



# Happy Birmingham

Group Name: Happy Birmingham (Group 51)

Members:

Andrew Foot, 2177545, [adf045@student.bham.ac.uk](mailto:adf045@student.bham.ac.uk)

Stephen Rollinson, 2175845, [sxr086@student.bham.ac.uk](mailto:sxr086@student.bham.ac.uk)

Jiwon Park, 2226442, [JXP042@student.bham.ac.uk](mailto:JXP042@student.bham.ac.uk)

Oday Jawaada, 2246956, [ojj056@student.bham.ac.uk](mailto:ojj056@student.bham.ac.uk)

Ameena Masood 2041247 [axm1615@student.bham.ac.uk](mailto:axm1615@student.bham.ac.uk)

SoHyun Jun, 2170033, [shj033@student.bham.ac.uk](mailto:shj033@student.bham.ac.uk)

Omar Elalfy, 2252602, [oxe002@student.bham.ac.uk](mailto:oxe002@student.bham.ac.uk)

Dimitrios Kontolampros, 2166855, [dxk055@student.bham.ac.uk](mailto:dxk055@student.bham.ac.uk)

Wajahat Hussain, 2066920, [wxx920@student.bham.ac.uk](mailto:wxx920@student.bham.ac.uk)

<b>A. Requirements Engineering(Unit 2):</b>	<b>3</b>
A1.	3
A2.	5
<b>B. Software Design with UML(Unit 3):</b>	<b>7</b>
B1.	7
B2.	9
B3.	11
B4.	13
B5.	13
B6.	19
B7.	20
B8.	21
<b>C.</b>	<b>23</b>
C1.	23
C2.	24
C3.	25
<b>D.</b>	<b>26</b>
<b>E. Usability and Prototyping (Unit 6).</b>	<b>31</b>
E1.	31
E2.	33
<b>F.</b>	<b>34</b>

## A. Requirements Engineering(Unit 2):

### A1.

Our system is a two part server app system integrated with the university's physical hardware and cloud services. The backend server will monitor an array of sensors spread around the entire university birmingham campus. The application will provide a summation of some of the information gathered from the sensors as well as some computed data, such as graphs or predictions based on previous data.

The server will manage light sensors, thermal sensors, motion sensors, microphones, security cameras, moisture sensors, bluetooth receivers and RFID scanners. Each device will connect to a main server which processes and stores data for machine learning computation. The server will be built with concurrency in mind so that the system shouldn't crash if the front end is being used by many users and sends many different data requests, this can be done by utilising either multiple servers or by using cloud processing services.

The application will provide detailed information to users, both real time data from the sensors and computed predictions from the stored data. The app will be as simplified as possible, overlaying heat maps on a map of the campus to make the data more visually understandable. The application should allow users to track the graphed data of certain buildings and compare values from building to building.

The backend server will utilise smart thermostats to control the temperatures of rooms throughout the campus, keeping them comfortable when occupied but keeping them at a more efficient but less comfortable temperature when unoccupied. The server will combine the smart thermostats with smart windows to be able to ventilate rooms and regulate their temperature more without unnecessary air conditioning. The windows should also be able to be controlled from the application giving users in the room climate control and increasing their comfort. The server will monitor the light levels in every room on the campus using light sensors, this data will be used to regulate both the blinds in a given room but also the lights in a room. All bulbs on the campus should be dimmable so that they can be switched on but very dim when a room requires only a bit more light. The bulbs can then be brightened as a room darkens. Temperature sensors can also be used in conjunction with data on how much energy is being used to heat a room to determine how quickly a room dissipates heat. This dissipation data can be displayed in the app and rooms with the largest dissipation values can be insulated to reduce it's value which would result in an increase of the campus' energy efficiency. Each building should be equipped with solar panels to generate its own electricity. The server will monitor every building's energy generation and consumption, this data could be shown on the application. This data could be analysed by the server to determine which buildings should be focused on for more energy efficient projects in the future.

The main focus for the front-end is to provide an intuitional and useful app which is both available on desktop and mobile. We aim it towards the university students and teachers to help with their daily lives while also reducing carbon emissions to hit the goal of net zero. It lends the user-verification management system that is already in place for university students. It will work as an interface to the api that will be supported for the back-end while also taking features of the user bluetooth to give a more reactive system. When opening the app the user is displayed a map of the campus and can interact with it to gain information on: capacity of rooms, energy generation ,and temperature control. The UI will have selectable view models that will relay different information like a heat map of where there is a lot of traffic on campus and bar-charts of most energy efficient buildings. This will be used as a part of our scheme to have a student vote for the most energy efficient building that will be rewarded with a *free cup of coffee*.

Overall we want the project to be reliable and robust with many back-protocols for a fail case in either front or back end. This will be enforced by offline backup of data mixed with the security of the existing university user verification system.

## A2.

### Functional requirements

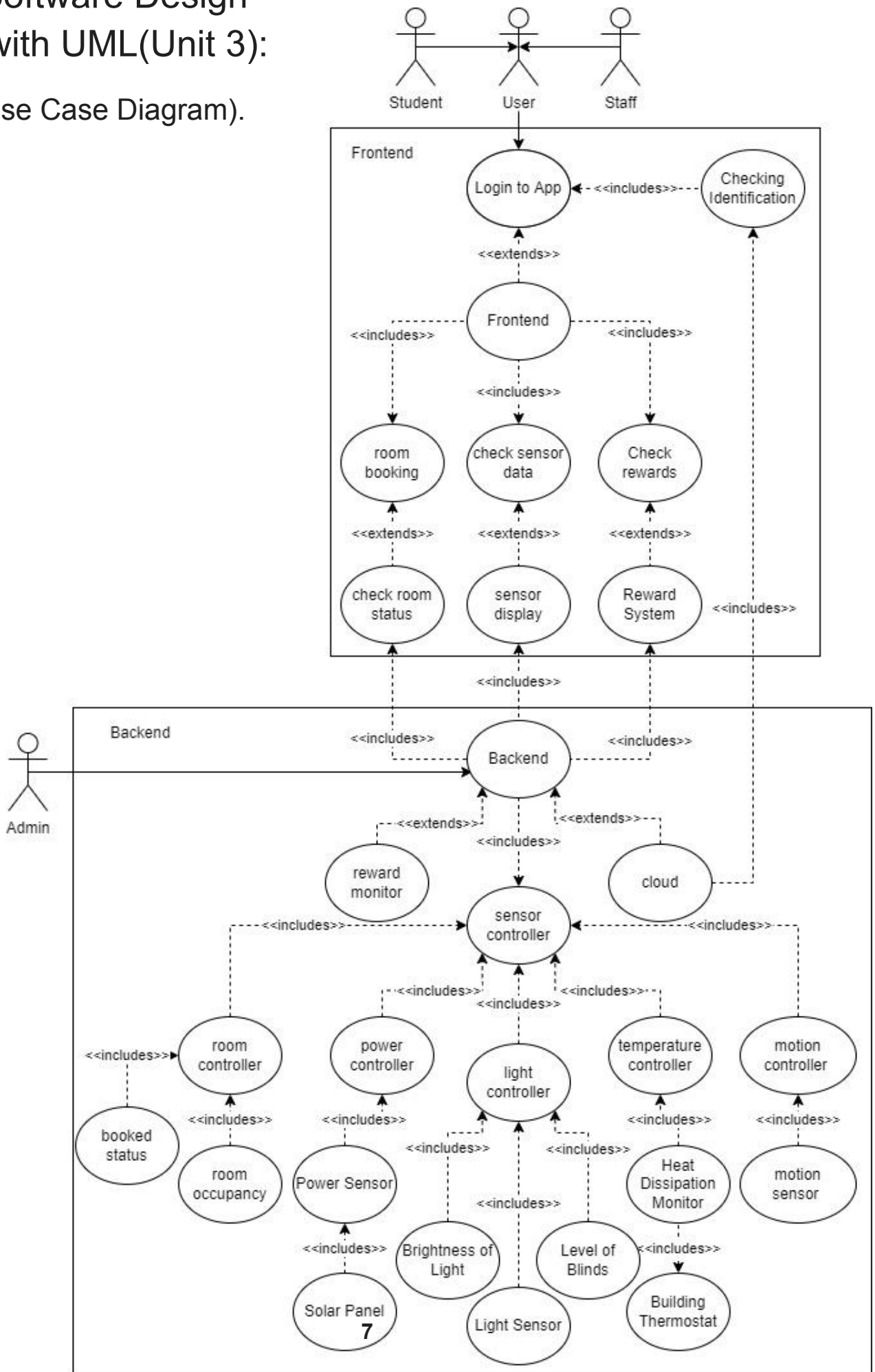
<u>ID</u>	<u>Functional requirement</u>	<u>Priority</u>
1	Backend server <b>must</b> monitor sensor inputs, collate data from cloud storage and modify values requested by the user	M
2	Cloud storage <b>must</b> store all sensor readings and must function as a cold backup for data retrieval in case connection is lost.	M
3	Backend interface <b>must</b> process inputs from the user and handle requests to the server	M
4	Frontend applications <b>must</b> present the user with a GUI, and deal with user inputs and backend data.	M
5	Cloud <b>must</b> monitor user credentials and check them against University of Birmingham user databases.	M
6	System <b>should</b> link temperature sensors to thermostats.	S
7	All buildings' windows and doors <b>should</b> be automated using bluetooth and the data from the sensors.	S/C
8	Light sensors <b>must/should</b> be used to lower blinds if a room gets too bright in the day time.	M/S
9	Light sensors <b>must</b> dim lights if a room is too bright at night time and turn on and brighten lights at night time.	M
10	System <b>could</b> monitor heat dissipation for every room on campus, so that the building with the highest heat dissipation can be insulated to reduce the cost of heating them.	C
11	System <b>should</b> implement systems that measure the difference between a building's power production from things like solar panels and its consumption.	S
12	Motion sensors/bluetooth <b>should</b> be used to monitor if the room is in use .	S
13	Motion sensors <b>should</b> be able to tell if a room is in use or not.	S
14	System <b>would</b> use the motion sensors to get a heat map of traffic	W

15	Reward system <b>must</b> track lowest usage and track by day, month, year.	M
16	System <b>must</b> be concurrent to allow multiple users at a time.	M

#### Non-Functional requirements

<u>ID</u>	<u>Non-Functional requirement</u>	<u>Priority</u>
1	<b>Availability:</b> This app <b>must</b> be tested on the latest version of iOS and Android systems for mobile users.	M
2	<b>Reliability:</b> Front-end and back-end <b>must</b> be kept separate with failover protocols in place and to receive data from the cloud and not the backend in case of downtime.	M
3	<b>Usability:</b> The program <b>must/should</b> be both available on the web and mobile phone app.	M/S
4	<b>Privacy:</b> Users <b>must</b> provide their user identification and transfer over to the user management system.	M
5	<b>Security:</b> Manual and system overrides <b>must</b> be implemented to stop malicious attacks of the automated windows and thermostats	M
6	<b>Standards:</b> App <b>would</b> have regular updates.	W
7	<b>Efficiency:</b> The system <b>must</b> prioritise serving as many users as possible over performing user requests as quickly as possible, given the systems non-critical nature.	M
8	<b>Performance:</b> When a user logs in, the system <b>must</b> respond within 1 sec.	M
9	<b>Error-tolerance:</b> The app <b>would</b> have 1 error in 1,000 lines of code.	W
10	<b>Error-tolerance:</b> <b>Could</b> keep the system enclosed to reduce errors and make sure user input is strongly enforced and error checked accordingly.	C
11	<b>Safety:</b> The system <b>should</b> handle any user data directly but only access through the university system.	S
12	<b>Safety:</b> System <b>should</b> implement user Identification and differentiation, only giving users access to data which they have permission to access.	S

## B. Software Design with UML(Unit 3): B1(Use Case Diagram).



## B2.

### **Name**

Manually changing the lighting in a room.

### **1 Brief Description**

A user accessing and logging into the system to change the current lighting in a room, brightening or dimming the lights in a room and raising or lowering the blinds.

### **2 Actors**

2.1 User

### **3 Preconditions**

The system must be connected to a database system which stores all current user data and is capable of checking credentials against its database.

1. The user already has a University account
2. All sensors in the room are powered up and connected the cloud storage network
3. The user is connected to University WiFi
4. The user has the latest version of the app installed
5. The current state of the blinds have not violated the light level constraints.

### **4 Flow of Events**

The system provides an interface to the user so that they can log in to the system. If the details are correct then the user is presented with another interface where they can select which building and which room they are in. Then the user selects that they want to change the room's lights as the system that they want to alter. Then the user alters or reads the values that they are interested in.

The user goes back to the main screen and then logs out of the system. The app returns to the login page.

### **5 Alternative Flows**

If the user's login details are not correct then the system produces an error and returns the user to the initial login page.

If the user enters a value that violates light level constraints, the backend server interface returns a 'Exceeds maximum' or 'Below minimum' error message to the user, prompting the user to retry.

### **6 Key Scenarios**

The backend server is not-responding.

Special Requirements:

- The blinds can not be lowered beyond the size of the window and they can not be retracted further than they have been extended.



- The lights can not be lowered below 0% of their brightness and can not be brightened to above 100% of their output.

Post-conditions:

- The system must terminate any connection that it has with the database upon the user logging out, but only after sending final updates to the database of any changes made by the user.

## **Name**

Monitoring building power

## **1 Brief Description**

A user logs into the system to monitor a building's power production and consumption.

## **2 Actors**

2.1 User

## **3 Preconditions**

The system must be connected to a database system which stores all current user data and is capable of checking credentials against its database.

1. The user already has a University account
2. There is a cold (offline) backup of data in case any of the power sensors are not currently contributing to real-time information.
3. The user is connected to University WiFi
4. The user has the latest version of the app installed
5. The current power usage of the building has not violated energy usage constraints.

## **4 Flow of events**

The user enters their credentials into the system,

1. The app begins with the login page and requests the user to enter his/her login details  
The users credentials are passed to the database to be checked,

- a. If the login validation check was unsuccessful then the system receives a 'failed check' response and gives an 'incorrect login details' message to the user, remaining on the login page until the user makes a successful attempt.
- b. If the login validation check was successful, the system receives a 'successful check' response and the user is transferred to the main screen of the app.

The user selects the building which they are interested in monitoring from a list of buildings on campus.

User selects 'Building analytics', from which the app presents the user with a list of options.

The user selects that they are interested in monitoring energy usage of the building,  
The user is presented with the data of the building's power production and consumption.

The user goes back to the main screen and then logs out of the system. The app returns to the login page

## **5 Alternative Flows**

### **5.1 Invalid User**

The database returns that the credentials are incorrect,  
The user is shown an Error message,  
The user is taken back to the login page.

### **5.2 Backend Server Is Down**

The server returns an error,  
The application sends a data request to the cloud,  
The cloud provides previously stored data instead of current data,

## **6 Key Scenarios**

The cloud is inaccessible.

## **7 Post Condition**

### **7.1 Successful Completion**

The user can see current power related data for the building.

### **7.2 Failure to Connect to Backend**

The user is presented with old data from the cloud.

### **7.3 Cloud is unavailable**

The user is presented with errors.

The app redirects the user to the specified building page, then the User selects 'Building analytics'.

The user selects 'Power usage' from a list of sensors.

- a. If at least one of the power sensors is undergoing maintenance or not connected to the cloud storage network for any reason, the information is accessed via an offline backup.

B3.

## **Story: Controlling and monitoring lights and power consumption in buildings**

A group of students at the University of Birmingham want to hold weekly meetings in a specific room. Their weekly meetings are focused on a university project. The project is focused on making the university environment friendly. They aim to focus on the university being environment friendly and the development of student well-being. They gathered electricity consumption data using IoT sensors. This data is displayed on the university application which provides data regarding campus information. David, one of the group members, noticed the inconvenience of controlling the electricity consumption and lights of the meeting rooms. The group of students decided to reach out to the university's IT team to implement a reliable and convenient way of managing power

consumption and light levels. University students should have access to adjust and light levels based on personal preference when using the room on the app. While the room is not in use, lights should be optimised to support lowering power consumption resulting in a more environment friendly campus . The IT team suggests making a function in the university's app that gives access to students who booked a specific room to control its light level beforehand and whilst using the room. When a user books a specific room they are asked to choose a preferred all of the room conditions before confirming the booking, so the room is set to their preference when they use the room.

### **Scenario: Site team doing monthly maintenance on building**

#### **Initial assumption:**

The University of Birmingham site team wants to improve the insulation of the buildings on campus but don't know which building or where to improve it.

#### **Normal:**

Reviews the data from all the buildings' heating usage over the past month from a portion on the app/website only available to people with certain security access. Can sort all buildings by heating usage to pick on a specific building to work on.

The site team can then arrive at the building and use tablets to look at different rooms temperature logs as they do a walk around of the site. This approach of human expertise and the data available to them, will improve their decision making and overall make the University a more eco-friendly space.

#### **What can go wrong:**

Weather is an unstable and ever changing environment so the data given might not give the full story or accurately display what is going on. If a sensor is faulty the site team will be wasting more time fixing the system to reap its benefits, instead of actually improving the campus heating efficiency.

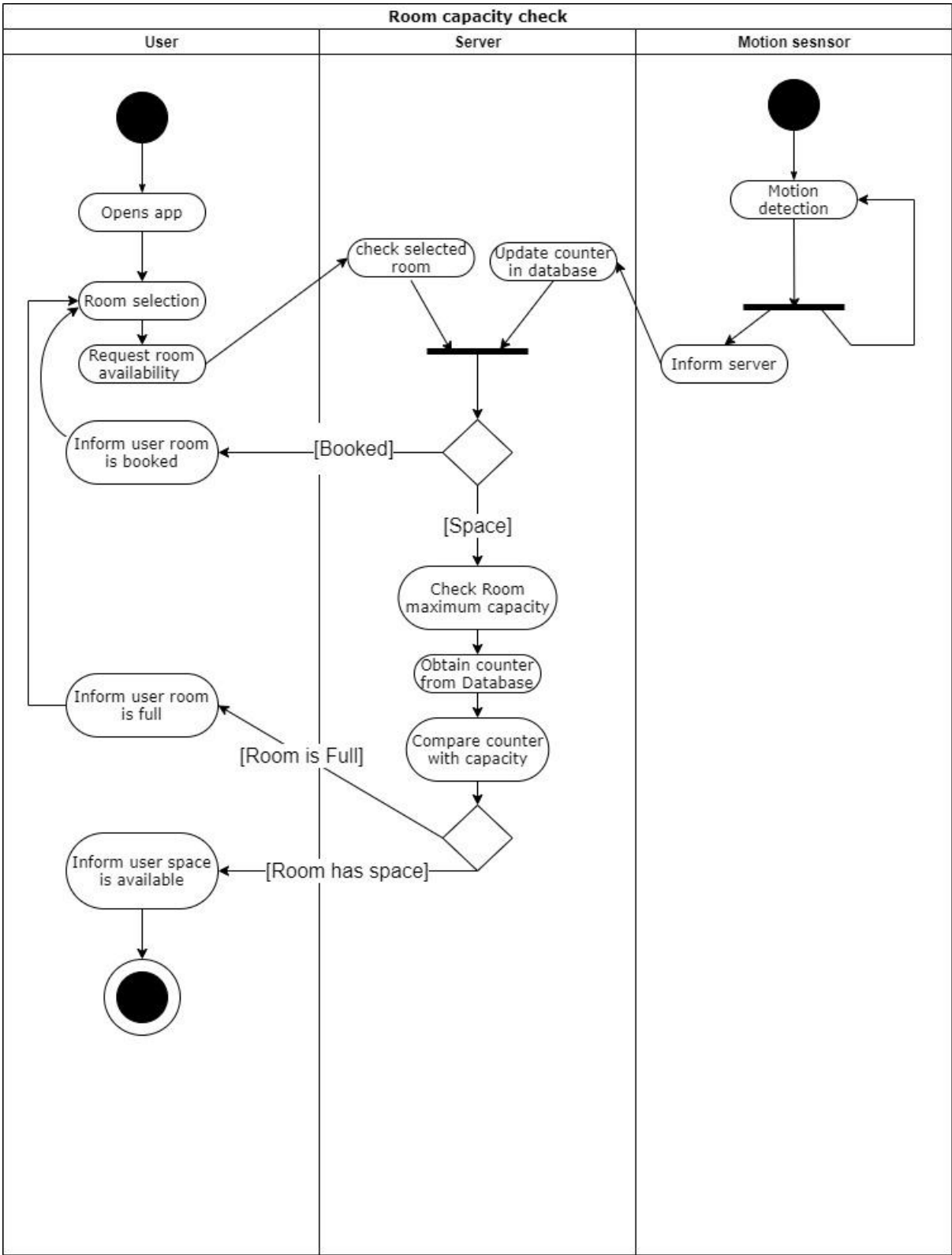
#### **Other activities:**

People can still change the temperature of the room while this is going on.

#### **System state on completion:**

Make a record of building maintenance happening on that day for future site employees and to improve efficiency.

B4 (Activity Diagram).



## B5.

- i. Perform noun-verb analysis over your specification: Identify the potential classes and potential operations for these classes using what discussed in Lectures.
- ii. Derive your CRC cards for the identified classes from the last step.
- iii. Produce a First-Cut Class diagram to combine the consolidated results of using the above two techniques – i.e., noun-verb analysis and responsibility-driven analysis.
- iv. Detail your Class Diagram. This should provide details on attributes, operations, relationships, visibility, multiplicity, etc) (refer to the class notes).

(i)

### **--Frontend--**

The purpose of this app is for students or lecturers to have accessibility to some features like checking the rooms that are currently in use or checking the heat map or taking part in a reward scheme by selecting the building that the user believes will be the most energy efficient.

#### **Login**

The system displays a login page in which the user types their ID and password. The system, depending on their log in details, validates if the user is a student or a lecturer. If the user is neither of them then the system denies the user from accessing the features mentioned above. After a successful login the user will select which type of feature they want to use from a menu. The menu will include the Rooms in use map feature, the Heat map and the Reward feature.

#### **Rooms in use map**

The system provides information on which rooms are currently in use and how much free space they have using data from motion sensors in each building and room. Additionally, the user can check and track the uses of the rooms by selecting a date of the format: (Day/Month/Year). This type of information will be displayed on a map of the university's campus. Every room from each building will display a fraction with the numerator being equal to the number of people occupying the room and with the denominator being equal to the number of the room's maximum capacity. Every room from each building will be coloured :red , orange, yellow, light green and green , light blue , blue depending on the real-time capacity.

The system displays a map of the campus which is colour coded to display the heat in each building using data from motion and temperature sensors. The rooms of the campus buildings will be coloured :red , orange, yellow, light green, green depending

on the real-time capacity with red meaning that the building has super high temperature and green meaning that it has super low temperature. This usage can also be tracked by date.

### **Reward System**

Monthly the user will select a specific building that they believe will be the most energy efficient. The system, based on energy consumption data and statistics that it will receive from heat sensors, will determine every month which building was the most efficient. If the users choose correctly, they will receive a voucher to their email for a free cup of coffee.

**--Backend--**

### **Light Sensors**

The system will determine light usage using the light sensors depending on the time of day. If the room is too bright the system will lower the blinds. At night time, the system should turn on and brighten the lights.

### **Thermostat**

The system will use temperature sensors and motion sensors to adjust the building temperature depending on the room temperature and how many people occupy the building. The temperature is regulated so there is a minimum temperature when the room is not in use to conserve energy.

### **Heat Dissipation Monitor**

The system monitors the heat dissipation from every room on campus, the buildings and rooms with the highest heat dissipation can be insulated to reduce the cost of heating them. Heat dissipation data is uploaded to the database and used for the reward system.

### **Power Sensors**

The system monitors the energy consumption of the building in relation to the solar panels which generate the buildings energy. This data is uploaded on the database and used for the reward system.

### **Cloud**

Cloud has to store the data from the backend, and data of user information (user ID, Password) to check user identification.

(ii)

Frontend	
Responsibility	Collaborator
<ul style="list-style-type: none"><li>• Let users log in to the app.</li><li>• Display features to users</li><li>• Access to reward system</li></ul>	<ul style="list-style-type: none"><li>• Login App</li><li>• Monitoring Rooms</li><li>• Heat map</li><li>• Reward system</li><li>• Heat Dissipation monitor</li></ul>

Login App	
Responsibility	Collaborator
<ul style="list-style-type: none"><li>• Check users identification</li></ul>	<ul style="list-style-type: none"><li>• Frontend</li><li>• Cloud</li><li>• Backend</li></ul>

Monitoring Room	
Responsibility	Collaborator
<ul style="list-style-type: none"><li>• Check room is free</li></ul>	<ul style="list-style-type: none"><li>• Frontend</li><li>• Backend</li></ul>

Heap map	
Responsibility	Collaborator
<ul style="list-style-type: none"><li>• Check temperature of buildings</li></ul>	<ul style="list-style-type: none"><li>• Frontend</li><li>• Backend</li></ul>

Backend
---------

Responsibility	Collaborator
<ul style="list-style-type: none"> <li>Send data of Heat Dissipation monitor, Light Sensors, Building Thermostat, Power Sensors and Cloud to Frontend's Functions make user can use them</li> </ul>	<ul style="list-style-type: none"> <li>Monitoring Rooms</li> <li>Reward System</li> <li>Heatmap</li> <li>Heat Dissipation monitor</li> <li>Light Sensors</li> <li>Building Thermostat</li> <li>Power Sensors</li> <li>Cloud</li> </ul>

Reward system	
Responsibility	Collaborator
<ul style="list-style-type: none"> <li>Check amount of power usage</li> </ul>	<ul style="list-style-type: none"> <li>Backend</li> <li>Frontend</li> </ul>

Light Sensors	
Responsibility	Collaborator
<ul style="list-style-type: none"> <li>determine light usage using the light sensors</li> </ul>	<ul style="list-style-type: none"> <li>Building Thermostat</li> <li>Backend</li> <li>Heat Dissipation monitor</li> </ul>

Building Thermostat	
Responsibility	Collaborator
<ul style="list-style-type: none"> <li>adjust the building temperature depending on the room temperature</li> </ul>	<ul style="list-style-type: none"> <li>Backend</li> </ul>

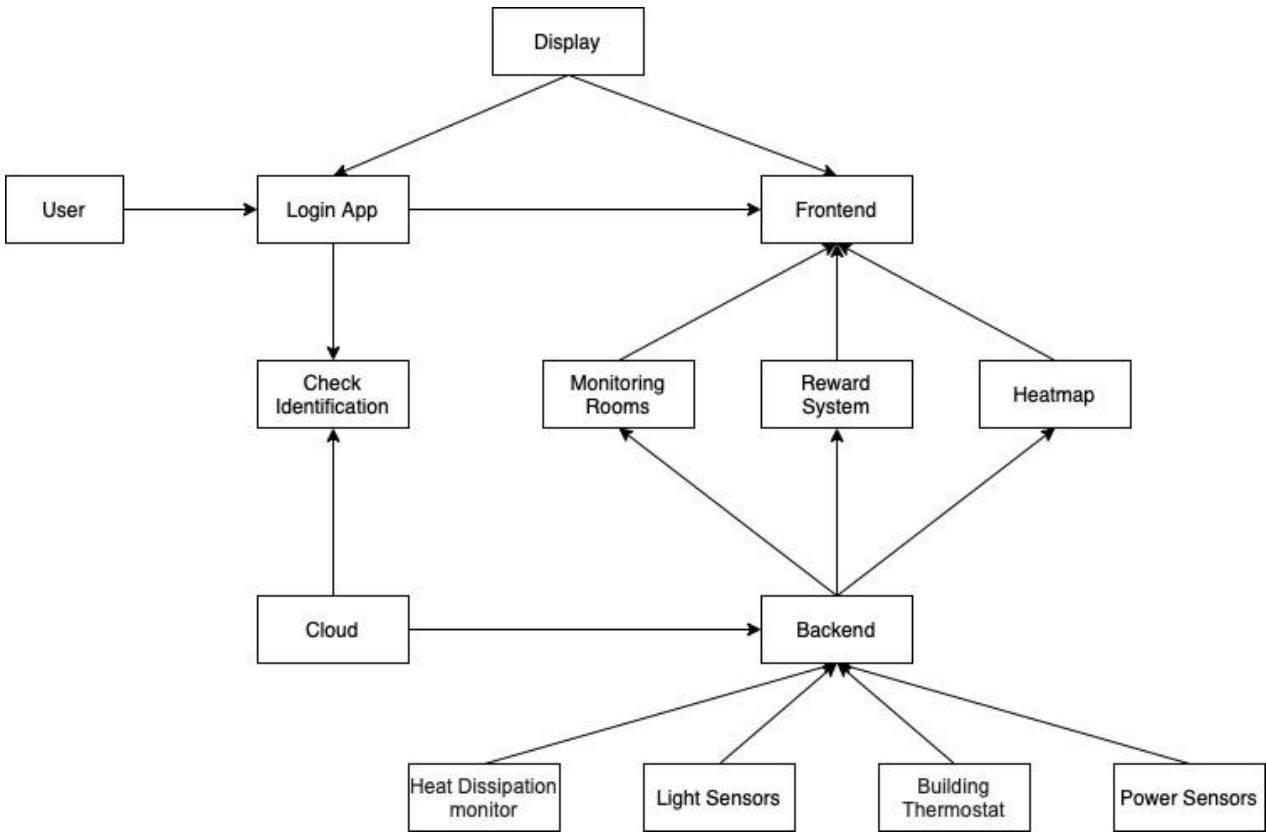
Heat Dissipation monitor	
Responsibility	Collaborator
<ul style="list-style-type: none"> <li>the highest heat dissipation can be insulated to reduce</li> </ul>	<ul style="list-style-type: none"> <li>Backend</li> </ul>



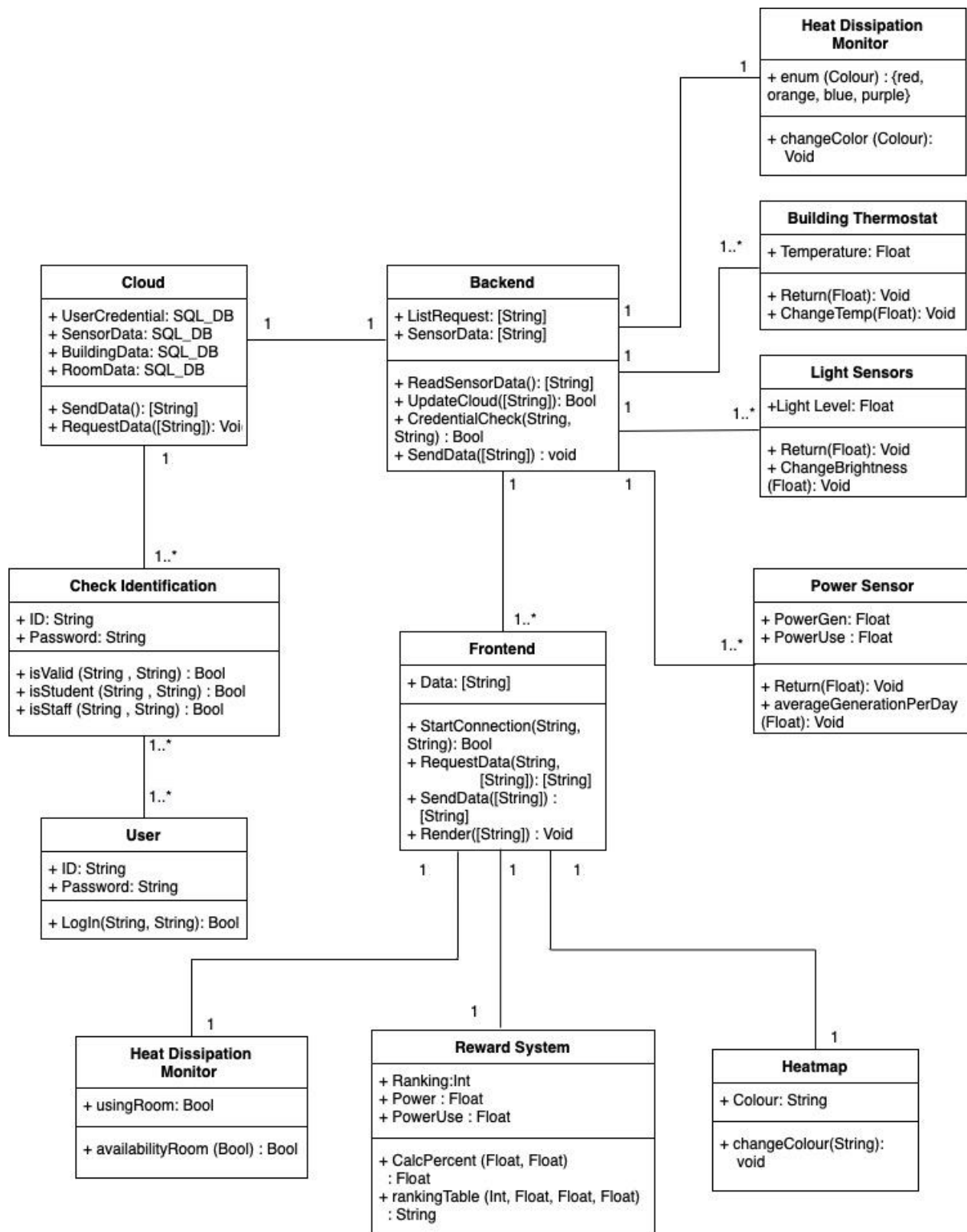
Power sensor	
Responsibility	Collaborator
<ul style="list-style-type: none"> <li>monitors the energy consumption of the building</li> </ul>	<ul style="list-style-type: none"> <li>Backend</li> </ul>

Cloud	
Responsibility	Collaborator
<ul style="list-style-type: none"> <li>Store data from backend</li> <li>User identification</li> </ul>	<ul style="list-style-type: none"> <li>Login App</li> <li>Backend</li> </ul>

(iii) (First-Cut Class Diagram)

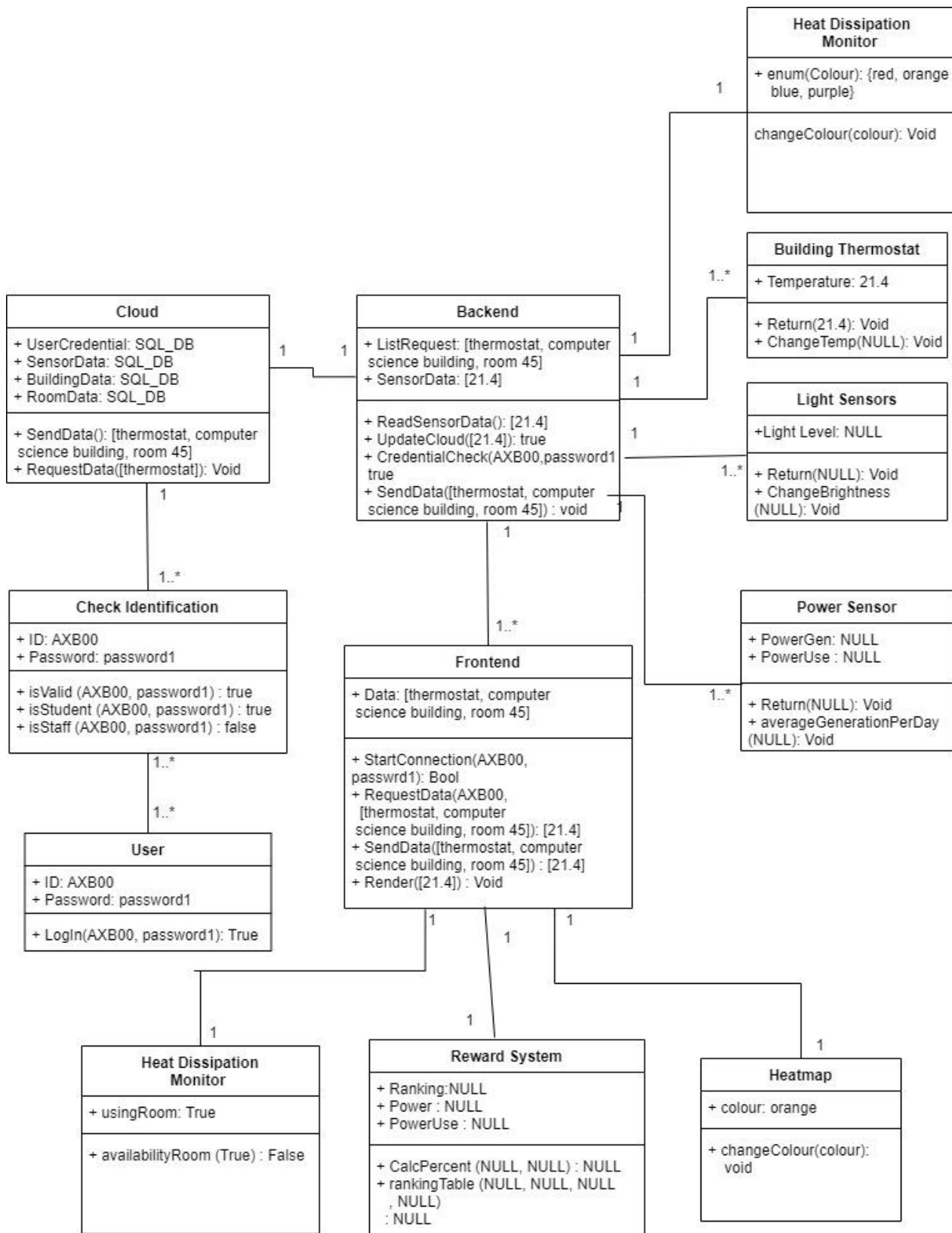


(iv) (Class Diagram)



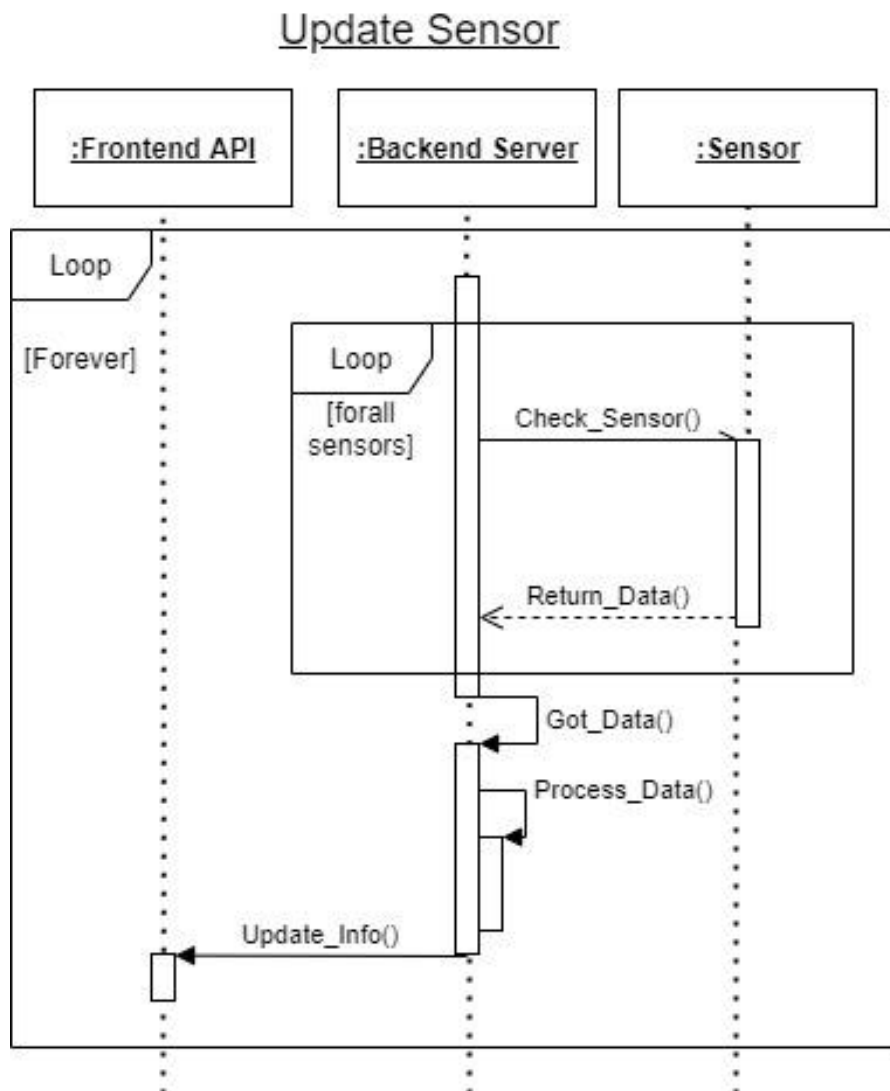
## B6 (Object Diagram).

Scenario: User logs into application and then requests to heat up a study space that they are in.

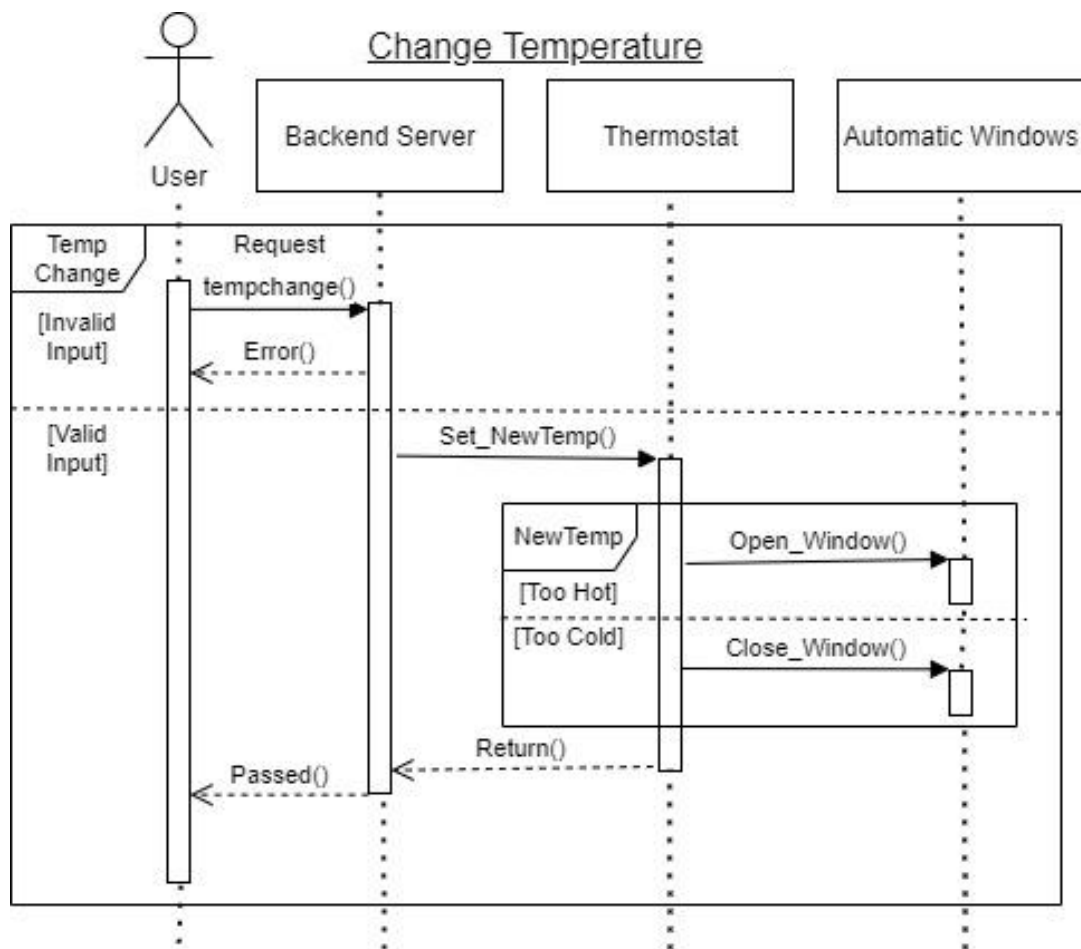


## B7 (Sequence Diagram).

### The process between a Sensor and the rest of the system



## A User changing the temperature and the system reacting

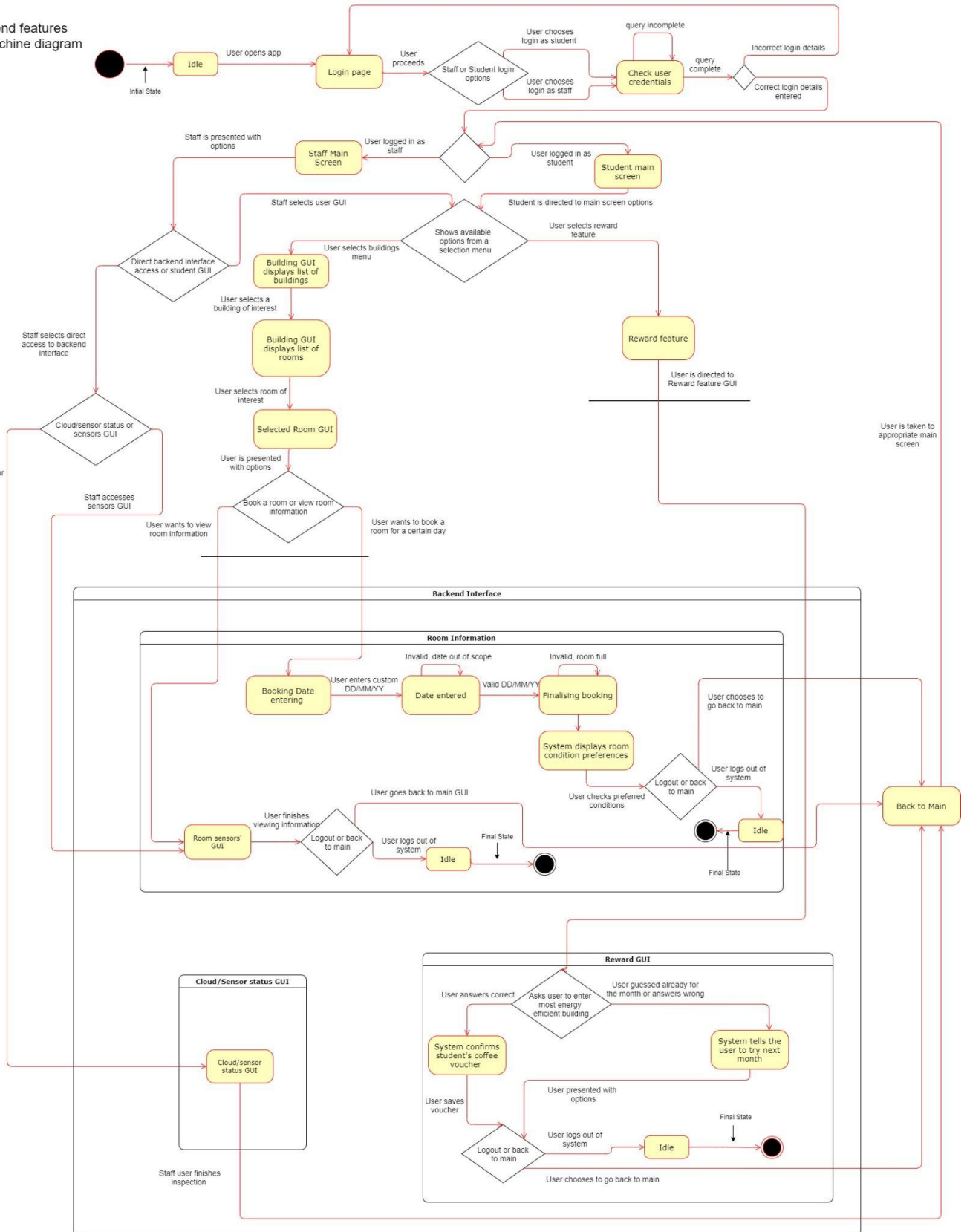


## B8 (State Machine Diagram).

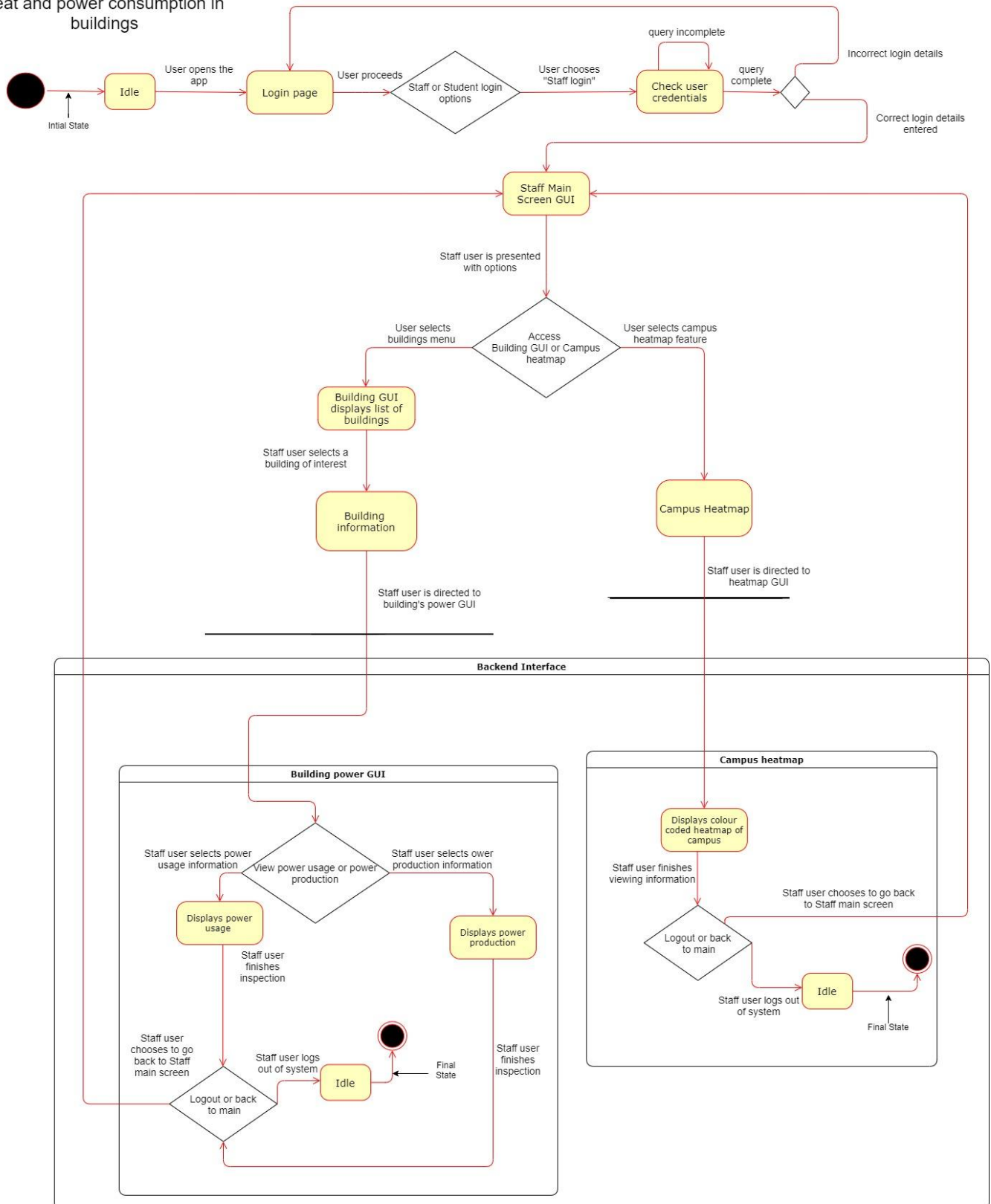
For the above questions that require modelling non-trivial scenarios, you may choose from those that were documented as part of the Use Case modelling (B1) and generated scenarios (B3) as this can simplify your work. Please ensure consistency with your description as much as possible.

Assumptions made: The backend interface is provided to the data collation interface in the backend server in order to access its sensors/status GUI, and all other backend interface GUIs.

## Frontend features state machine diagram



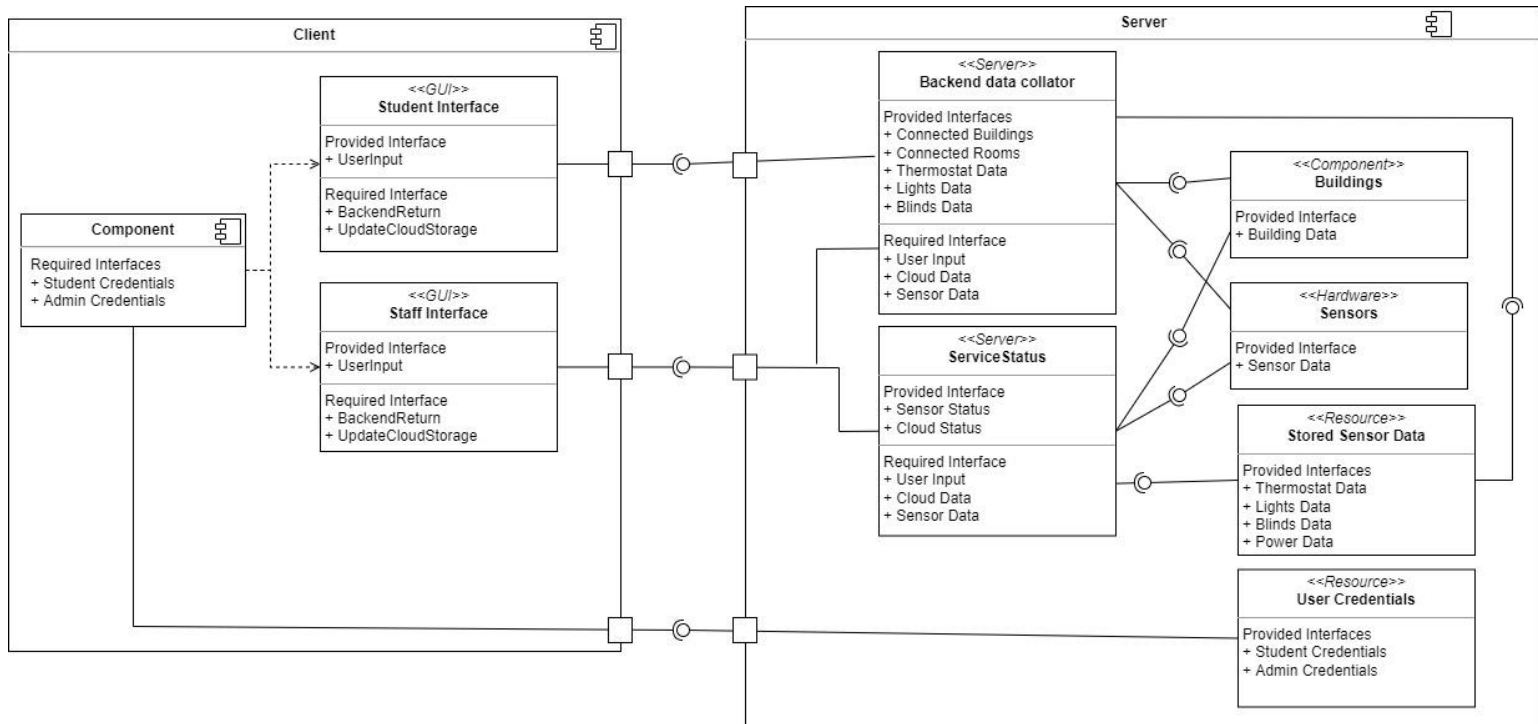
# Staff controls and monitors heat and power consumption in buildings



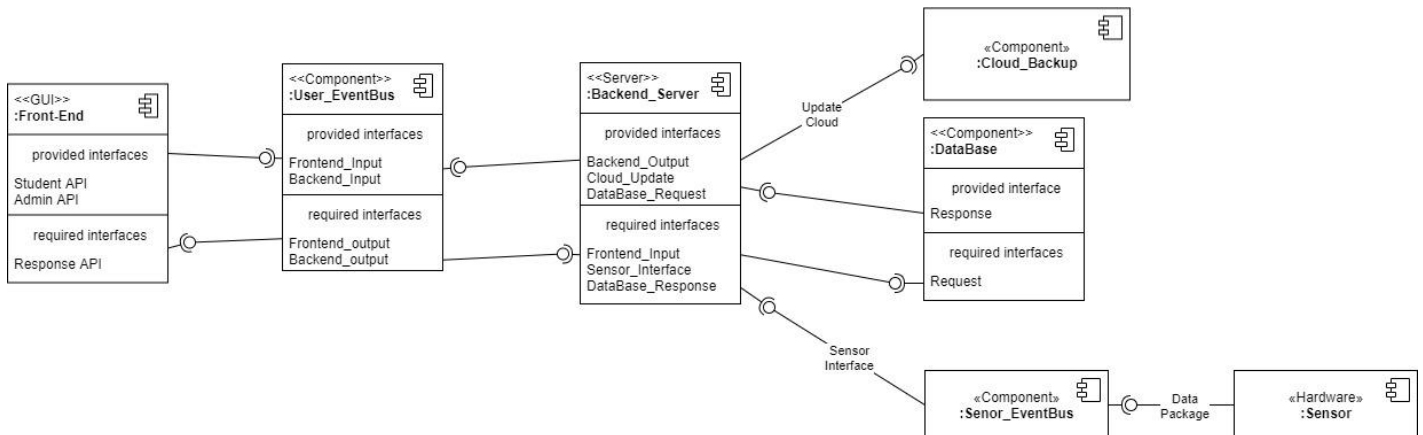
C.

## C1 (Component Diagram).

### Server-Client



### Event-based

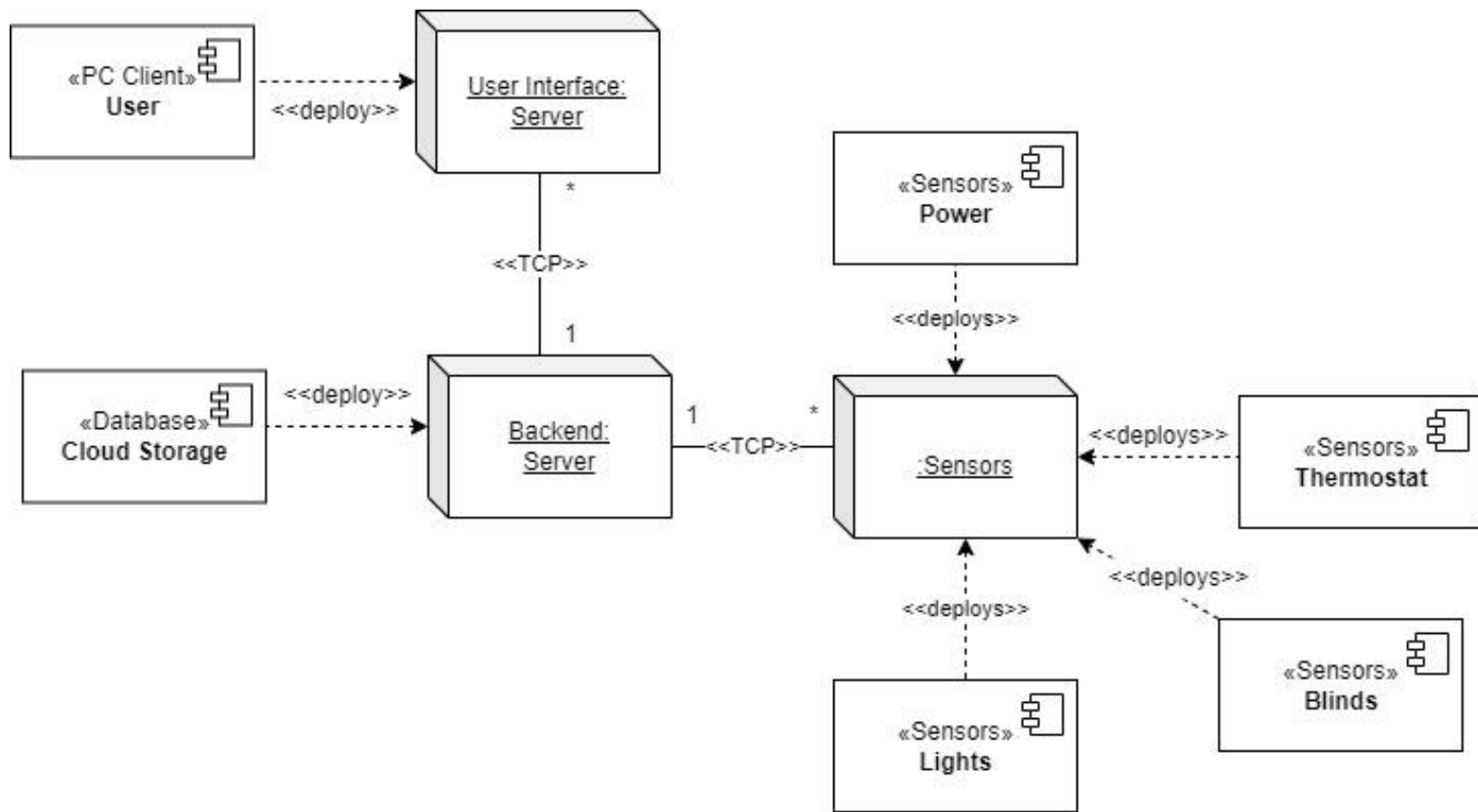


Event-Based architecture relies on a shared memory queue but with only specific classes are allowed insertion or(Exclusive) deletion. EventBus classes should be used where high traffic of data can be organised to reduce workload and synchronisation.

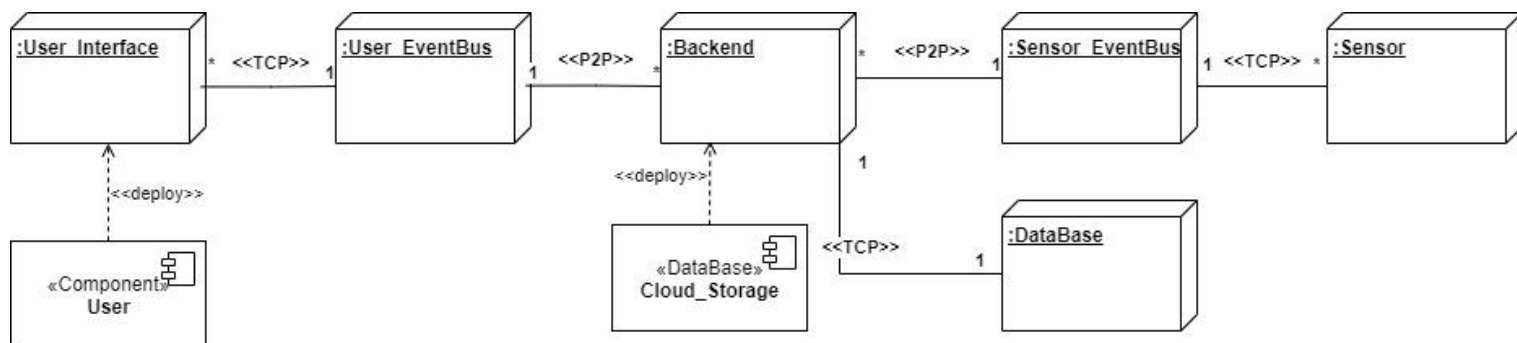


## C2 (Deployment Diagram).

### Server-Client



### Event-Based



EventBus components are connected with P2P to the Backend because the hardware will be set up in LAN. For everything else it needs to be a secure network and point of contact to the WAN so TCP will be used.

### C3.

Software architecture is an abstraction of a system that includes the software's components and connectors in regards to how they interact with each other and how they act in a certain environment. Components assume Externally-visible properties of other components including provided services, fault handling and performance characteristics. A Client-Server based model consists of two categories, a server and multiple clients. The server listens to client requests and provides relevant services to those clients. The server uses TCP/IP protocols to send and receive data to clients. Event-based architecture detects "events" or important moments and acts on them in real time. In event-based systems components don't have to wait for each other to move onto their next task. Event-based architectures have three main components which are event producers, event routers and event consumers. Producer services and consumer services are decoupled, which allows them to be scaled, updated and deployed independently.

The key difference between both architectures is that Client-Server architecture calls are synchronous while on the other hand, Event-based architecture calls are asynchronous. Obtaining data in a Client-Server model is slower than Event-based. Clients have to wait for a response after sending requests to the server in the Client-Server model. However, communications are Event-based that don't require components to wait for each other. When an event occurs, other components consume those events so that they can perform any of their tasks needed as a result of the event.

Event-based models are more complicated to implement compared to Client-Server models. Despite their complexity, Event-based models are often used in projects that demand scalability and real time response. The client-server system stores all the files in a central location where if any part of the network fails a lot of disruption can occur. Data interchange is more efficient in Event-based systems than Client-Server systems. Finally, the cost and maintenance of Client-server based systems is higher than Event-based systems.

In conclusion, Our system is designed to gather large amounts of data and be accessible in real time by users. Event-based model is preferable to use in implementing our desired project. The scalability and real time response are key requirements to capture data easily and more flexible from IoT devices in our system.

D.

#### Functional requirement

1. Backend system for monitoring sensor inputs, user data requests, processing inputs and storing them in cloud storage.
2. Frontend application for presenting the user with a GUI, receiving user inputs, sending and receiving user requests to end from the backend, looking up data from cloud storage, monitoring user credentials and checking them against Birmingham user databases.
3. Reward system for lowest usage and Track by day, month, year
4. Use the motion sensors to get a heat map of traffic
5. Light sensors can simultaneously be used to lower blinds if a room gets too bright in the day time, dim lights if a room is too bright at night time and turn on and brighten lights at night time
6. Link temperature sensor to thermostat
7. System that measure the difference between a building's power production from things like solar panels and it's consumption

#### Non-Functional requirement

1. Front-end and back-end are kept separately with failure-protocols in place
2. Users must provide their user identification and transferred over to the user management system
3. Manual and system overrides to stop malicious attacks of the automated windows and thermostats

#### Introduction

Based on the data of the motion sensors, check if the number of people that are currently using the room is lower than or equal to the room's capacity. Also upload and store the data from booking information and motion sensors in the cloud to ensure that the amount of people within the room is less than or equal to the number of people that had requested booking the room. The system should adjust the information according to what the user proposes. The energy usage and production of each building should also be stored correctly within the cloud. The concurrency of the system will be tested through

making multiple users login simultaneously and use the same function. The light sensors must get the correct information of the sun's brightness and the system should adjust the room light according to that data. The temperature sensors and thermostat will be tested to be well linked by decreasing or increasing the temperature of the room and checking their response. The systems calculation of difference between power production and consumption of each building will be checked against actual readings of the buildings power production and consumption and making the calculation to ensure that it is correct.

Separately do error-checks on both the frontend and backend and develop a system that will locate the error. Also, confirm that the ID and password of the user are accurately sent to the checking identification system, and ensure that the identification system has access to the cloud. The system has to also be able to react to malicious software at automated windows and thermostats.

- Response time for login to the app with valid ID and password.
- Response time to give information from functions
- Response time to adjust temperature
- Response time to modify the lights of the room
- User response time when 500 users are logged in

## **Test Items**

This app has to be tested on the latest version of iOS and Android systems for mobile users. Furthermore, it has to provide a website for desktop users, so it has to be tested on the latest version of Internet explorer, Chrome, FireFox, Safari. Furthermore, using testing software Testim to test the usability of the program.

## **Features To Be Tested**

- As a student, logging into the app as a student.
- As a staff, logging into the app as a staff
- As a user (student or staff), checking room in use
- As a user, checking heatmap
- As a user, checking rewarding system
- As a admin, dim or brighten lights
- As a admin, lower blind
- As a admin, checking the identification of users
- As a user and admin, checking energy production of each building from solar panel
- As a admin, testing concurrency of the app

- As a user and admin, checking energy consumption of each building
- As a thermostat, adjusting the temperature of the room
- As a light sensor, checking the brightness of the room and change to enough brightness
- As a motion sensor, testing heat map of traffic is well tracked

## **Features Not To Be Tested**

We won't check the calibration of every thermostat and take the word of the manufacturer that it is correct.

For the reward system, we could not check energy consumption and production of every building, so it could not be tested.

## **Approach**

Programmers will test each Non-Functional requirement using black-box technique creating an evaluation report based on viewed outputs and energy efficiency. We will do so with a stress test approach trying to overload the system with multiple users to match what can be expected in a worst case scenario. At the end of this test we will record the recovery time to boot up and check if data is lost in a recovery test.

For the functional requirements we want a more in-depth testing of the code and that is why we've chosen white-box testing. A monkey testing environment along with unit testing to automatically check for fail cases along with the added benefit for monitoring recovery systems in case of a crash. We also value the collaboration of the different parts of the system so integration testing will be a part of our approach. For example, they could test the ranking system by using integration testing to test cloud and backend cooperated well.

If some tests have errors, the programmers will send them to the developing team, and they will fix and send them again to QA. While the test is passed, do this process infinitely.

## Pass Fail criteria

Project Name: Happy Birmingham

Group Number: Group 51

Test Case ID	Test description	Test steps	Test Data	Expected result
TD-1.0	Verify the login with valid student's userID and password	-Open the app/website -Enter Userid,Password -Click login	UserID: student_name Password: ABC	User should be able to login as student
TD-1.1	Verify the login with valid staff's userID and password	-Open the app/website -Enter Userid,Password -Click login	UserID: staff_name Password: ABC	User should be able to login as staff
TD-1.2	Verify the login with invalid userID and valid password	-Open the app/website -Enter Userid,Password -Click login	UserID: wrong_name Password: abc	User should not Login into an application
TD-2	Check rooms in use	-Go to Room in use map section		A map showing which rooms are in use
TD-3	Check heatmap	-Let some people go into the room -Watch heatmap is well operating	Temperature of people	Show the temperature of people in the room
TD-4	Check rewarding system	-Go to reward system section		A list of campus building sorted by energy usage
TD-5.1	Control the blind to be upper as an admin	-Make blind upper	Operation of blind	Changing level of blind
TD-5.2	Control the blind to be lower as an admin	-Make blind lower	Operation of blind	Changing level of blind
TD-6	Check identifications of users	-Click or tap profile of user -Check identification	Data of users	Show correct identification
TD-7	Check energy production of each building from solar panel	-Click or tap GUI of building -Check production is well shown	Energy production data	Show correct amount of energy production
TD-8	Check energy consumption of each building	-Go to Buildings Energy Consumption section -Select a building	The building	Show correct amount of energy consumption
TD-9	Check motion sensor well track traffic of heat map	Compare data of sensor and real world	Data of traffic	Well-tracked traffic as real world

## Exit Criteria

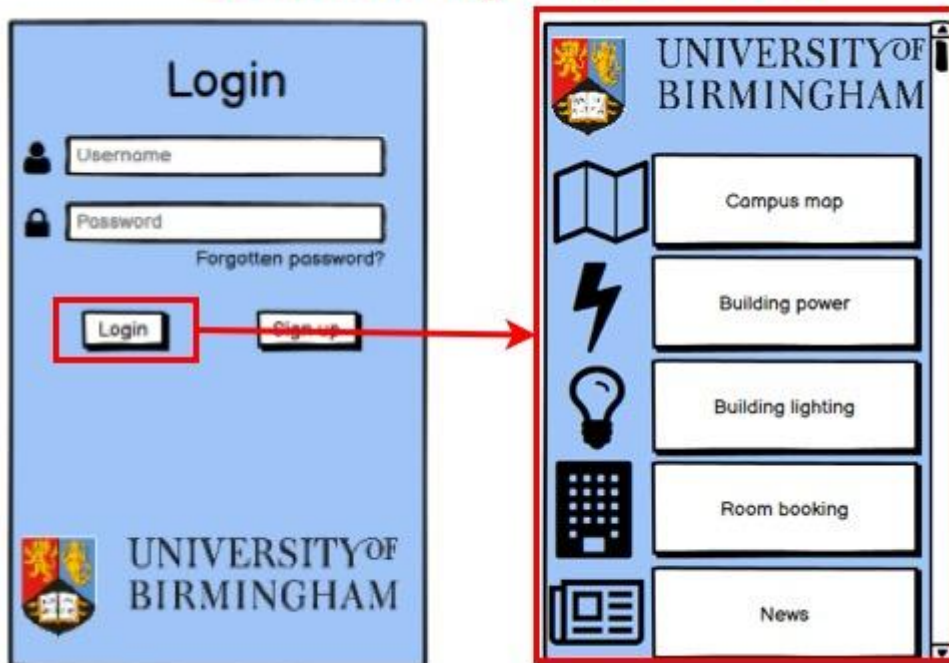
92% of all test cases should pass with no failed critical cases.

## E. Usability and Prototyping (Unit 6).

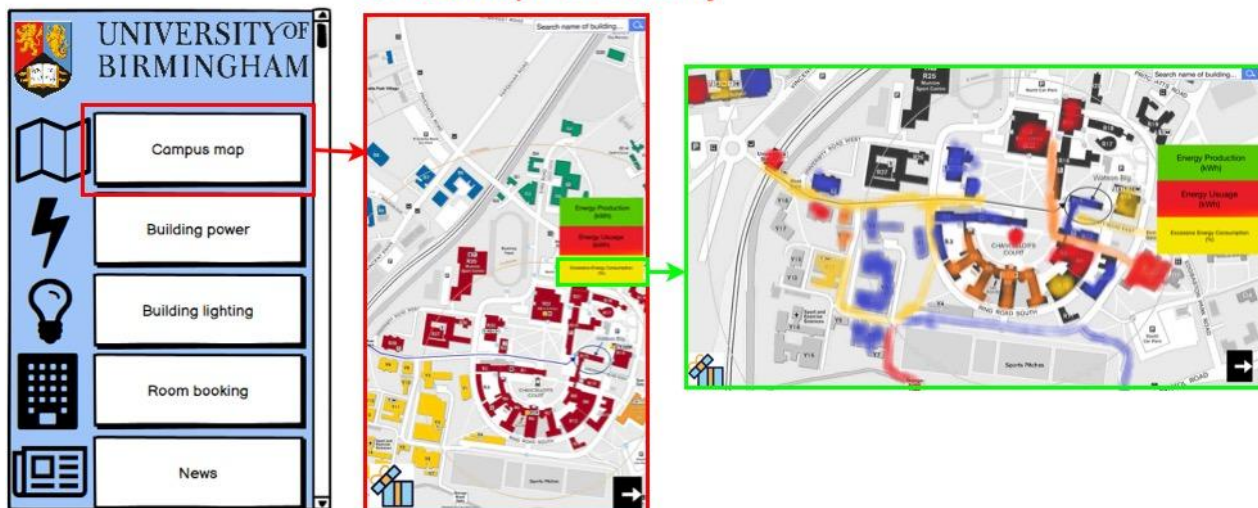
E1.

System Interaction ProtoType


### Succesful log in open menu




### Canvas Map Functionality





## Select Building





**UNIVERSITY OF BIRMINGHAM**

 Campus map


 Building power

 Building lighting


 Room booking

 News

Name	Distance	Occupancy	Button
Occupational Health, 0m, 50/200			Button
Public Health, 0m, 50/200			Button
Murray Learning Centre, 0m, 50/200			Button
Computer Science, 0m, 50/200			Button
School of Engineering, 0m, 50/200			Button
UKRIIN, 0m, 50/200			Button
Collaborative Teaching Laboratory, 0m, 50/200			Button
Bioscience, 0m, 50/200			Button
Haworth Building, 0m, 50/200			Button



**UNIVERSITY OF BIRMINGHAM**




**CS Building**

Name: Y9  
Capacity: 700  
Energy Ranking: 1st  
Net-Energy: +80kW  
Energy generation: 120kW

**Rooms**

RoomID	Room	Capacity	Available
UG01	2/10	TRUE	
UG02	5/5	FALSE	
UG03	8/8	FALSE	
UG04	74/200	TRUE	
UG05	28/90	TRUE	
UG06	3/10	FALSE	
UG07	0/30	FALSE	
AVON	2/400	FALSE	

## Change Temperature



**CS Building**

Name: Y9  
Capacity: 700  
Energy Ranking: 1st  
Net-Energy: +80kW  
Energy generation: 120kW

**Rooms**

RoomID	Room	Capacity	Available
UG01	2/10	TRUE	
UG02	5/5	FALSE	
UG03	8/8	FALSE	
UG04	74/200	TRUE	
UG05	28/90	TRUE	
UG06	3/10	FALSE	
UG07	0/30	FALSE	
AVON	2/400	FALSE	



**Room 217**

Capacity: 6/6  
Available: In-Use  
Room type: Pre-Book  
Booked: You  
Temperature: 22°C


Change Temperature



Energy Usage

Room: UG04  
Staff access required

22°C



**UNIVERSITY OF BIRMINGHAM**



## Reward Vote

UNIVERSITY OF BIRMINGHAM

Campus map

Building power

Building lighting

Room booking

News

Claim Reward!

Ranking	Name	Energy Production	Energy Consumption	(Production-Consumption)/Consumption
1	Computer Science	200kWh	120kWh	+ 66.6%
2	Main Library	400kWh	250kWh	+ 60.0%
3	Engineering	250kWh	160kWh	+ 56.2%
4	University Center	300kWh	200kWh	+ 50.0%
5	Music	150kWh	120kWh	+ 25.0%
6	Old Gym	200kWh	180kWh	+ 11.1%
7	Haworth Building	220kWh	220kWh	0%
8	Chemical Engineering	180kWh	230kWh	- 21.7%

News

E2.

See attached mp4.

## F.

The app has been created within the interest of the public through the inclusion of functions that have been implemented to benefit the staff and students of the university in regards to their needs on campus. These functions have been created to be easily accessible and usable to contribute to the user's experience on campus and also to benefit society through the monitoring of energy consumption.

This app gives information on energy consumption of buildings but does not show individual users' contribution to this energy consumption in order to reduce individual harm, both intentional and unintentional. There is no way to discriminate between specific users on what buildings they decide to use in order to respect their privacy. Also, the ranking system which has been implemented as a rewards system is used to reduce energy usage in the wider context of reducing environmental harm. There may be some danger to manipulate rankings within the reward system, so the app will show specific data of energy consumption and generation for the user to ensure the accuracy of readings.

To ensure a high quality of computing, the programming and developing team of this app must engage with the University of Birmingham to understand the needs of the clients through a process of communication and reflective analysis in order to understand any challenges in creating an app specifically for the University. This should be constantly reviewed at all stages by an independent team.

In the case of any errors when using the app, a notification is sent to the admin in order for the issue to be fixed as soon as possible to keep the system running as required.

This app only takes the student or staff ID from the user to verify their identification from the University, so there is no way to inappropriately use their personal information. No data will be shared or stored to respect confidentiality.

To maintain professional standards, every member involved must meet the focus to judge that every stage is well designed. Members involved with creating this project should work rotationally and remain under the maximum working time. Their physical and psychological well-being should be prioritised to ensure a high quality of working life within a safe environment. The work should be divided fairly between developing groups and programmers should be rewarded or penalised appropriately for their work. Programmers should have prior knowledge relevant to creating this app. The infrastructure of the app should also be constantly checked in order to monitor if there are any uses of the app that could have negative consequences.

Any potential concerns regarding the ethics of the app should be resolved as soon as possible. All members of the developing team should adhere to the criteria of the ACM code and any violations reported.