HO CHI MINH CITY UNIVERSITY OF TECHNOLOGY
FACULTY OF COMPUTER SCIENCE AND ENGINEERING

Software Engineering - CO3001

Urban waste collection aiding system - UWC 2.0

Assignment's Report

Lecturer: Trương Tuấn Anh
Students: Nguyễn Thế Bình - 2052002
Cao Minh Quang - 2052221
Lâm Quang Khải - 2052128
Trần Cao Duy Trường -2052299

HO CHI MINH CITY, September 2022

Contents

1 Member list and Workload

1.1 Task 1

No.	Full name	Student ID	Work contents	Contribution
1	Nguyễn Thế Bình	2052002	Sections 2.1 and 2.2.1 and 2.2.2	25%
2	Cao Minh Quang	2052221	Sections 2.3.2 and 2.3.3	25%
3	Lâm Quang Khải	2052128	Sections 2.2.3 and 2.3.1	25%
4	Trần Cao Duy Trường	2052299	Sections 2.3.4 and 2.3.5	25%

1.2 Task 2

No.	Full name	Student ID	Work contents	Contribution
1	Nguyễn Thế Bình	2052002	Section 3.3.2	25%
2	Cao Minh Quang	2052221	Sections 3.1 and 3.3.3 and 3.3.5	25%
3	Lâm Quang Khải	2052128	Sections 3.1 and 3.3.1 and 3.3.4	25%
4	Trần Cao Duy Trường	2052299	Sections 3.1 and 3.2	25%

1.3 Task 3

No.	Full name	Student ID	Work contents	Contribution
1	Nguyễn Thế Bình	2052002	Section 4.2	25%
2	Cao Minh Quang	2052221	Section 4.1	25%
3	Lâm Quang Khải	2052128	Section 4.1	25%
4	Trần Cao Duy Trường	2052299	Section 4.2	25%

2 Task 1

2.1 Identify the context of this project

Q1. Who are the relevant stakeholders ?

A typical waste collection process involves:

1. Back officers, who operate a central system to create calendar, coordinate front collectors and janitors.
2. Collectors, who drive different types of vehicles.
3. Janitors who manually collect garbage from Major Collecting Points (MCPs).

Q2. What are their current needs?

Calendar and tasks were assigned among teams of janitors and coordinated by back officers. These assignments are often arranged on a weekly basis. Back officers also plan which vehicles to use and their routes. This planning activity happens every month.

Q3. What could be their current problem ?

Communication between the janitors and the back officers may meet some constraint as there is still no unified system.

There is no effective mechanism to make sure that all stakeholders are online all the time.

Requiring a better way on managing the operation of the whole system.

Q4. In your opinion, what benefits UWC 2.0 will be for each stakeholder ?

For janitors, providing a better way to synchronize with the collector to ensure that they never miss the garbage truck.

For collectors, the system will give them the most optimal route that saving travelling time bewteen MCPs, reduce cost spending on fuel and others expenditure.

For back officers, equipping a effective system to monitor and manage the operation of the whole system.

2.2 Requirements

2.2.1 Functional Requirements

Back officers:

View the calendar and assign task to janitors and collectors.

View the statistic of all operating garbage truck.

Send/receive message to/from other stakeholders.

Collectors:

View information about the vehicles, travelling routes of the day and notifications about the MCPs as well as report the completed tasks.

View the own assigned task and calendar.

Send/receive message to/from other stakeholders.

Janitors:

Check in when ever they arrive at the MCPs and check out when they complete collecting the garbage.

View the own assigned task and calendar.

Send/receive message to/from other stakeholders.

2.2.2 Non-functional Requirements

Efficiency:

The latency for each request is expected to be less than 1 second for all time.

The routing algorithm should respond in real time.

Reliability:

The service is likely be able to handle at least 1000 concurrent requests during peak hours.

The service downtime on working hours is minimized as much as possible (less than once per week).

Organization: The user interface should be friendly for all stakeholders.

External requirements:

Collectors and janitors will be provided with a mobile version of the system.

Meanwhile, the web version is available for the back officers.

2.2.3 Use-case diagram for the whole system

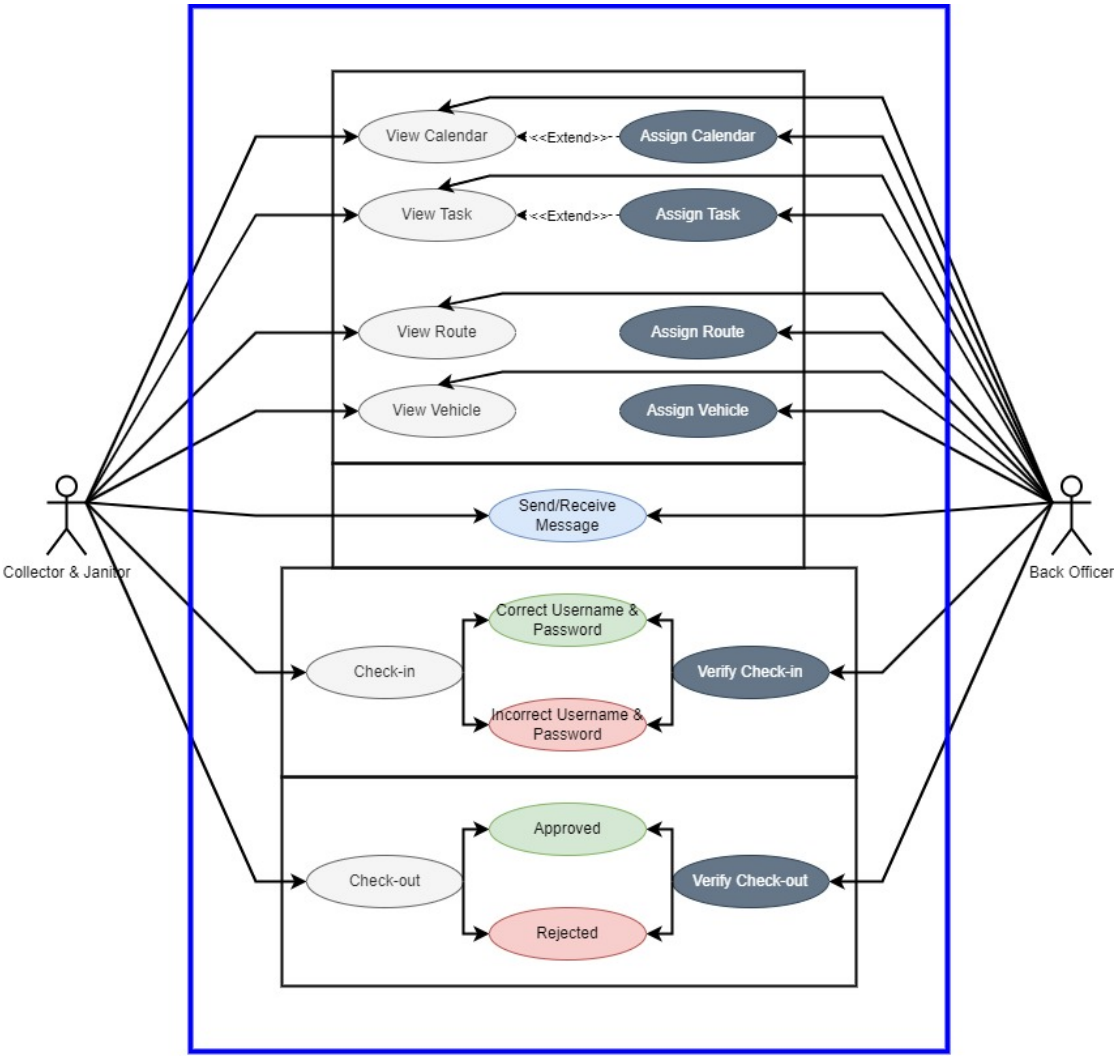

Figure 1: Use-case diagram for the UWC 2.0 system

2.3 Detailed task assignment module use-case

2.3.1 Displaying information

2.3.1.a Use-case table

Use-case ID	1
Use-case Name	Display necessary information for all Actors
Actors	Back officers, collectors, and janitors

Description	The system continuously provide information about the working calendar, the tasks that have been assigned. Especially for the collectors, they can also view all the route that need to be pass through and the status of the vehicle. On the other hand, back officers can take an overview of all the employees, vehicles and MCPs.
Pre-condition	The mobile app and website are ready to use.
Normal flow	**Step 1:** The users launch the mobile app / using any web browser to access the system's website (for back officers) **Step 2:** Users on both platform will log in to the server **Step 3:** On the "Home" page, choosing "View" section then a list of calendar, task, MCPs, vehicle will appear **Step 4:** Selecting one of the item above will display the desired information
Exception	None
Alternative flow	None

2.3.1.b Use-case diagram

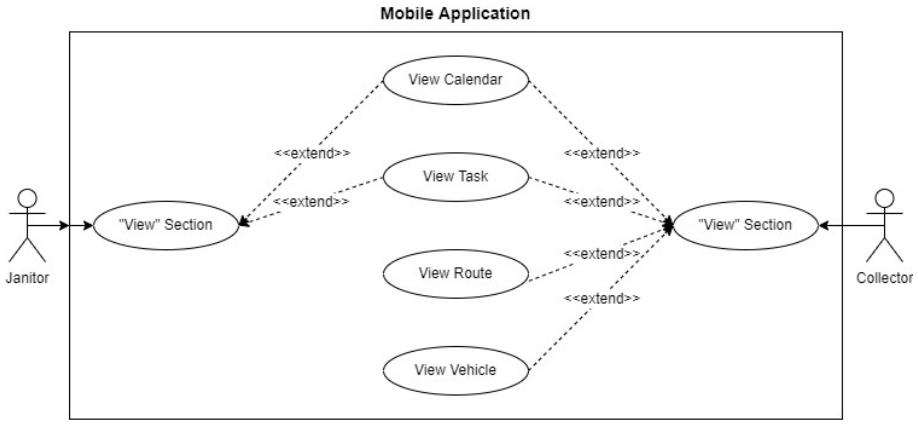

Figure 2: Use-case diagram for Displaying information module on mobile application

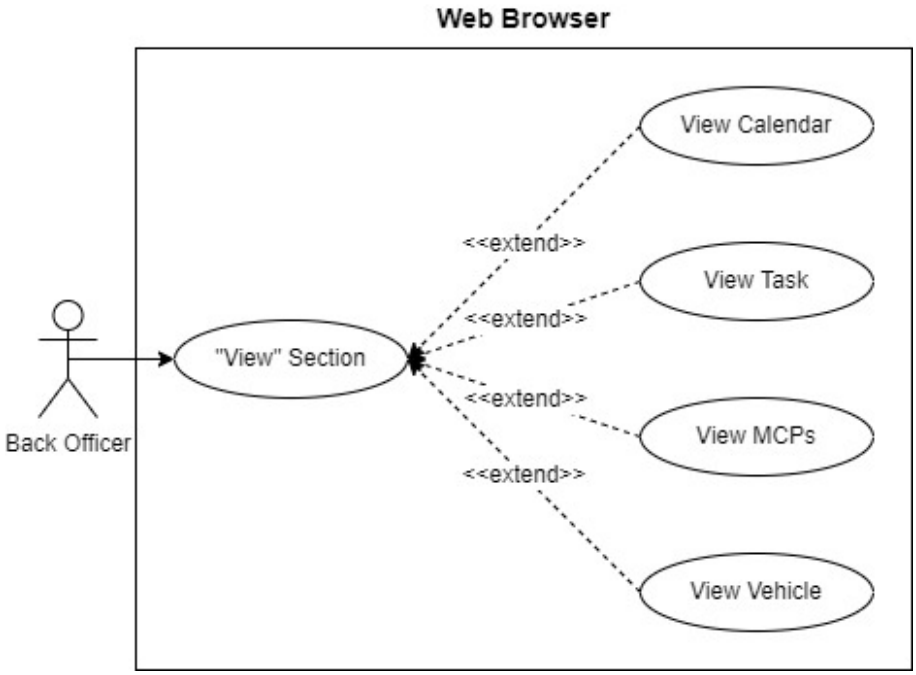

Figure 3: Use-case diagram for Displaying information module on web browser

2.3.2 Assigning the operation

2.3.2.a Use-case table

Use-case ID	2
Use-case Name	Assigning the operation is specifically used by back officers
Actors	Back officers
Description	Back officers can assign the task, changing calendar and allocating the vehicles. They can also edit a route base on the suggestion produced by the algorithm in term of optimizing the fuel consumption and travelling distance.
Pre-condition	The mobile app and website are ready to use.
	Changing calendar:

	Step 1: Back officers log-in the system through the website and click on the "Assign" tab at the home screen. Step 2: On the "Assign" tab, choosing "Changing calendar". Step 3: A list of all employees will appear.Click on the icon of the person whose calendar need to be changed. Step 4: A calendar presented in tabular form with the date at each cell showing the current assignment. By choosing the date, back officers can allocated the assignment to another date. **Assigning task:** Step 1: Back officers log-in the system through the website and click on the "Assign" tab at the home screen. Step 2: On the "Assign" tab, choosing "Assigning task". Step 3: A list of jobs that have not been appointed shows up. Click on the desired job. Step 4: Choose from the list of available employees to assign the task. **Editing route:** Step 1: Back officers log-in the system through the website and click on the "Assign" tab at the home screen. Step 2: On the "Assign" tab, choosing "Editing route". Step 3: A map of all current operating route pops up. Click on the route that need to be changed. Step 4: Back officers then need to pick a new destination and the system will suggest all possible route. Step 5: Back officers pick one new route. **Allocating vehicles:** Step 1: Back officers log-in the system through the website and click on the "Assign" tab at the home screen. Step 2: On the "Assign" tab, choosing "Allocating vehicles". Step 3: A list of all available vehicles appears. Back officers then choose the one that need to be allocated. Step 4: A list of all available collectors also presents. Back officers will appoint one collector to the pre-selected vehicle.
Exception	+ **Changing calendar:** There is no others available date to move the task. + **Editing route:** Tracking device may not work properly. As a consequence, the system can not provide the best route.
Alternative flow	**Changing calendar - No more available date:** At step 4: 4.1 A message inform that this employee can not be changed calendar. 4.2 An option appear suggest whether the back officers want to choose another employee.

	Editing route - Tracking device is not working properly: At step 3: 3.1 The map also show which route is not updated with real-time data. 3.2 An option is available for the back officers to send message to the collector asking them to provide information about their route.

2.3.2.b Use-case diagram

2.3.3 Real-time chatting functionality

2.3.3.a Use-case table

Use-case ID	3
Use-case Name	Real-time chatting
Actors	Back officers, collectors, and janitors
Description	All actors are able to communicate instantly on one unified platform, which is a part of a system.
Pre-condition	The mobile app and website are ready to use.
Normal flow	Step 1: The back officers and employees will log-in to the system through their corresponding platform. Step 2: At the Home screen, choosing "Message" tab. Step 3: A list of all available contacts shows up. (For back officers: A list consists of all employees in the company, For employees: A list only display the contact in the same team) Step 4: Choosing the contact that they want to send message.
Exception	The system exceeds it limits which may lead to a sudden crash.
Alternative flow	N/A

2.3.3.b Use-case diagram

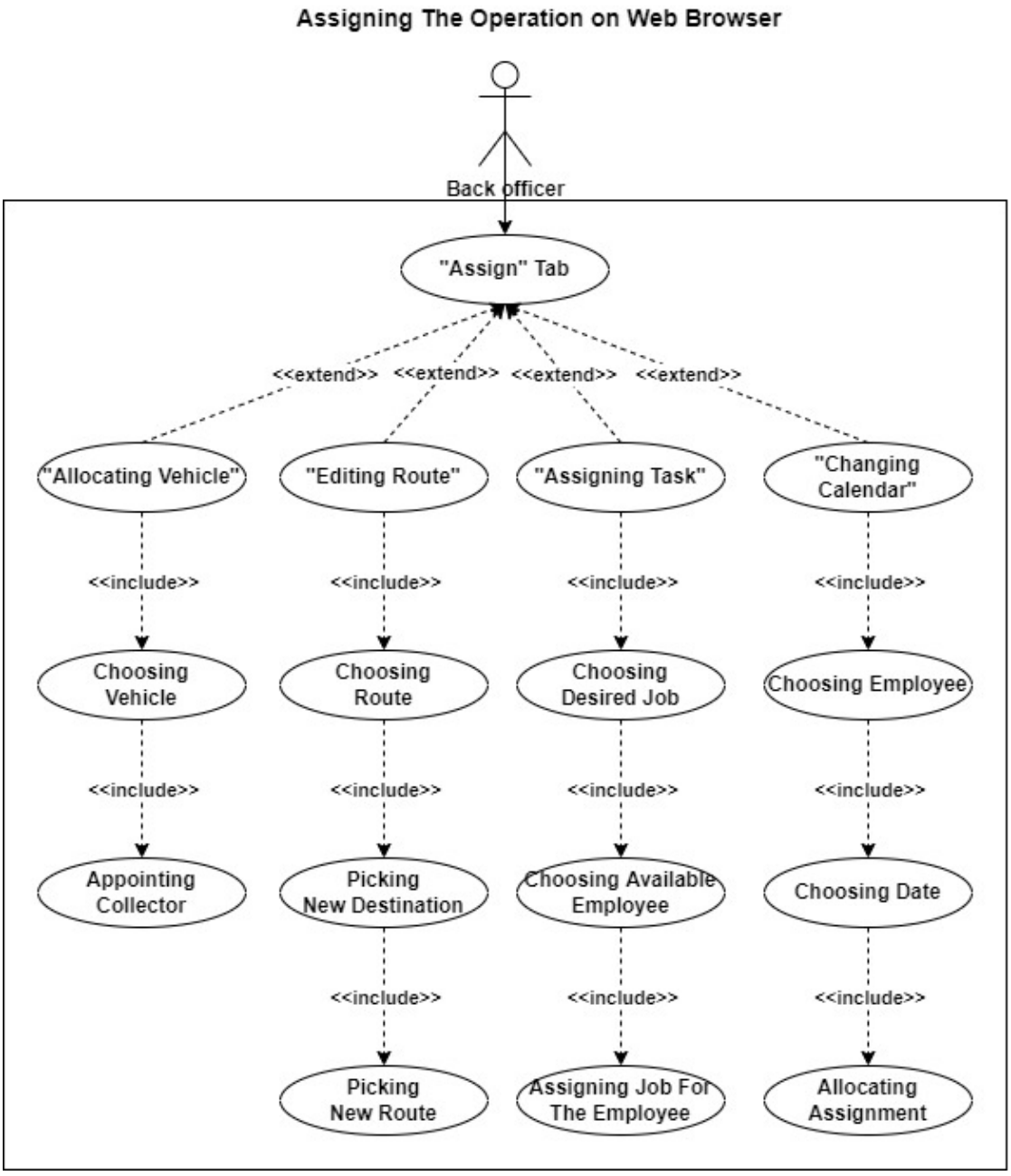

Figure 4: Use-case diagram for Assigning the operation module on web browser

2.3.4 Log-in for back officers

2.3.4.a Use-case table

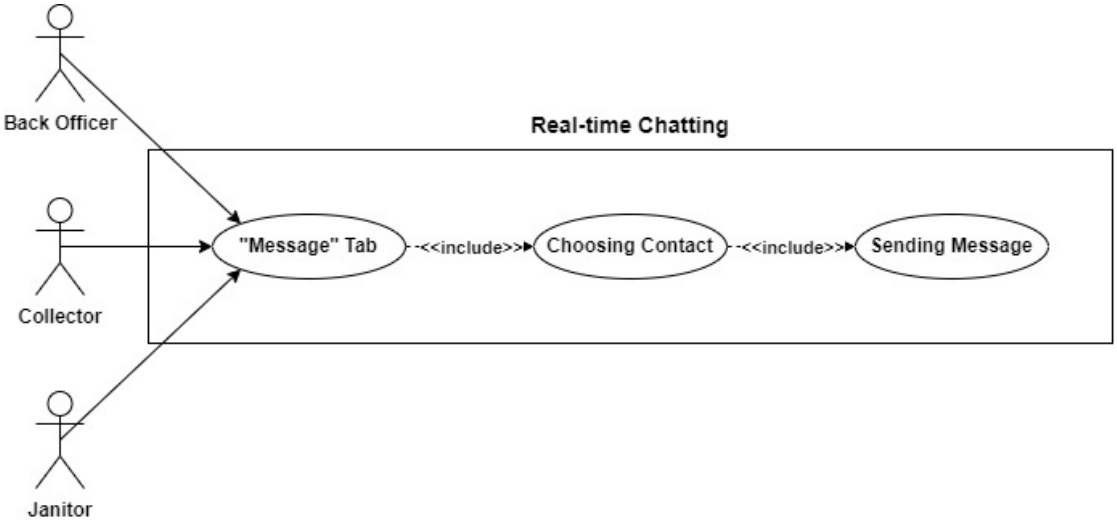

Figure 5: Use-case diagram for Real-time Chatting module

Use-case ID	4
Use-case Name	Log-in for back officers
Actors	Back officers
Description	Creating account for the back officers if they still don't have one. Allow access to the system if the account has already existed.
Pre-condition	The mobile app and website are ready to use.
Normal flow	Step 1: At the log-in page, type in user name and account. If the back officers haven't had one yet go to step 2. Step 2: Choosing the "Creating Account" option instead of typing in account. Step 3: Choosing distinctive user name and password and choose "Submit". Step 4: Refresh the page and type in the newly created account.
Exception	None

Alternative flow	None

2.3.4.b Use-case diagram

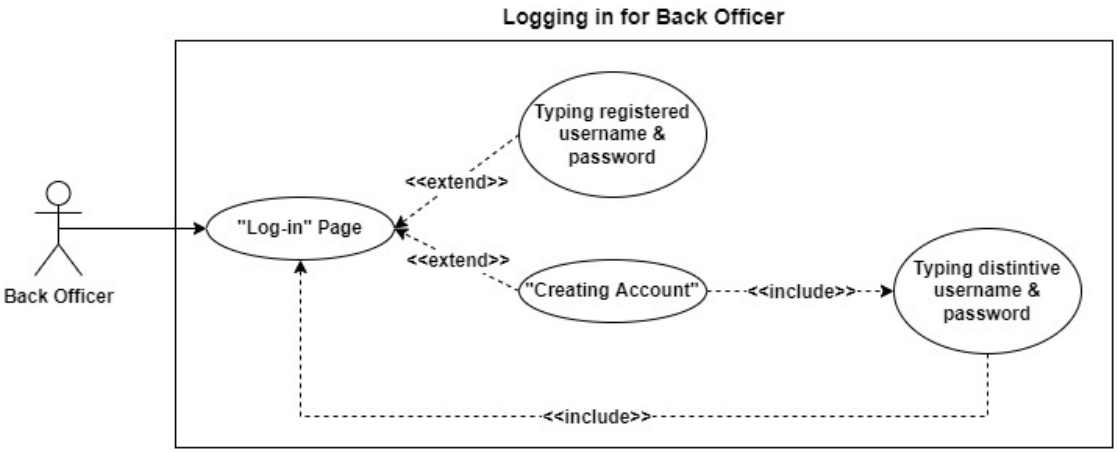

Figure 6: Use-case diagram for Real-time Chatting module

2.3.5 Check-in and Check-out

2.3.5.a Use-case table

Use-case ID	5
Use-case Name	Check-in and Check-out
Actors	Collectors, and janitors
Description	Pushing notification to the back officers when the janitors or collectors reach the MCPs and let the janitors or collectors report their completed work.
Pre-condition	The mobile app and website are ready to use. Employees need to meet back officers to create account.
Normal flow	Step 1: Log-in to the system through mobile app. Step 2: Choosing the check-in/check-out section.

	Step 3: A map showing all operating MCPs shows up. Employee choosing their current location. Step 4: For check-in, press check-in button. For check-out, type in the capacity of the MCPs then press check-out.
Exception	None
Alternative flow	None

2.3.5.b Use-case diagram

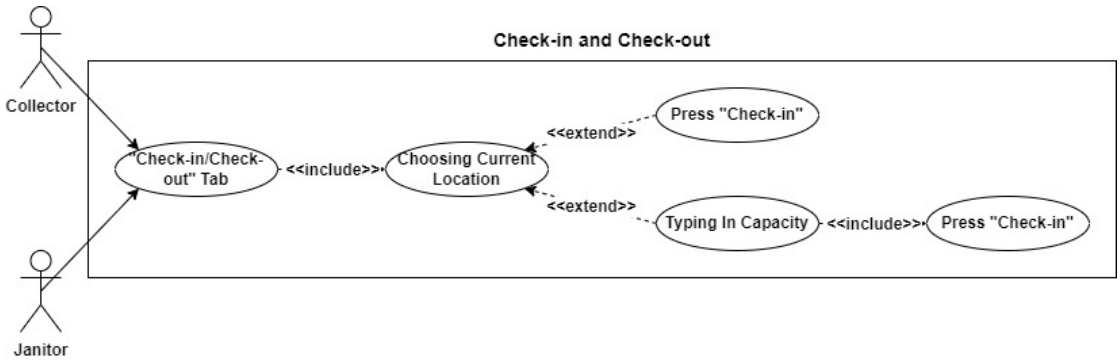

Figure 7: Use-case diagram for Check-in/Check-out module

3 Task 2

3.1 Activity diagram for capturing the business process between the system and stakeholders

Since the schematic has a limit, we will only sketch the activity diagram to illustrate major functional requirements of our desired system, which are "View" information and "Assigning" the operation as they are critical for an effective management. And also in the "Assigning" section, we only draw a detailed schematic for the first operation.

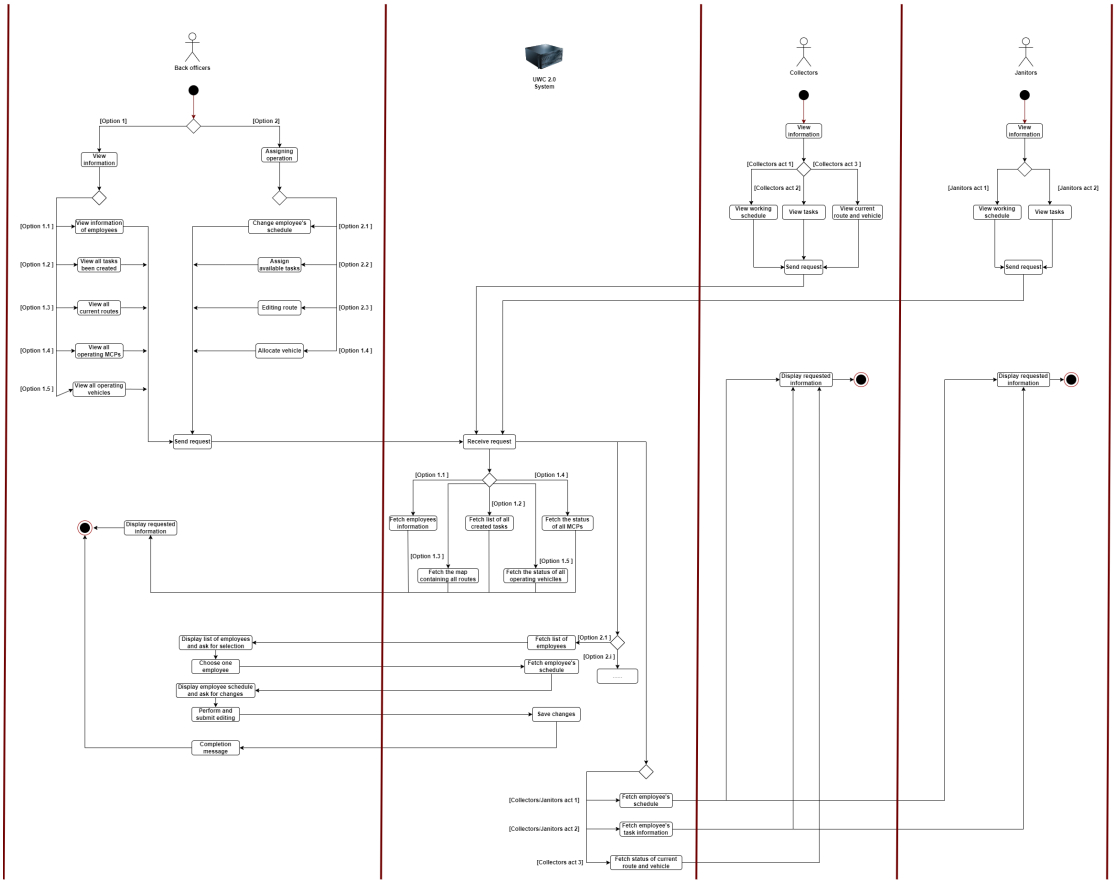

Figure 8: Activity diagram for core operation of the system (If the image is not visible, please zoom in for clearer view).

3.2 Route Planing Proposal Solution

Back officers will send their collectors schedule(the starting point and ending point) as the position of MCPs and vehicle at a specific time and then the sever will sort in priority orders (applying some sorting algorithm) to return the route to officers.

In case the officers want to view the route if they're suitable or not, the server will receive a request then return the information needed. In the case of any change happen due to collectors, officers just need to resent the current schedule and real time solution will be provided in order to give the best routes. Beside, the schedule of collectors can also be updated automatically base on the last route returned to ensure the route system will always be real time.

The server will be using MySQL database which is a place not only to store the data but also they can be sort with given condition so that the data will be return in needed order. Beside, the API using for communicating and controlling database will be written in PHP so that we can use MySQL command to communicate with the database. The benefits of using this method is that we don't need to focus on any sorting algorithm but only care about communicating with the database which make the processing time hasten. Also, the data can be sort automatically at anytime comparing to writing sorting API and get the raw data from server , this will not also save the time but also save us a lot of money especially when the number of data becomes enormous. All of the ideas can be summary below:

- The database is running on a server, often on a distributed system and so it has access to more resources.This make the processing speed increase as the accuracy can also be improved.
- Databases are designed to handle large data, so they are not limited by the memory in a single thread. Meanwhile, API can only handle limited data in a specific time
- Often more data needs to be passed back to the application than itself strictly needed. Consider a problem such as getting the top 5 of something.
- Mixing processing in the application and the database often requires multiple queries and passing data back and forth, which is expensive and cost a lot of resources to give maintenance to the system.
- There are some situations where it might be more efficient or convenient to do work in the application or in API. A common example would be when the application language has capabilities that are not directly in MySQL or are hard to implement in MySQL. That's why we have PHP as the language to write the API which is fully supported and easy to write.

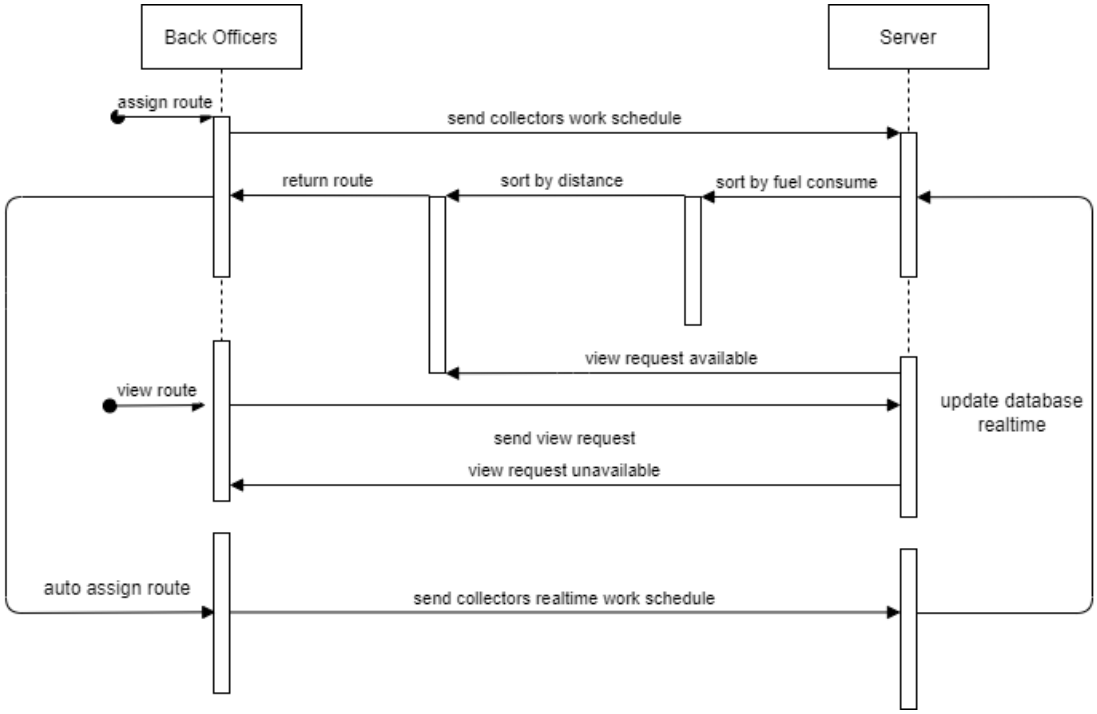

Figure 9: The conceptual model.

3.3 Task assignment module - Class diagram

3.3.1 Displaying information

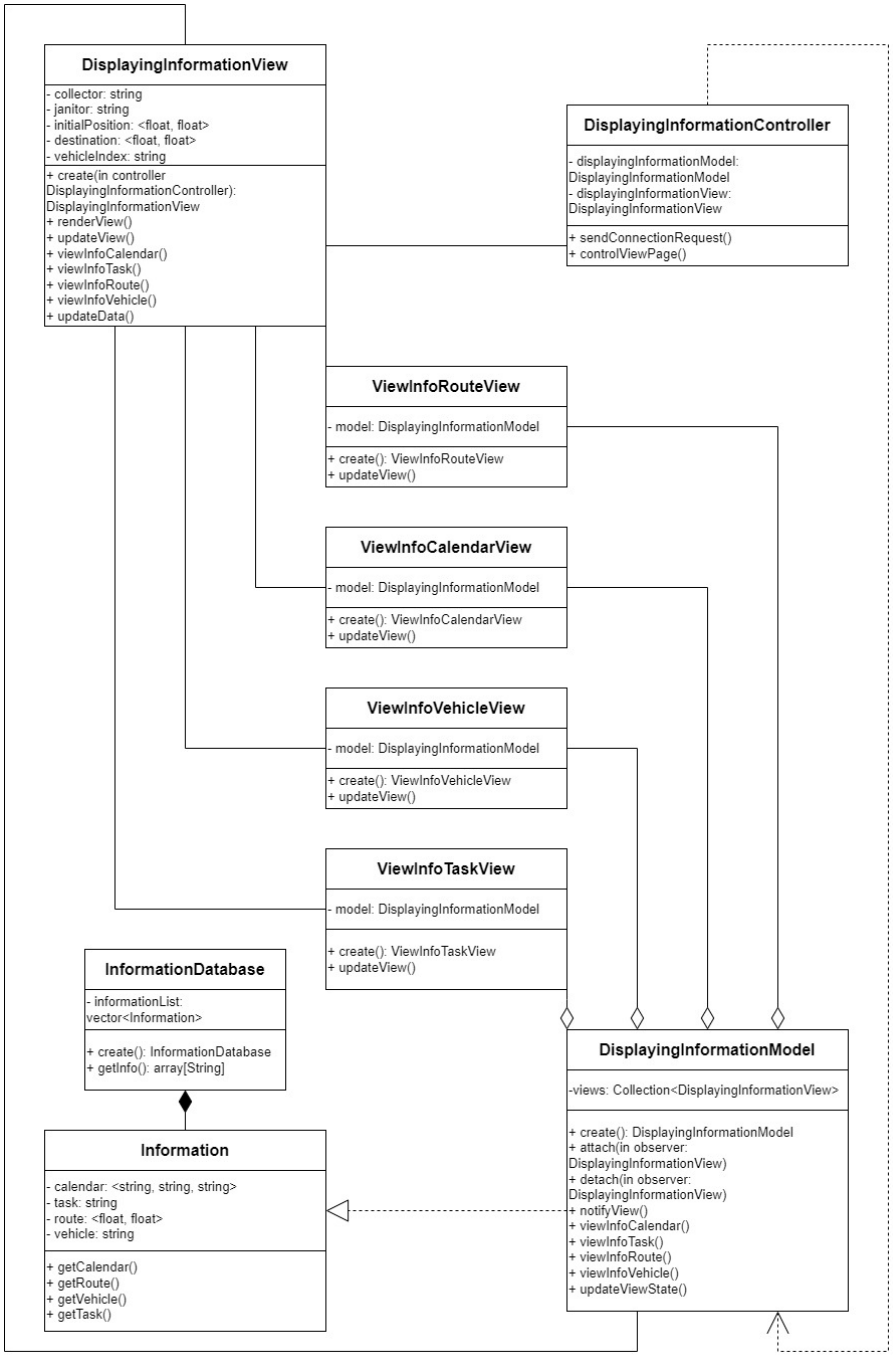

Figure 10: Displaying information class diagram

3.3.2 Assigning the operation

3.3.2.a Changing Calendar

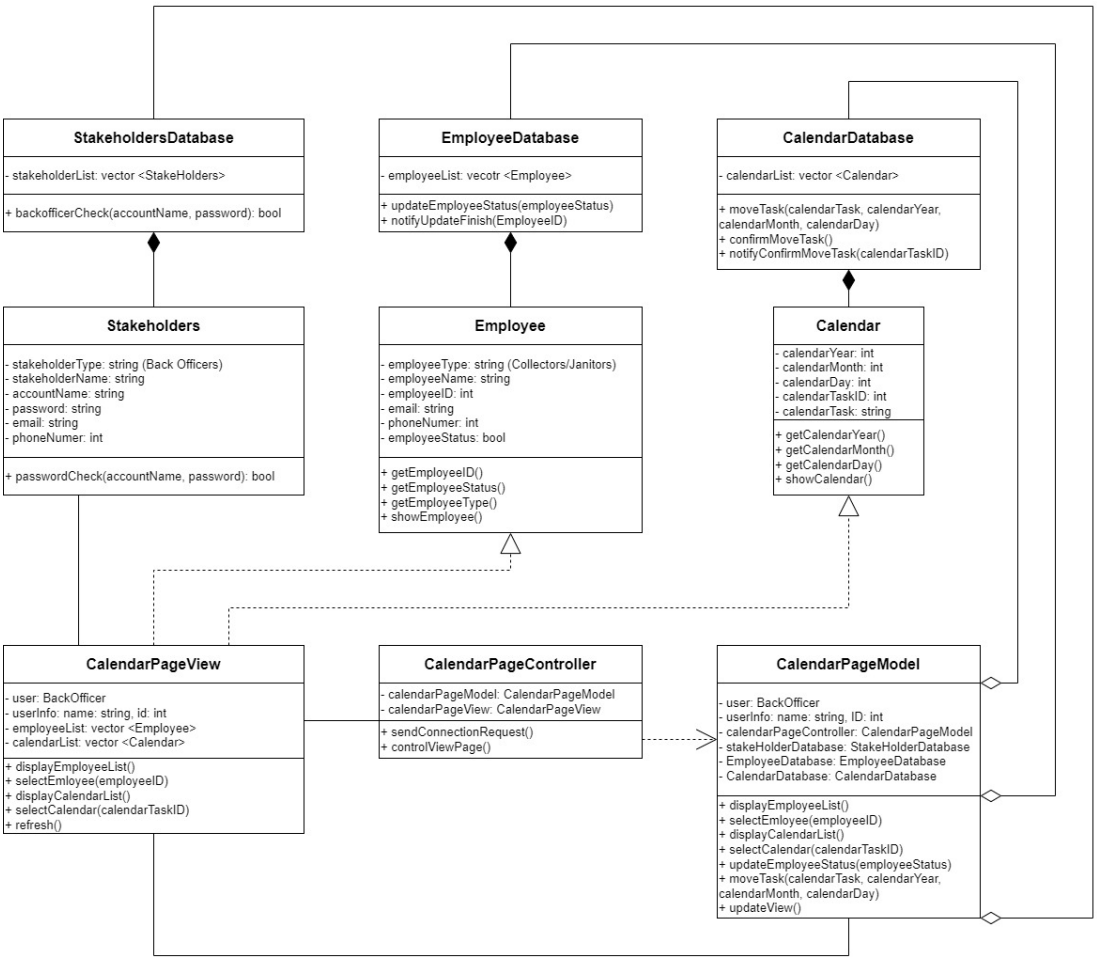

Figure 11: Changing Calendar class diagram

3.3.2.b Assigning task

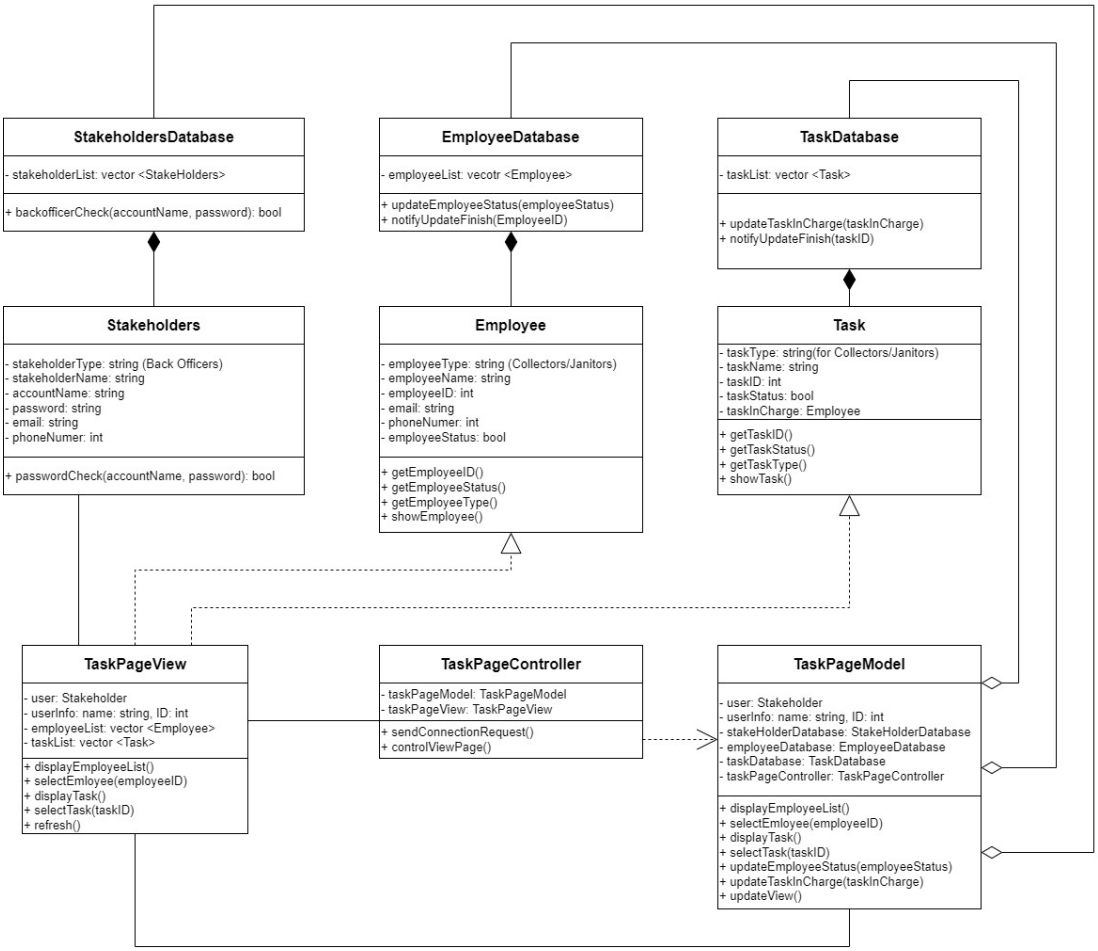

Figure 12: Assigning task class diagram

3.3.2.c Editing route and Allocating vehicles

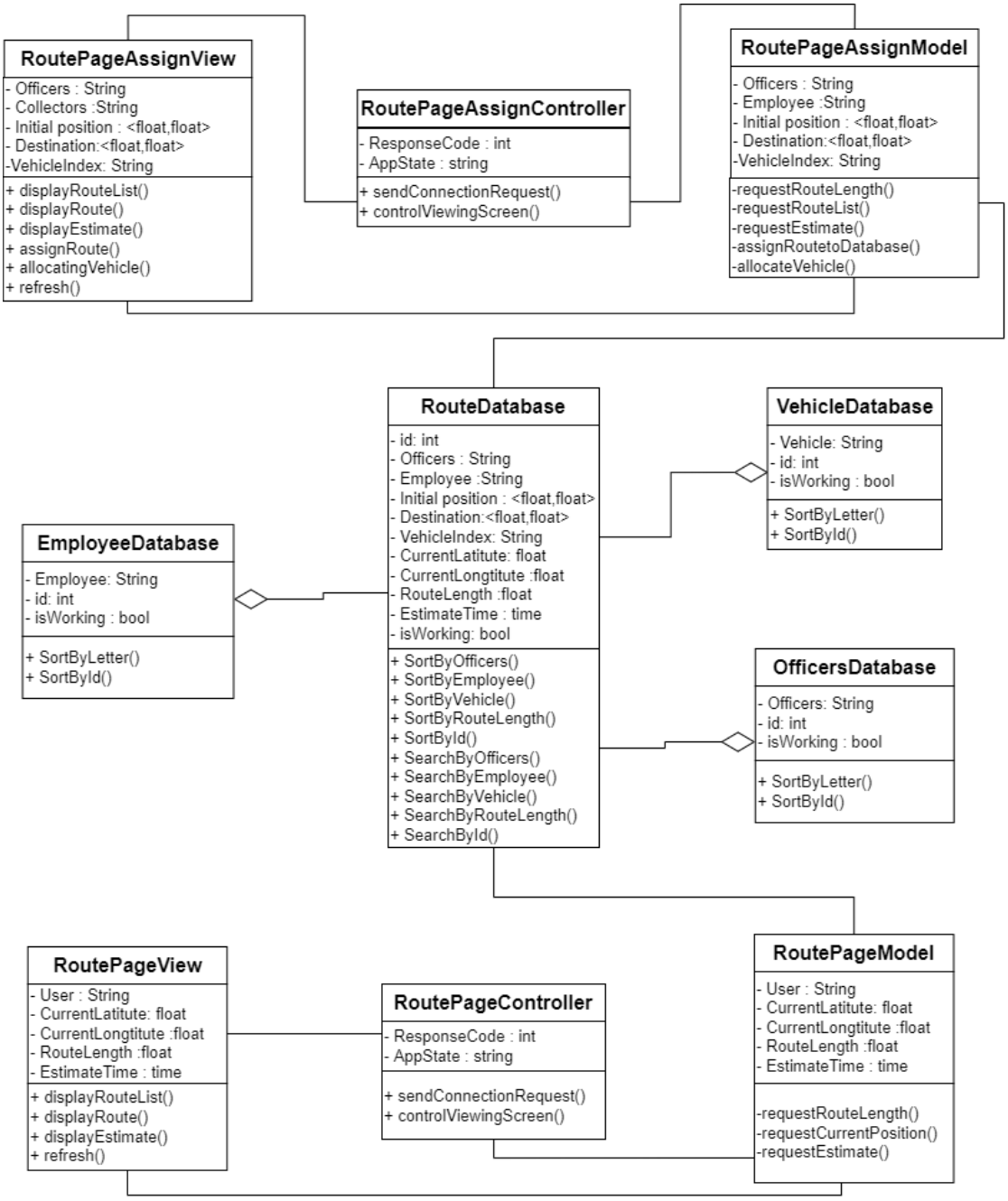

Figure 13: Editing route and Allocating vehicles class diagram

3.3.3 Real-time chatting functionality

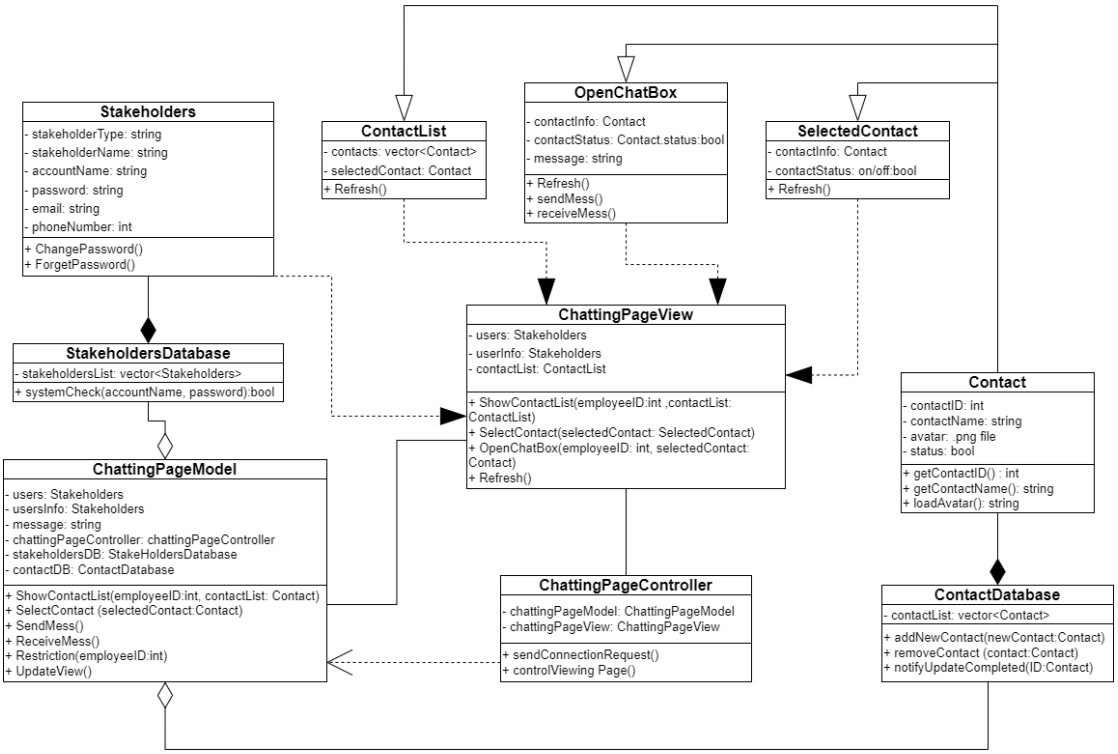

Figure 14: Real-time chatting functionality class diagram

3.3.4 Log-in for back officers

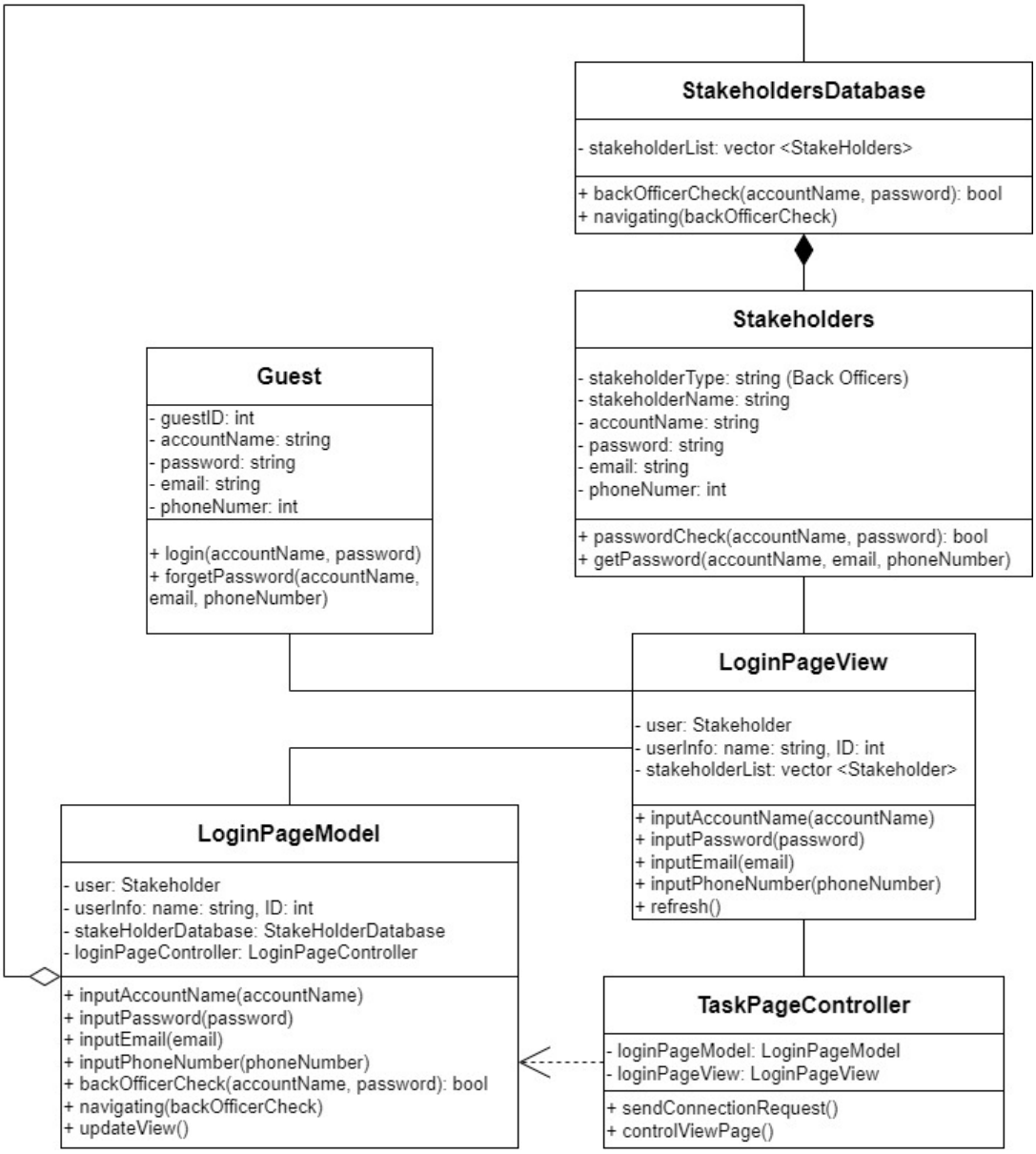

Figure 15: Log-in for back officers class diagram

3.3.5 Check-in and Check-out for collectors and janitors

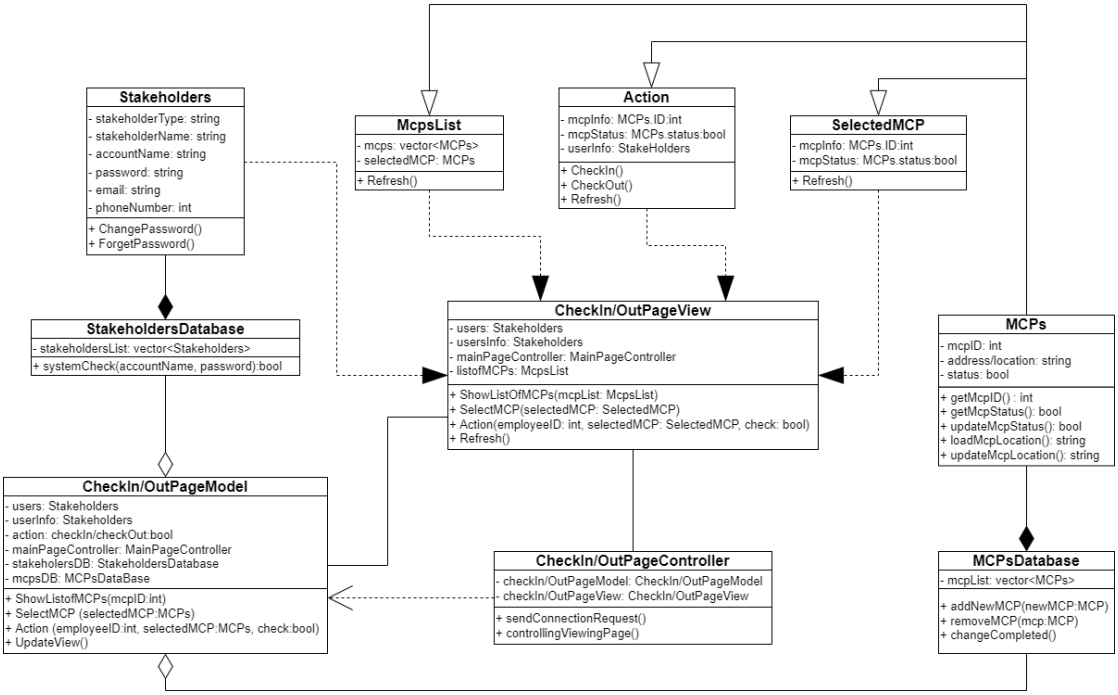

Figure 16: Check-in and Check-out for collectors and janitors class diagram

4 Task 3

4.1 Describe an architectural approach will be used to implement the desired system

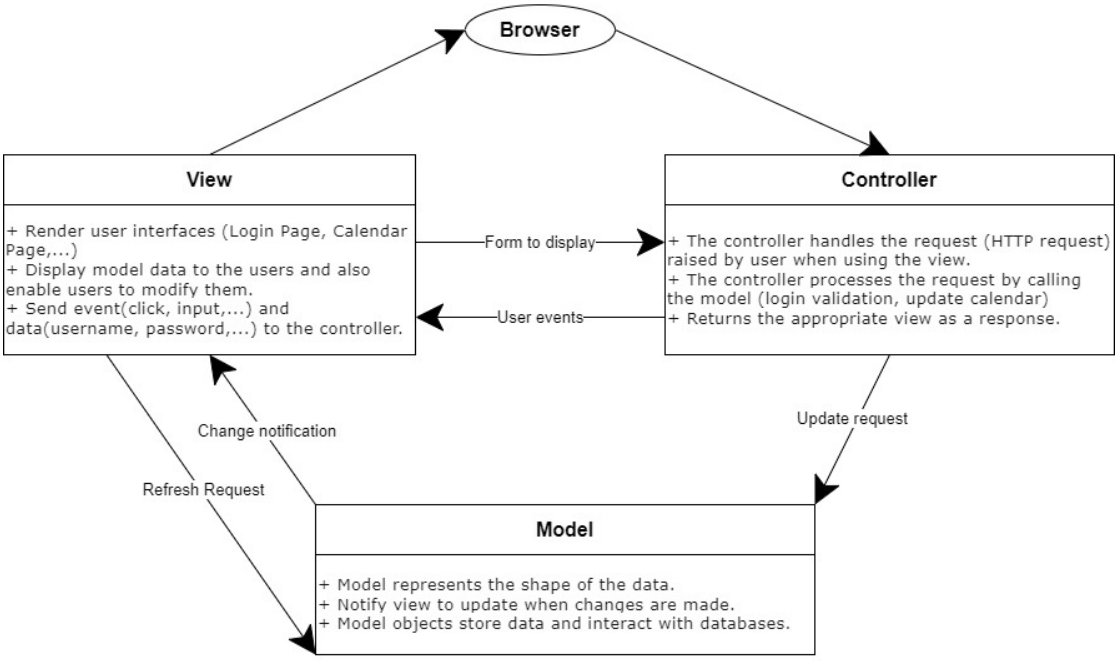

Figure 17: MVC Architecture

Name	MVC Architecture Details
Benefits	- Organizes large-size web applications – Because the code is divided into these three levels, it is very simple to separate and arrange the logic of web apps into complex applications. It makes it simple to add new functionality and makes it easier to locate certain code sections. - Supports Asynchronous Method Invocation (AMI) – The MVC design facilitates the usage of AMI and integrates well with JavaScript and its frameworks, enabling programmers to create web applications that load more quickly. - Easily Modifiable – The new type of views are simpler to add and update. Therefore, modifications made to one part of the application will never modify the architecture as a whole. This will improve the application's adaptability and scalability.

	- Faster Development Process – Since there are three levels, two developers can work on any three levels at once, each on a separate section. This makes it simple to apply business logic and contributes to a four-fold acceleration of the development process. - Easy planning and maintenance – It provides a roadmap for the developer on how to structure their concepts into actual code. Additionally, it is a great tool for reducing code duplication and making application maintenance simple. - Returns data without formatting – The MVC framework gives you the ability to develop your own view engine by returning unformatted data. So the same components can be re-used with any interface. - Supports TTD (test-driven development) – Large-scale programs can be more easily debugged since the application's multiple levels are well defined structurally and appropriately written. - Multiple Views – It gives you the ability to create several view components, preventing code duplication by separating data from business logic.
Drawbacks	- The separate development processes used by the business logic, controller, and user interface (UI) teams could cause delays in the deployment process, which would delay getting the product to the client. - A large number of people with a variety of skills are needed to implement MVC because it involves knowledge of multiple languages and technologies. - Code maintenance is quite difficult because the Controller contains all of the code. - The Views can be overloaded with requests for updates. Furthermore, because they are difficult to update rapidly, views like graphic displays can take longer to create. The view component consequently lags behind any significant modifications.

4.2 Draw an implementation diagram for major (not all) functional requirements

4.2.1 Assigning the task

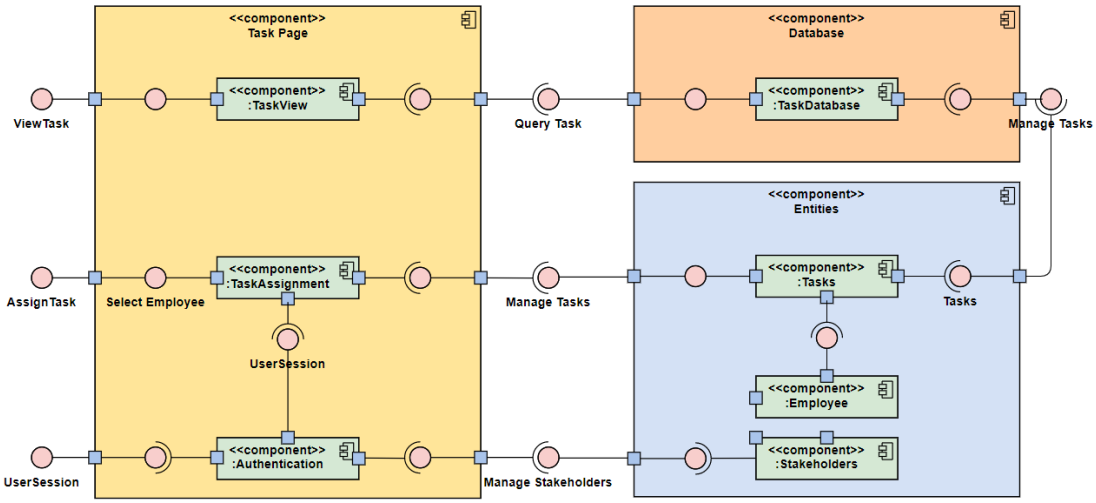

Figure 18: Component Diagram of Assigning Task

5 Task 4

5.1 Setting up. The team creates an online repository (github, bitbucket, etc) for version control. Folders this stage, no need for a database to store all menu items, customers, etc. Data can be hard coded in code files.

Here is the layout of our GitHub repository:

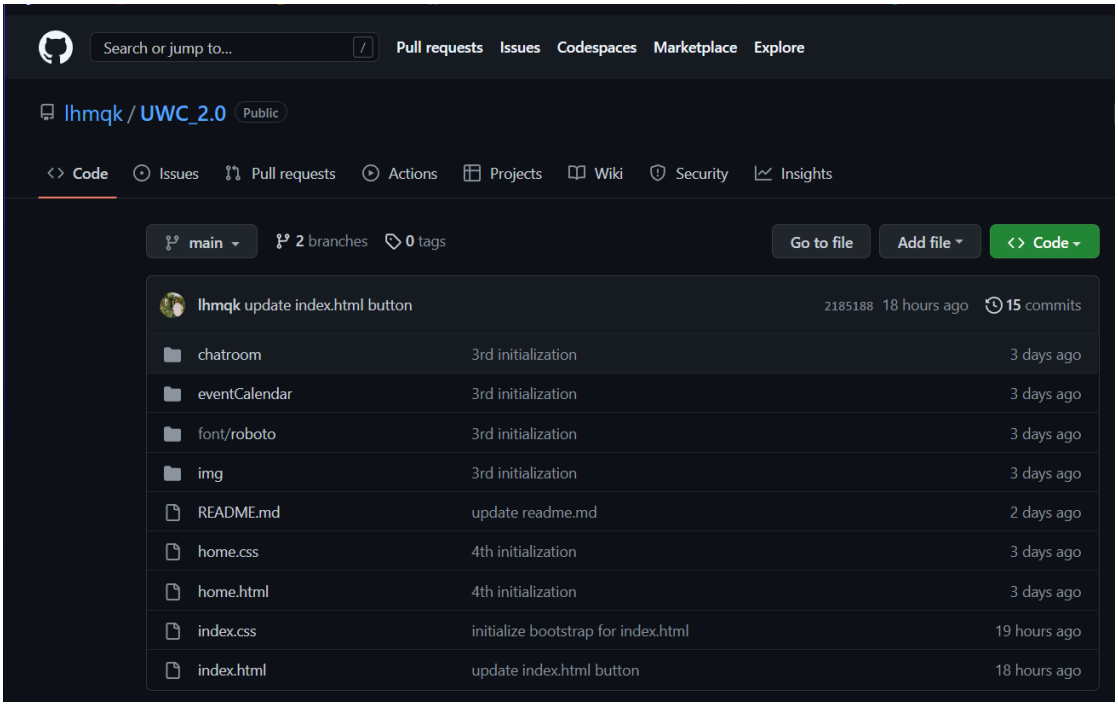

Figure 19: GitHub Repository layout

And this is the log of our version control.

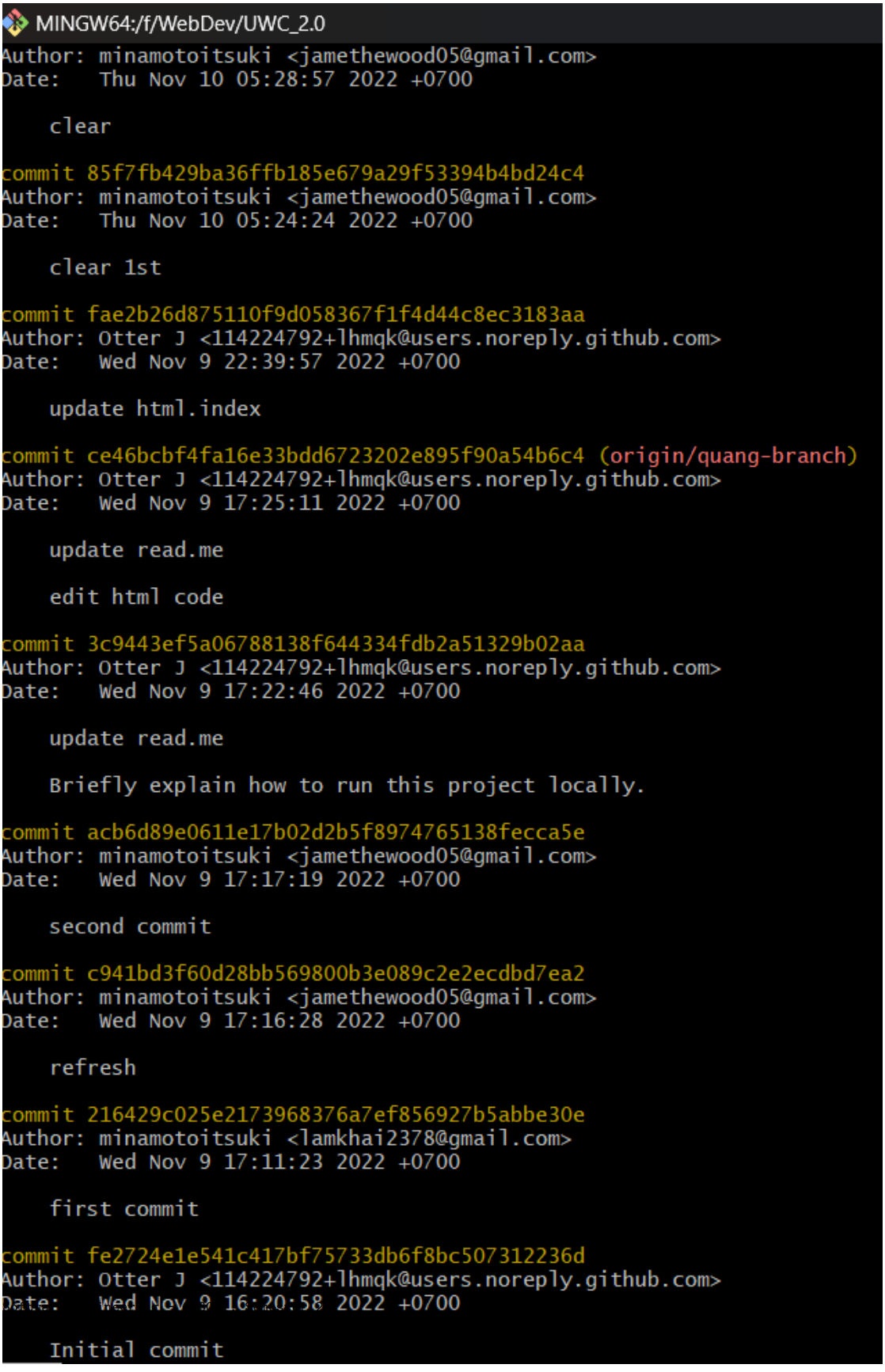

Figure 20: GitHub Repository layout

5.2 Adding documents, materials and folders for Requirement, System modelling and Architectural design. Use the selected version control system to report the changes to these files.

In order to gain the full control of our project, please access the following link https://github.com/lhmqk/UWC_2.0.

All the equivalent documents and materials is included in one single pdf file.

5.3 Implement MVP1 – design an interface of either a Desktop-view central dashboard for Task Management for back-officers OR a Mobile-view Task assignment for Janitors and Collectors. Decide yourself what to include in the view. Design use a wireframe tool

To reduce the time consumption and optimize our work, we have decided to skip this part. Instead of that, we will utilize the design process by using bootstrap. Using the Bootstrap framework saves time in many ways by taking advantage of reusable code for Navbars, Dropdowns, Labels, Alerts, List groups and JavaScript plugins. Using the framework offers these design benefits:

- Easy to prevent repetitions among multiple projects
- Responsive design that can be used to adapt screen sizes and choose what shows and what doesn't on any given device
- Maintaining consistency among projects when using multiple developer teams
- Quick design of prototypes
- Cross-browser compatibility

Bootstrap offers a vast selection of plugins and components from the open source community, so it just takes a few clicks to begin using your front-end resource. Staff members can save many hours that are usually needed for repetitive coding. The benefits of the Bootstrap framework include