using Flask)
        - Service layer
            - Processes camera footage, computes occupancy counts
            - Contains business logic for handling users of the platform
            - Performs CRUD operations on Records as well as sends notifications to users
    - Components/Classes:
        - Objects
            - User
                - Object representing a user of the system
            - Notification
                - Object representing a notification
            - Record
                - Object representing a record storing information about the occupancy of a particular corec room at a certain time
        - Service Classes
            - UserService
                - Used for managing users of the platform (e.g. adding users, fetching users, deleting users, updating user info, etc)
            - NotificationService
                - Used for sending notifications to users
            - RecordService
                - Used for keeping track of occupancy records and performing CRUD operations on record objects
            - FacilityUsageService
                - Calculates the number of people in each room of the corec using footage captured by the corec's cameras
                - Predicts occupancy of corec rooms at future times of the day
            - CameraService
                - Every so often, the server will send a request to this service to take a snapshot of each corec room and generate an image file. It then calls FacilityUsageService using the path of the snapshot and generates a Record object.

Design Issues (chosen solutions are marked as bold)

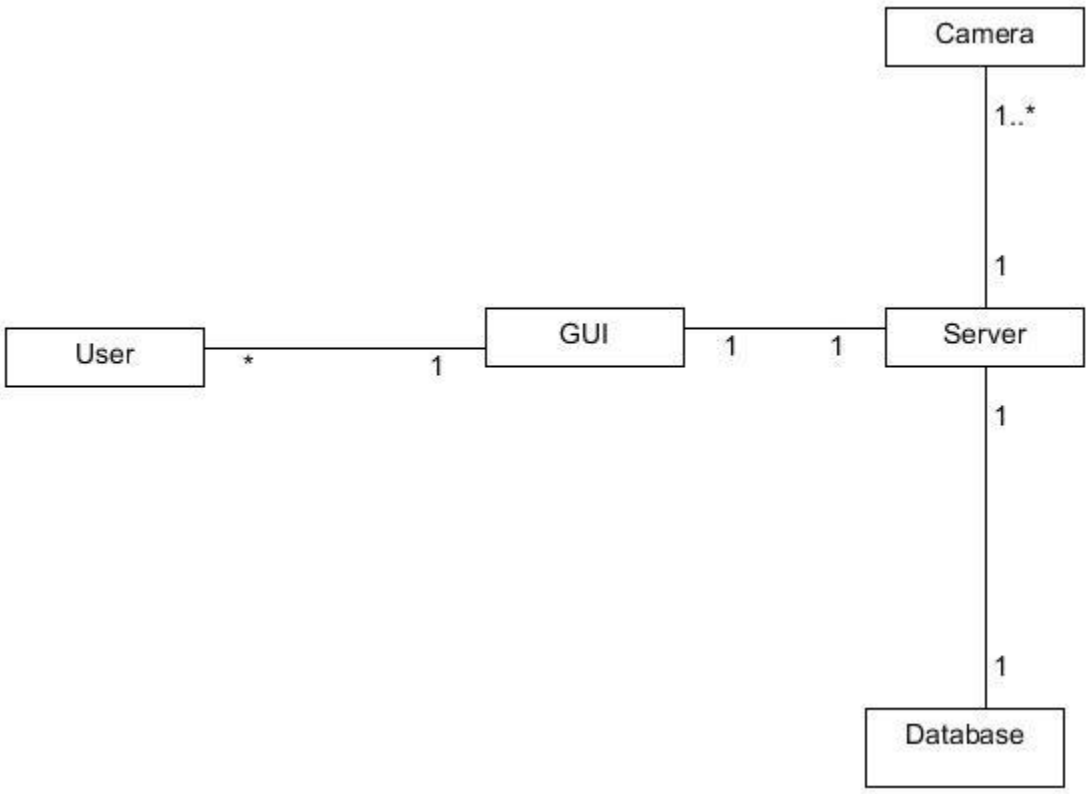
Functional

1. How will occupancy data be displayed?
    a. Solution 1: Use a line graph to display data. Each area or set of machines will have a line graph that should update every 5 minutes during Corec open periods
    b. **Solution 2**: Use bar graphs to display the predicted occupancy of a room or set of machines.
        i. We pick this solution because bar graphs are one of the most easily digestible graphs. Users will be able to easily see and analyze the occupancy trends throughout the time of day while also being provided with numeric values.
2. Users can get to separate floors of the Co-Rec to find their desired workout area as fast as possible
    a. **Solution 1**: I would like to separate each floor to have its own individual webpage. This would allow any user to click on the floor they know their machine or area is out to quickly figure out if it is full.
        i. Separating the floors would make organization and accessibility easier for the user
    b. Solution 2: I could also display all of the information on a single web page. This would decrease any slow-downs due to slow internet connection. We could still separate each floor with headers and display the information of the areas and machines under each floor.
3. I want to "clean" the record table in the database every so often so that the table doesn't become full and cause a reduction in performance
    a. **Solution 1:** Have occupancy records expire after 2 weeks. Every 24 hours, delete all records in the database that are over 2 weeks old.
        i. Since the prediction portion of our app relies on historical data, we don't want to delete recently recorded data. At the same time, having records that are super old will end up overcrowding the database and lead to slowdowns in the app. We believe that gathering data within the last two weeks will give us pretty accurate predictions, and data that's any older than that will be extraneous and may possibly cause a skew in the data. In addition, solution 2 would cause delays in updating current occupancy count when the number of records in the database becomes really large, and since we want to have our app update as close to real time as possible, we want to minimize delays.
    b. Solution 2: Keep all records without deleting.
4. How will users get notified when a room is not busy?
    a. Solution 1: Have a built-in notification system inside the app
    b. **Solution 2**: Receive notifications via email and/or SMS
        i. Solution 2 is better because with solution 1, users won't be alerted if they don't have the website open (for desktop). Solution 2 ensures that users will always be alerted regardless of where they are and whether or not they have the app open. This is, of course, assuming they have some sort of electronic device with them which 99% of people will.
5. What happens when there's overlap in an area (two cameras scanning a singular area with overlapping regions)?
    a. **Solution 1:** Find a singular camera that captures the largest portion of a given area. We then estimate the percentage of the room that the camera is capturing. We then take the number of people that the camera successfully captures and multiply it by *1 + (1 - percentage)*. For example, if we estimate that a particular camera captures 80% of a room and records 80 people, the occupancy count that is displayed to the user will be *80 * 1.2 =* ***96***
        i. With a singular camera, we won't have to account for overlap and it'll make our calculations easier
    b. Solution 2: For each area of overlap, assign a constant value to that area. Add the number of people captured in the relevant cameras and subtract the sum by that constant to get the "actual" occupancy count.

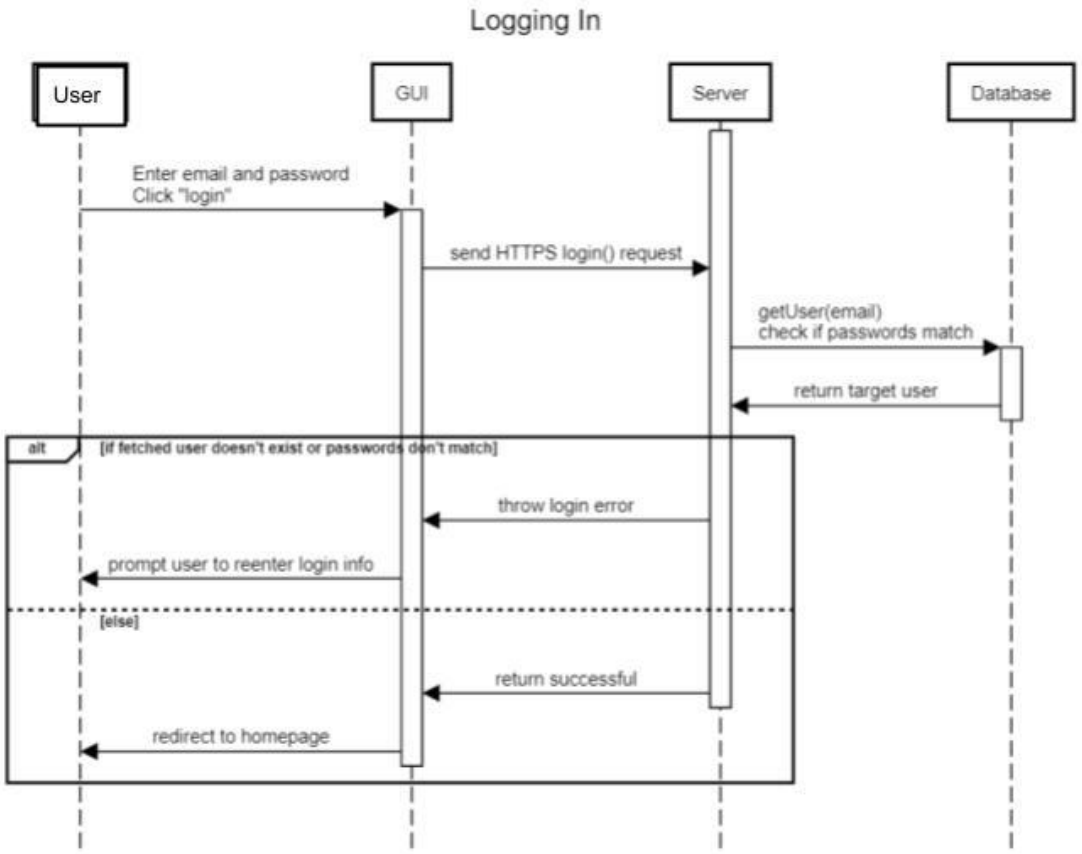
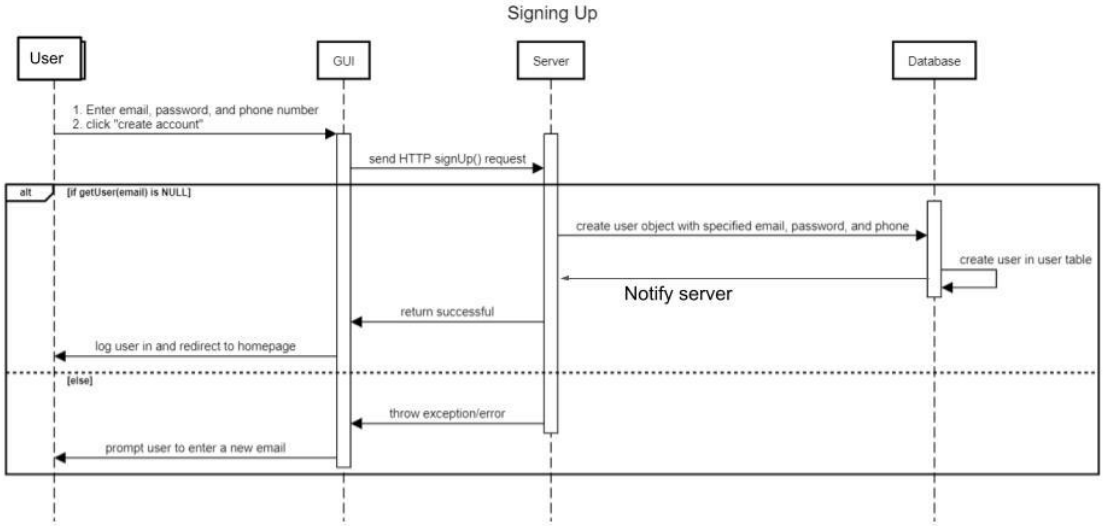
Non-functional

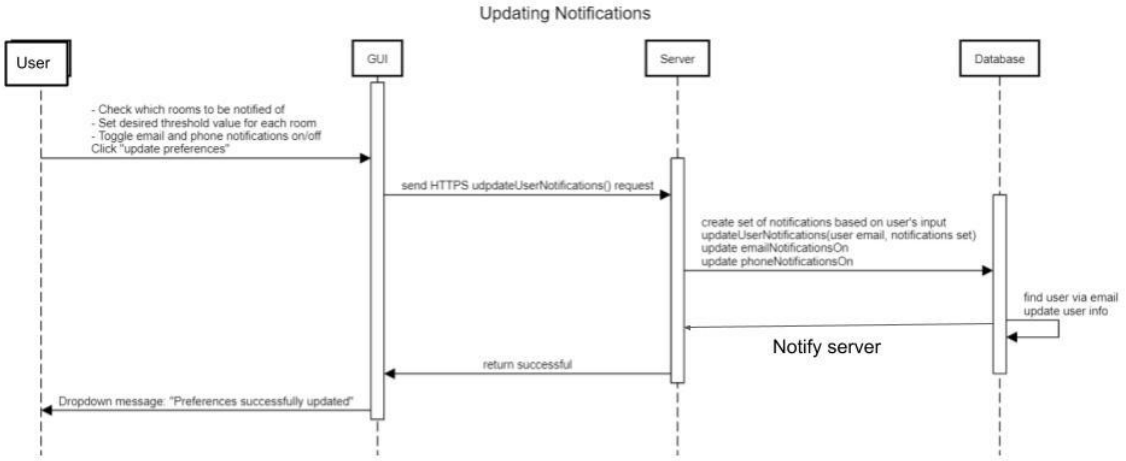
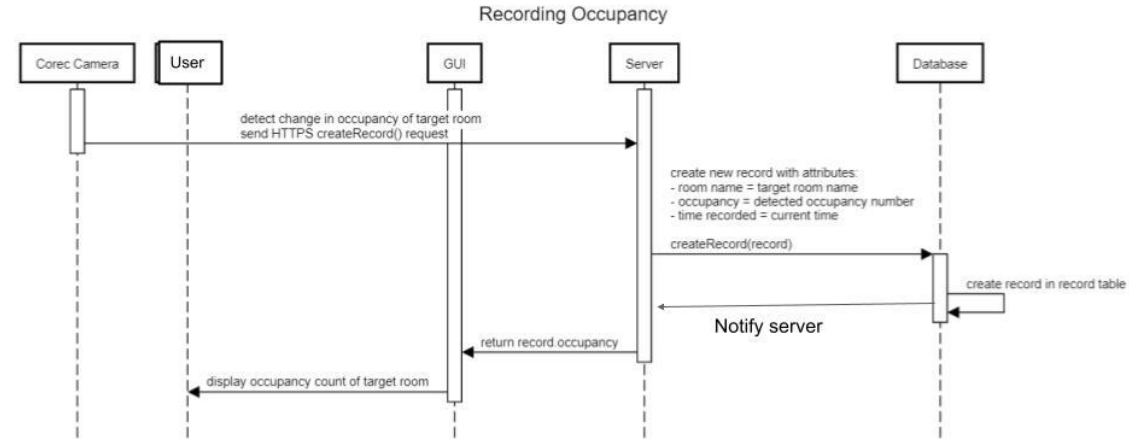
1. Performance
    a. How will the information be available in real-time when a user requests it?
        i. Solution 1: Update the occupancy count in the UI as soon as the corec cameras detect a change in occupancy in any of the corec rooms
        ii. **Solution 2**: Send requests to the server every couple of seconds and have the backend services compute and return the current occupancy
            1. Solution 2 is better because it's hard to program the cameras themselves to detect change. The cameras are just there to record so they don't have any detection capability.
    b. I would like our program to be able to accurately count the number of people in a certain area (around 90-95% correctness)
        i. **Solution 1:** Select cameras that are placed in locations that have as little obstacles as possible
            1. Solution 1 is better than solution 2 because if a person is too far from the camera, facial detection might not be able to capture their face. In addition, a room with little obstacles will yield more accurate occupancy values.
        ii. Solution 2: Use facial detection to detect a person instead of scanning their entire body
    c. I would like the prediction to be within an a range of 70-90% correct
        i. Solution 1: Make predictions by mathematically analyzing trends in past data *(e.g. in the past week, the rate of increase in occupation for the basketball courts was 5 people per hour between 2 pm and 5 pm. Assume it's 2 pm right now and the current basketball court occupation is at 45. The app will predict an occupancy of 60 people by 5 pm)*
        ii. **Solution 2**: For each corec area, take the average occupation count for each hour of the day and have that average be the predicted value for the corresponding hour *(e.g. the average number of people in the basketball courts at 5 pm is 60. Therefore, the system will predict that 60 people will be occupying the basketball courts tomorrow at 5 pm)*. We can accomplish this by accessing the occupancy records in the database and filter by the hour of the day. We then simply compute the average of the records we've obtained.
            1. From a coding standpoint, solution 2 is easier to implement than solution 1 because all we are doing is calculating the average which is a simpler operation.
2. Security
    a. I would like general users to be able to use the website without needing to use any credentials
        i. **Solution 1**: Upon opening the app, present the user with an option to continue as guest
            1. We picked solution 1 because not everyone has a Google account or wishes to use it for this app.
        ii. Solution 2: Have users login through Google
3. Usability
    a. As a user, I would like the website to be easy to use on both a mobile web application and a desktop application
        i. Solution 1: Have the app be a website so that users can access it through their mobile device's browser
        ii. **Solution 2**: create an iOS version of the app
            1. Solution 2 is better because users can create custom iOS notifications which are more effective than email/SMS

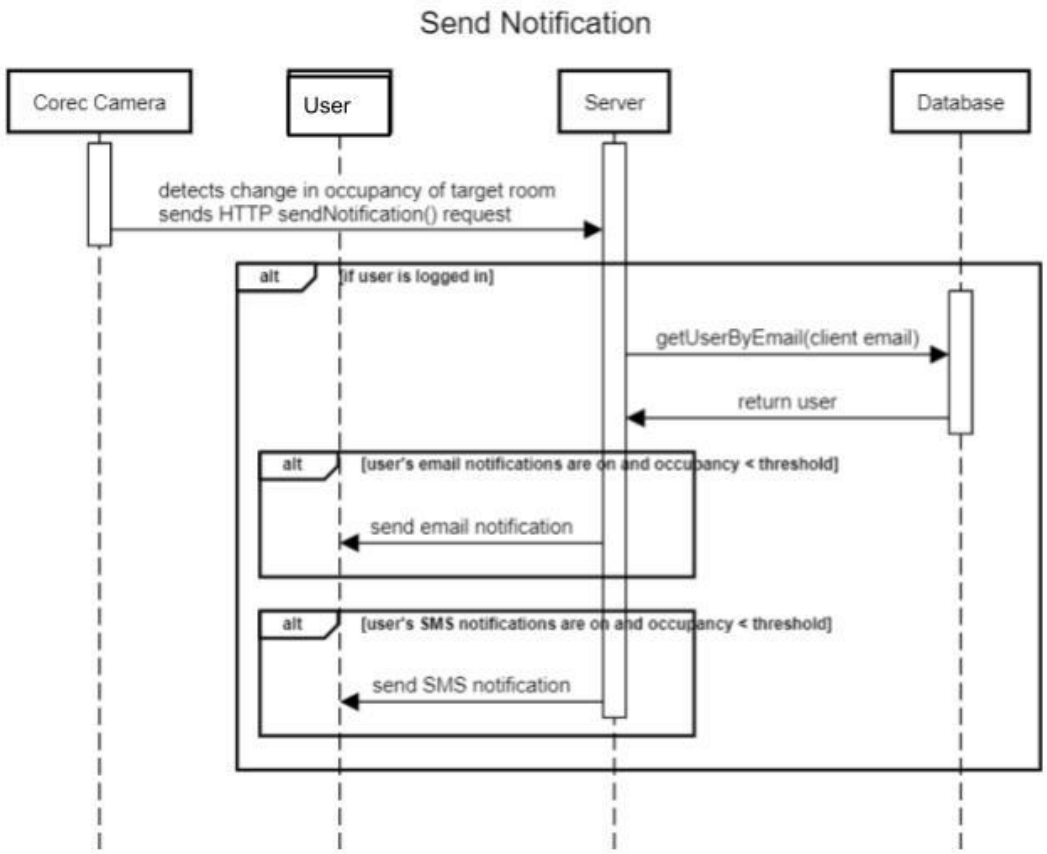
UML Diagram



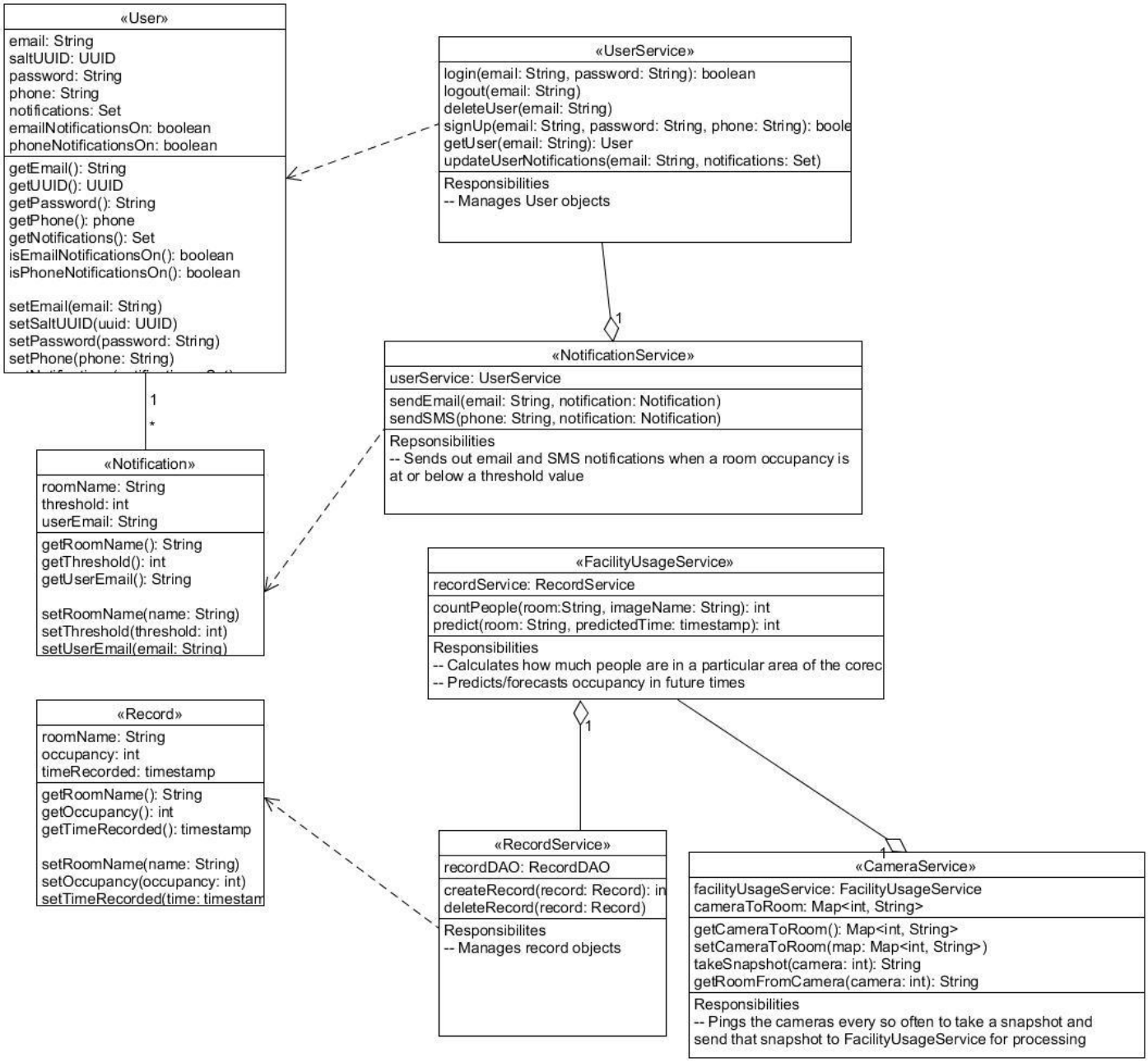
Sequence Diagrams

Activity	Diagram
Logging In	
Signing Up	

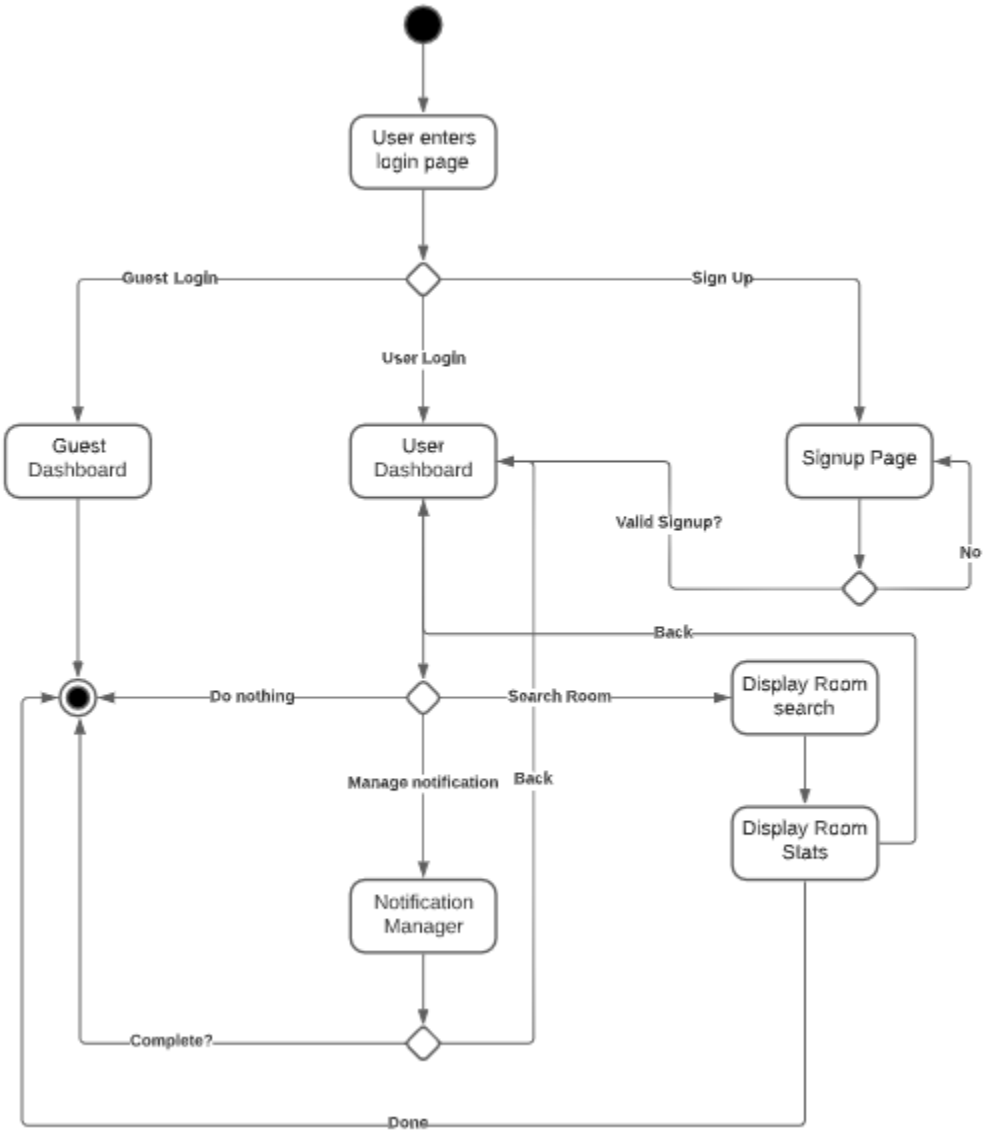
Updating Notification Preferences	
Recording/Updating Occupancy Values	

Sending notifications	

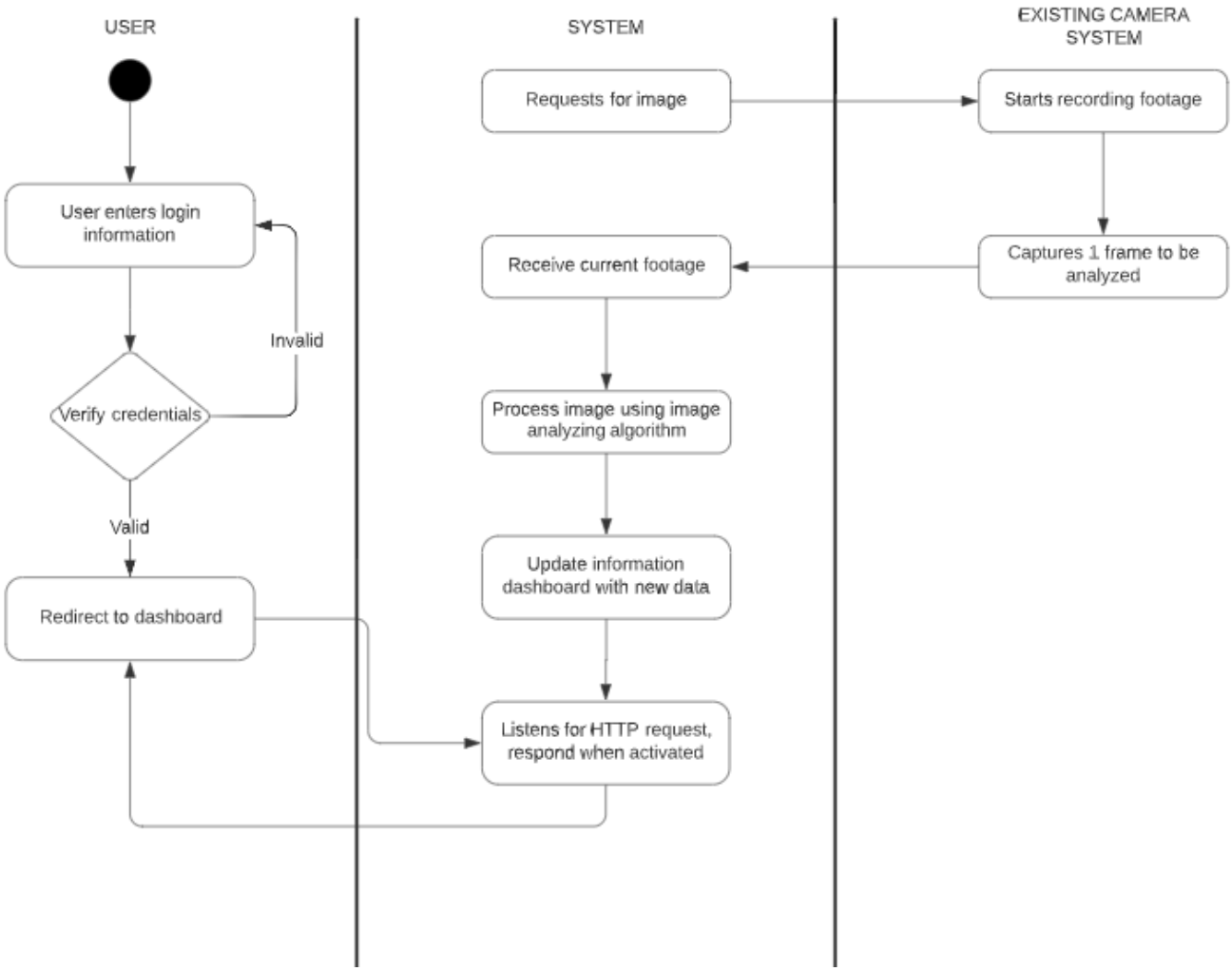
Class Diagram



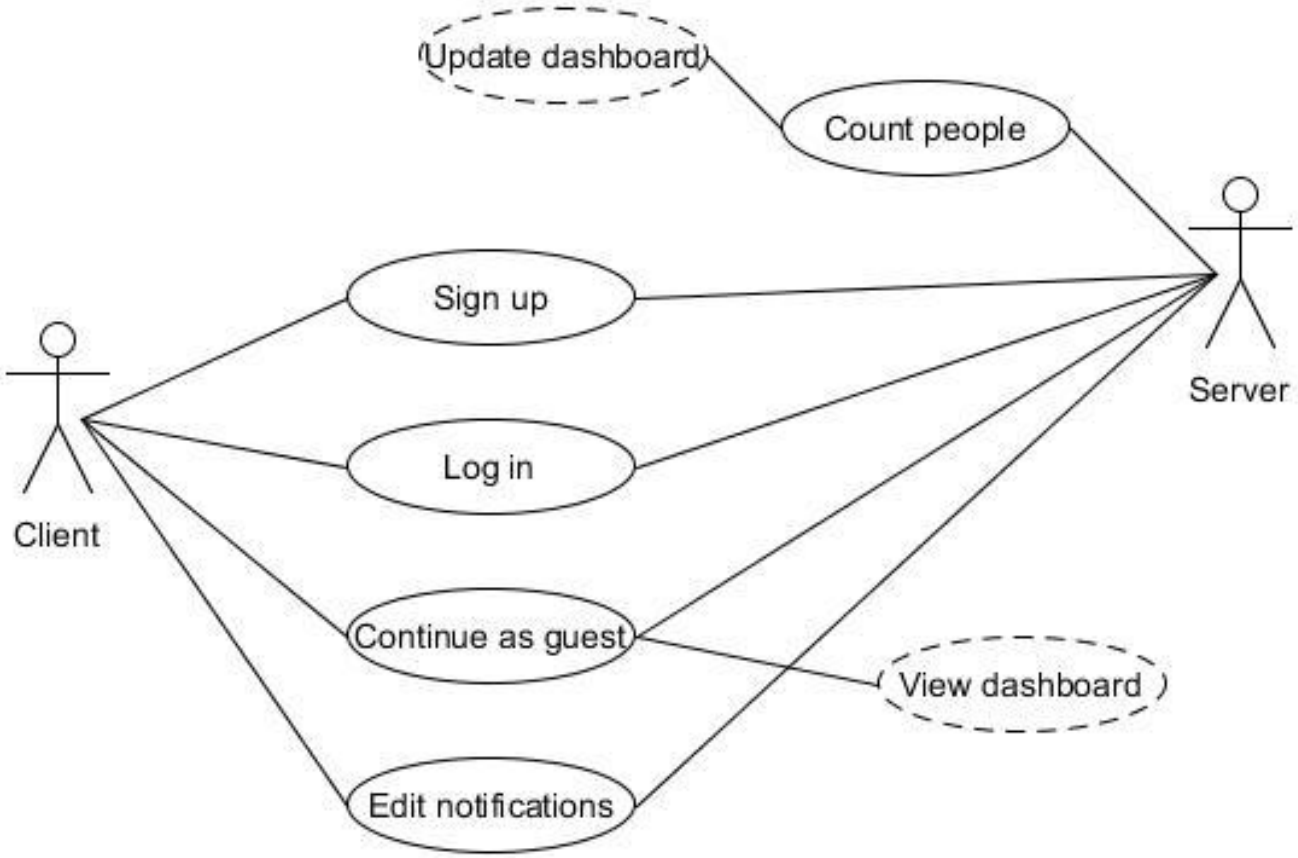
Activity Diagram



ACTIVITY DIAGRAM



Use-Case Diagram



UI Mockups

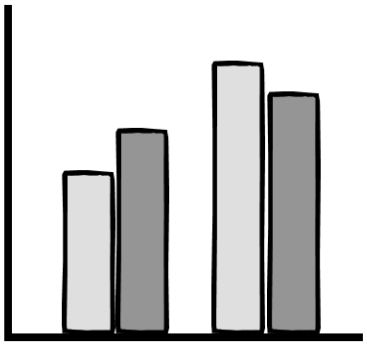
CoRec Tracker

- Log In | Sign Up | Continue as Guest

Basketball Courts | Gym | Pool | Track | More...

Basketball Courts

58



- < Back

Settings

Receive Email Notifications

Receive SMS Notifications

Receive Notifications For... Threshold

- ☐ Basketball courts 45
- ☑ Pool 10
- ☑ Gym 25
- ...

Receive Notifications Between

- 5:00 am and 11:00 pm