Ho Chi Minh City University of Technology

Faculty of Computer Science and Engineering

Software Engineering (CO3001)

Assignment Report

URBAN WASTE COLLECTION AID UWC 2.0

Lecturer: Quản Thành Thơ

Group name: Take the trash out!!!

Team members: Trương Tân Hào Hiệp – 2011211

Nguyễn Phạm Hoàng Vũ – 2052324

Lê Phước Gia Lộc – 2052155

Nguyễn Quốc Huy – 2053044

Trương Huỳnh Đăng Khoa – 2053145

Ho Chi Minh city, 21 September, 2022

TABLE OF CONTENTS

TEAM EVALUATION

Member	ID	Workload	%	Score
Trương Huỳnh Đăng Khoa	2053145	Report(4) + Diagram(5) + App(7) + Misc(5)	21	
Trương Tân Hào Hiệp	2011211	Report(6) + Diagram(5) + App(4) + Misc(6)	21	
Nguyễn Quốc Huy	2053044	Report(4) + Diagram(5) + App(7) + Misc(6)	21	
Nguyễn Phạm Hoàng Vũ	2052324	Report(5) + Diagram(5) + App(5) + Misc(5)	21	
Lê Phước Gia Lộc	2052155	Report(3) + Diagram(3) + App(3) + Misc(5)	16	

TASK 1

1.1. Requirement elicitation

1.1.1. Identify the context of this project.

In urban contexts, solid waste management is costly and ineffective. Improvement of waste collection and management is emphasized by governments and organizations for positive impacts on cities, societies and environments. Therefore, Urban waste collection aid - UWC 2.0 is created to help management and solid waste collection processes be more effective and economical.

1.1.2. Who are relevant stakeholders? What are their current needs? What could be their current problem?

Stakeholders	Problems	Needs
Back officers, employee (janitors & collectors)	- Communication between co-workers is ineffective. - Non-optimal waste-collecting route. - Application doesn't have enough features for employee & back officers to work on.	- Better communication channel. - More user-friendly interfaces in application. - New features in managing the flow of work.
Application provider company	- Complaints about the inefficiency of the application.	- Update the application to meet the new requirements.

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

UWC 2.0 will be an improvement of UWC 1.0 in terms of performance, accessibility and convenience thereby satisfying the stakeholders requirements and bringing benefits to its users (including community, employees and employers).

1.2. Use-case diagram

1.2.1. Describe all functional and non-functional requirements that can be inferred from the project description.

Functional Requirements:

General requirements:

– Account management for the back officers and employee (log-in, manage personal information, ...)

– Real-time communication (through messages) between staffs.

– Changes in route (in case of busy MCPs, better route found, …) must notify employee in advance.

Back officers:

– Collectors and Janitors management.

+ View/justify collectors and janitors' personal information.
+ View collectors and janitors work calendar.
+ Add/ Remove collectors and janitors.

– Vehicles management.

+ View/ Justify technical information of vehicles (weight, capacity, ...).
+ View current status of vehicles (position, fuel, ...).
+ Assign vehicles to collectors, trollers to janitors.
+ Add/ Remove vehicles.

– MCPs management.

+ View/ Justify information of MCPs (position, capacity, ...).
+ View current status of MCPs (empty/non-empty status, ...).

– Send notification (when MCP is busy, announce route for janitors and collectors, ...).

– Task assignment.

+ Assign route for collectors.
+ Assign janitors to MCPs.

Janitors and collectors:

– Check in/ out work.

– Have an overview of work location on virtual map.

– View work calendar:

+ View general work calendar.
+ View work calendar in details (daily, weekly tasks and important informations regarding the task such as location, time, etc.)

Non-functional Requirements:

Performance

- Communication should be carried out with a delay of less than 1 second.
- At most 2 sec delays for system's responses to any interaction with users.

Scalability

The system should be able to handle real-time data from at least 1000 MCPs at the moment and 10.000 MCPs in five years.

Usability

The user interface is designed to be simple, appealing and convenient. The system has a friendly interface with simple interactive button for every particular feature to adapt all the function of the software. Furthermore, it has minimal yet enough information and images; in order for our software to be easy to use and accessible - so that users can conveniently use it without needing any prerequisite experience or knowledge.

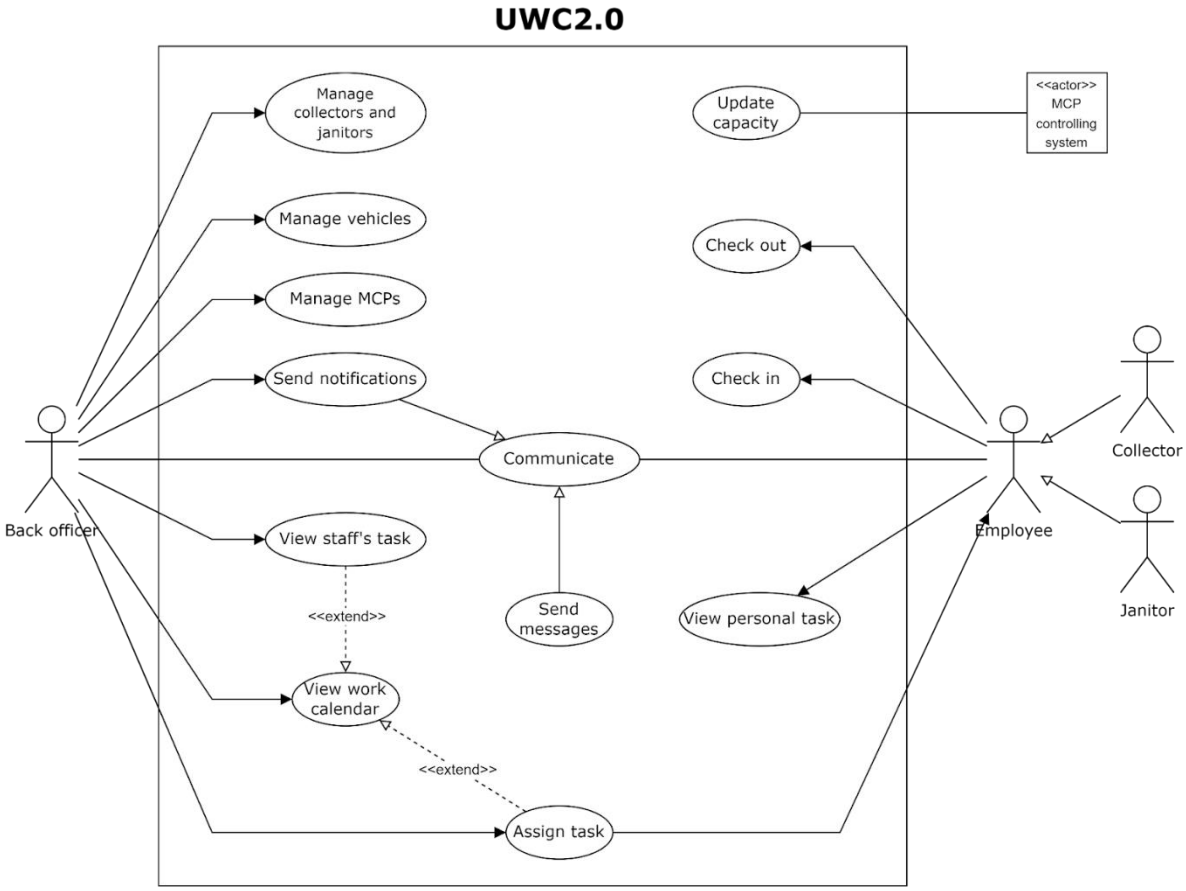
Availability

- The system must be accessible throughout regular business hours (from 7 a.m to 6 p.m). There can be no more than five seconds of downtime during regular business hours on any given day. Therefore, data is available by 7 am local time after an overnight update.
- Information should be updated from MCPs every 15 minutes with the availability of at least 95% of their operating time.

Adaptability

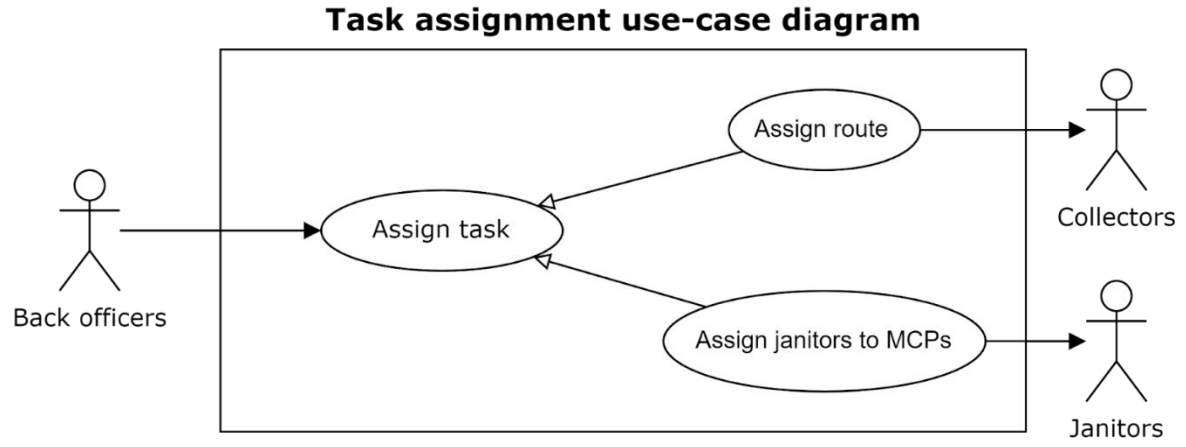
- UWC 2.0 system interfaces should be in Vietnamese, with an opportunity to switch to English in the future.
- It is expected that the Task Management be interoperable with the UWC 1.0 as much as possible.

1.2.2. Draw a use-case diagram for the whole system



1.3. Task assignment module

1.3.1. Draw the Task assignment module use-case diagram



1.3.2. Describe the use-case using a table format

Use-case name	**Assign task**
Actor	Back officers
Description	Back officers assign tasks (of a chosen area) to collectors and janitors
Trigger	Back officers chose an area in the areas list (from "Task management menu")
Preconditions	Back officers have signed in using legitimate system accounts commensurate with their position.
Normal flow	1. The system displays the "Task assignment page", which includes: - Map of area's MCPs. - Area's task calendar. 2. Back officers interact with either component in the menu. 3. System changes/modifies the interface based on the user interactions and moves to the relevant use case.
Postconditions	- "MCP symbol" selected: Move to "Assign Janitor to MCPs" use-case. - Work shift entries (in area's task calendar) selected: Move to "Assign Collector" use-case.
Exceptions	
Alternate flows	

Use-case name	**Assign route**
Actor	Back officers

Description	Assign route to collectors in designated work shifts.
Trigger	Back officer clicked on a “work shift” entry (in the task calendar from “Task assignment page”).
Preconditions	Back officers have signed in using legitimate system accounts commensurate with their position.
Normal flow	1. System displays a list of available collectors. 2. Back officer assigns a collector from the list to the selected work shift. 3. System assigns an optimized route to the collector. 4. System notifies the collector about the newly assigned work.
Postconditions	Collectors are assigned to selected work shift with optimized routes.
Exceptions	Exception: at step 2 2a. Back officer clicks the ‘X’ button to close the list of available collectors. 2b. Continue with the “Task assignment page”.
Alternative Flows	Alternative Flow: at step 3 3a. Back officer choose Action button to go into ChooseRoute overlay. 3b. Back officer manually assign systematically optimized route to the chosen collector. 3b. Continue step 3 in normal flow.

Use-case name	**Assign janitors to MCPs**
Actor	Back officers
Description	The back officers assign janitors to MCPs.
Trigger	Back officer clicks the “MCP symbol” (on the map from “Task assignment page”).

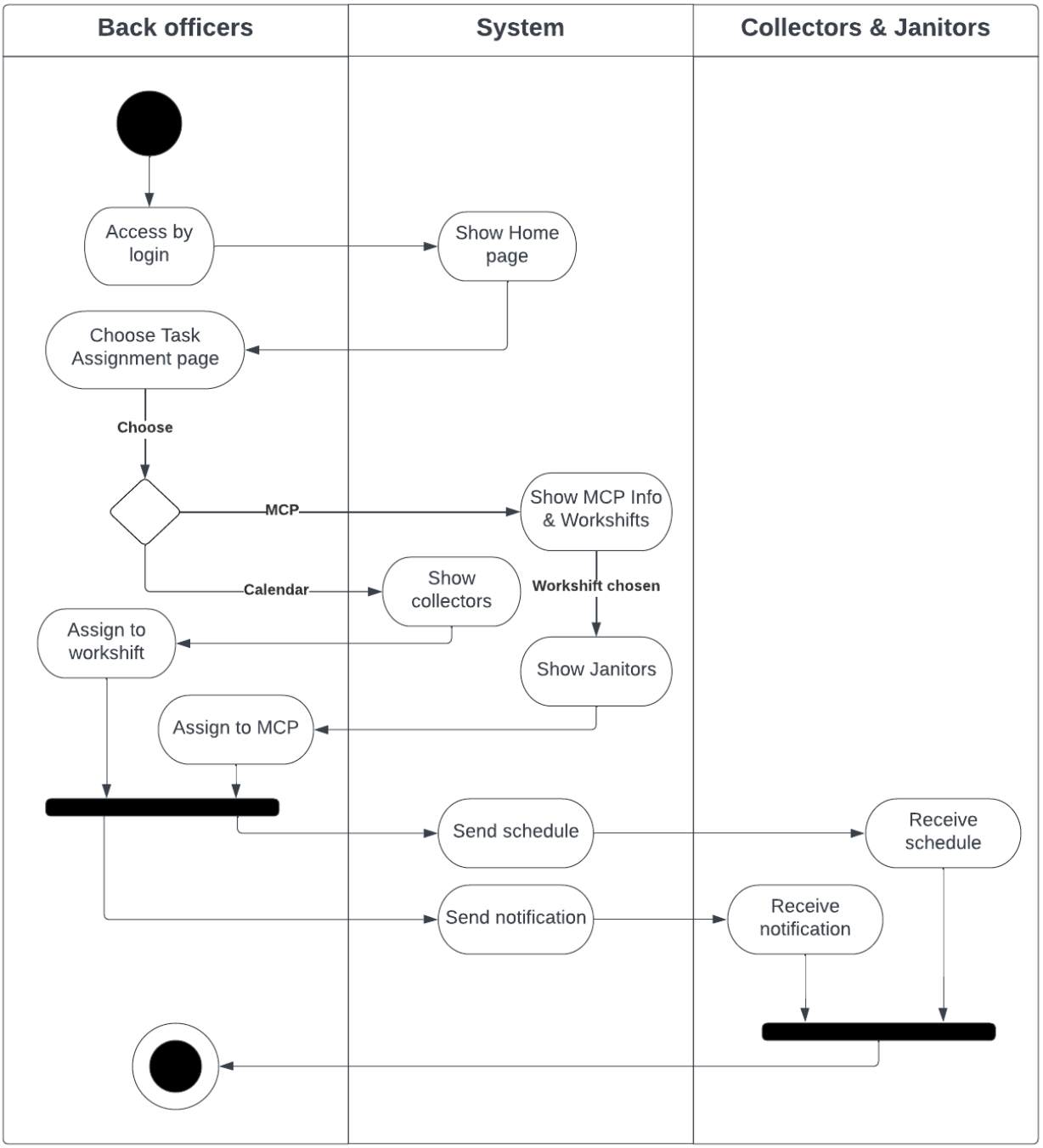
Preconditions	Back officers have signed in using legitimate system accounts commensurate with their position.
Normal flow	1. The system displays the “MCP management menu”, which includes: - MCP’s information (capacity, status, …). - MCP’s task calendar. 2. The Back Officer chooses a work shift from MCP’s task calendar. 3. The system shows a list of available janitors to that work shift. 4. The Back Officer assigns janitors to that work shift. 5. System notifies the janitor about the newly assigned work.
Postconditions	Collectors are assigned to selected work shifts and MCPs.
Exceptions	Exception 1: at step 2 2a. Back officer clicks the ‘X’ button to close the MCP’s management menu 2b. Continue with the “Task assignment page”. Exception 1: at step 4 2a. Back officer clicks the ‘X’ button to close the list of available janitors. 2b. Continue with the “MCP Info overlay”.
Alternative Flows	Alternative Flow 1: at step 2 2a. Back officer clicks on the symbol of another MCP. 2b. Continue step 1 in normal flow. Alternative Flow 2: at step 4 4a. Back officer clicks on a different work shift. 4b. Continue step 3 in normal flow.

TASK 2

2.1. Activity diagram

2.1.1. Draw an activity diagram to capture the business process between systems and the stakeholders in Task Assignment module

Activity Diagram:



2.1.2. Description

– Back Officer's identification has been verified, and the system will now display the “Area task assignment menu” menu selections which has “Map of area’s MCPs” and “Area’s task calendar”.

– The task's recipient is determined by the back officer. If “Map of area’s MCPs” is selected, the system will display the “MCP Info menu”, which includes: MCP’s information (capacity, status, …) and MCP’s task calendar. The back officer chooses a work shift in the MCP’s task calendar. Then, the system shows a list of available janitors to that work shift and the Back Officer assigns janitors to that work shift.

– If “Area’s task calendar” is selected instead, the system will display a list of available collectors and Back Officer assigns a collector from the list to the selected work shift. System will then assign an optimized route to the collector.

– System will assign schedule to the employee then send notification about task assigned to the employee’s calendar.

2.2. Conceptual solution for route planning task

2.2.1. Conceptual solution

Assuming that a city has many areas.

– System will show the list consisting of all areas and its status of the tasks assigned.

– Back officers can choose one of the areas to assign task.

– System shows all work shifts of every weekday of that area.

+ Work shifts can be changed (still guaranteed during prime time) based on MCPs data from previous weeks (to optimize efficiency for the most waste concentrated time only).

– Back officers will assign available collectors to shifts (one collector each). This assignment only notifies the collector his shift.

– The route will be optimized by the system, then it will be updated and sent to the collector along with a notification before his/ her shift.

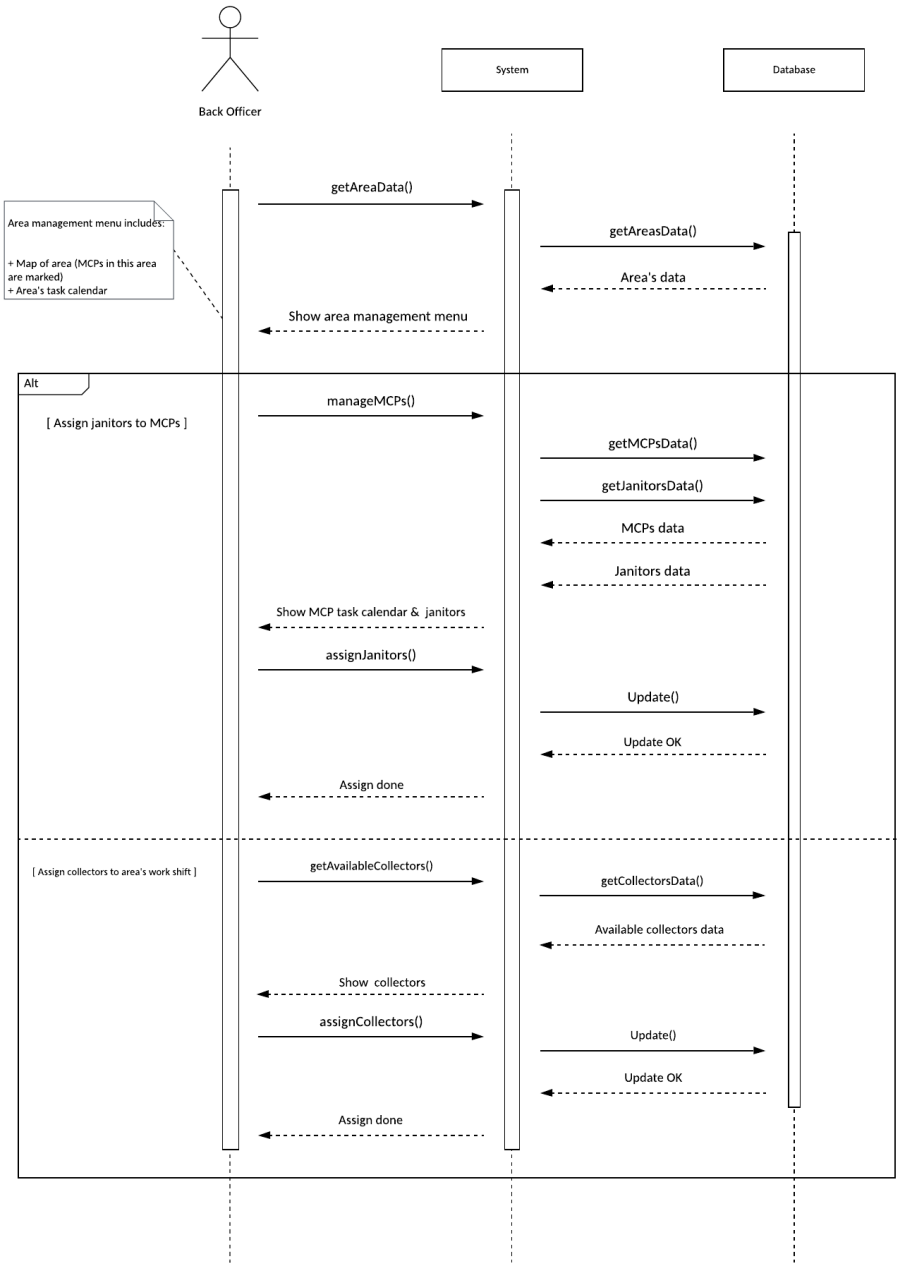
– The MCPs’ criteria to be selected for one route will be based on:

+ The MCPs that reach the threshold (real-time data).

+ The MCPs that is likely to reach the threshold soon (The prediction will be based on the previous weeks data).

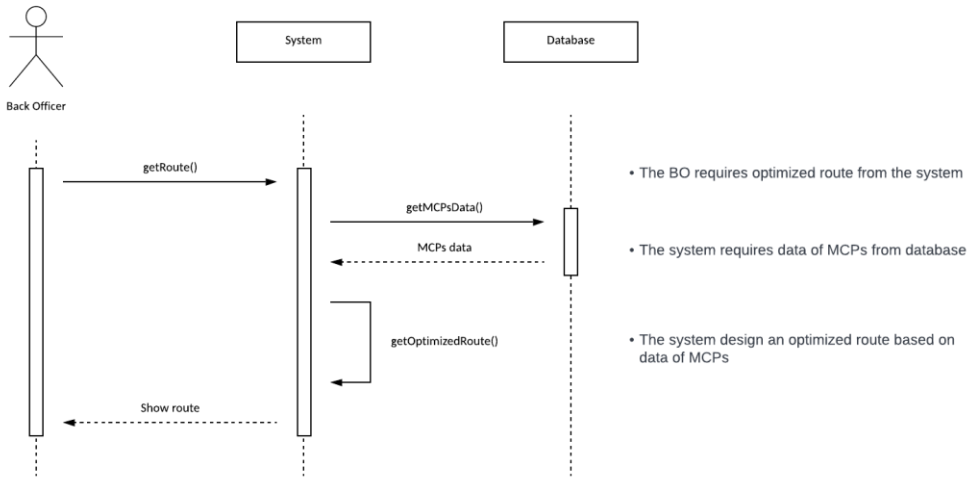
2.2.2. Sequence diagram

Task assignment:

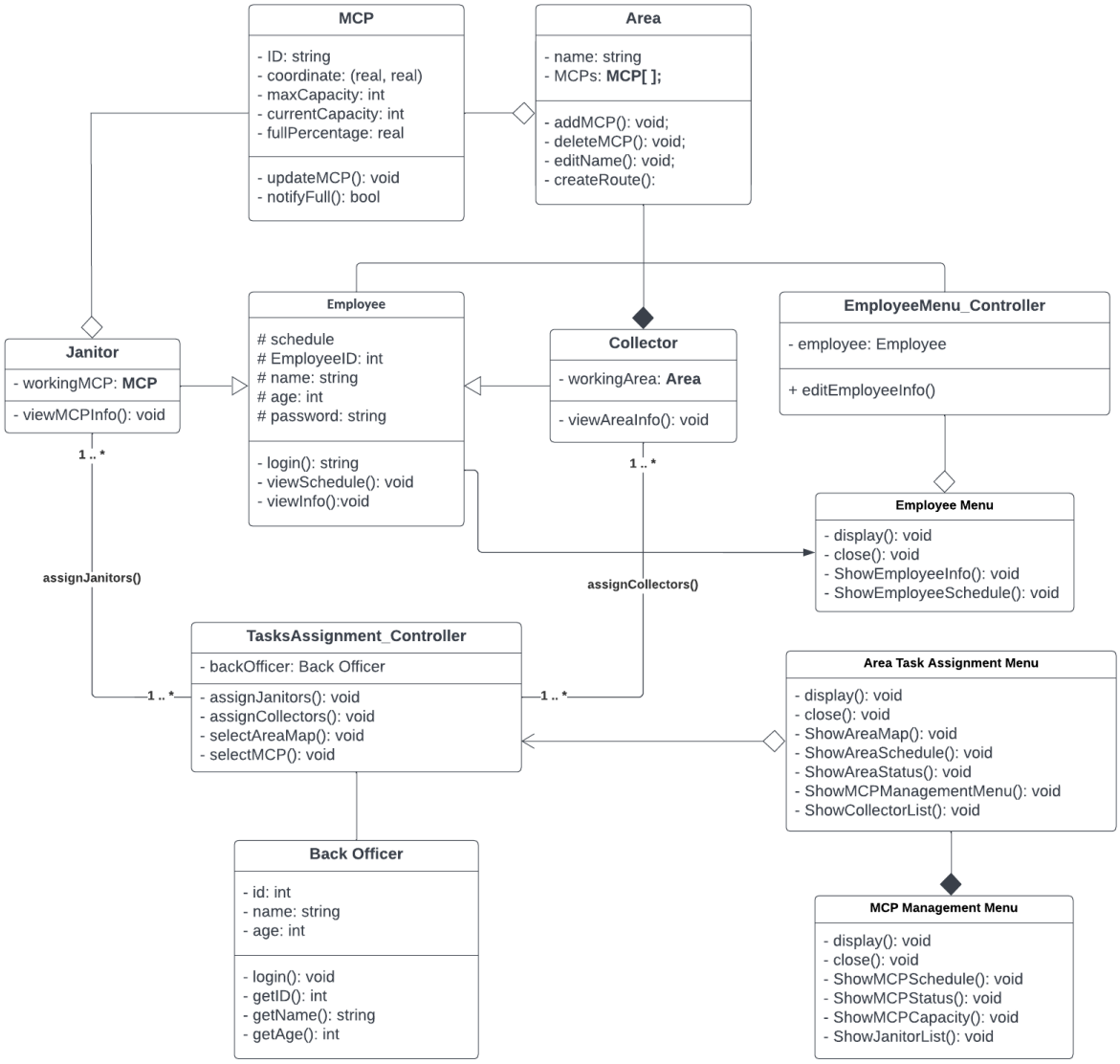


- The BO start the assignment by choosing the needed area
- In case the user clicks on an MCP (from the MCPs map), the system will show that MCP's task calendar and the list of available janitors
- The BO assign janitors to MCP's work shift (maybe by dragging and dropping the janitors in the given list into the work shift of MCP)
- In case, the user clicks on the area's task calendar below, the system will show the list of collectors.
- The BO assign collectors to area's work shift (maybe by dragging and dropping the collectors in the given list into the work shift of MCP)

Route planning:



2.3. Class diagram for Task Assignment Module



TASK 3

3.1. Architectural approach

3.1.1. Describe an architectural approach you will use to implement the system.

In the process of applying the design system, our team choose the MVC model which is an architectural pattern that separates an application into three main logical components such as Model, View, and Controller. Each architecture component is built to handle specific development aspect of an application, which is responsible for adding source code into 3 main parts, each part has its own task and is handled independently of the other parts of the software.

– Model (M): the component that stores all the data of the application and its related logic, is the bridge between the two components View and Controller. It responds to the request from the views and also responds to instructions from the controller to update itself. The model can be displayed as a database, it explicitly represents operations with the database such as viewing, retrieving, processing data, etc.

In our application, model is used to store information about janitor, collector, list of information of MCP, Vehicles, Routes, … Furthermore, it is also a method to call API, communicate with database to perform basic operations functions such as CREATE, READ, UPDATE, DELETE (CRUD).

– View (V): the component whose functionality is to manage the data presented to the user. A view requests directly to the model to give information or indirectly through the Controller so that it presents the output presentation to the user.

Our system can utilize this component for: login, registration, view tasks, view information, ... More details are displayed registration forms, login, buttons with their own functions such as submit user request or redirect, …

– Controller (C): component that handles user actions for the purpose of notifying the View as well as the Model if there is a change request.

Our application uses Controller to handle performing actions on the data through user interaction, returning the corresponding view from the request made.

Interaction flow:

User sends request -> Controller receives request -> Handle model if needed -> Return View result -> Display interface to user -> Wait for next request.

Our application works with a good number of features that require interactions between views and data so separates it into view, controller and model made it easier to implement and also further utilize the extensibility of the architectural approach (for any additional requirements in the model component do not affect what we defined in the view component).

The downside when it comes to this architectural pattern is for its difficulty to ensure the Model-View-Controller interaction which makes the implementation more complicated.

3.1.2. How many modules do you plan for the whole WMC 2.0 system? Briefly describe input, output and function of each module.

Module	Area Management
Input	name: string MCPs: MCP[ ] collectorID: string schedule: {name: string, day: enum Weekdays, workshift: int, collectorID} []
Output	newArea: Area existingArea: Area
Function	createArea(name, MCPs): newArea getAreaByName(name): existingArea updateAreaName(id, name): void updateAreaMCPs(name, MCPs): void deleteAreaByID(name): void updateAreaSchedule(schedule): void

Module	MCP Management
Input	coordinate: {x: float, y: float} maxCapacity: int currentCapacity: int ID: string janitorID: string schedule: {name, day, workshift, janitorID} []
Output	newMCP: MCP existingMCP: MCP
Function	createMCP(coordinate, maxCapacity, currentCapacity): newArea getMCPByID(ID): existingMCP

	updateMCPMaxCapacity(ID, newMaxCapacity): void updateMCPCurrentCapacity(ID, newCurrentCapacity): void updateMCPSchedule(schedule): void deleteMCPByID(ID): void

Module	Communicate

Input	backOfficerID: string
	employeeID: string
	id: string
	message: {
	content: string
	sender: “Back Officer” | “Employee”
	}

Output	conversation: Conversation
Function	createConversation(backOfficerID, EmployeeID): conversation getConversation(id): conversation addMessage(id, message): void

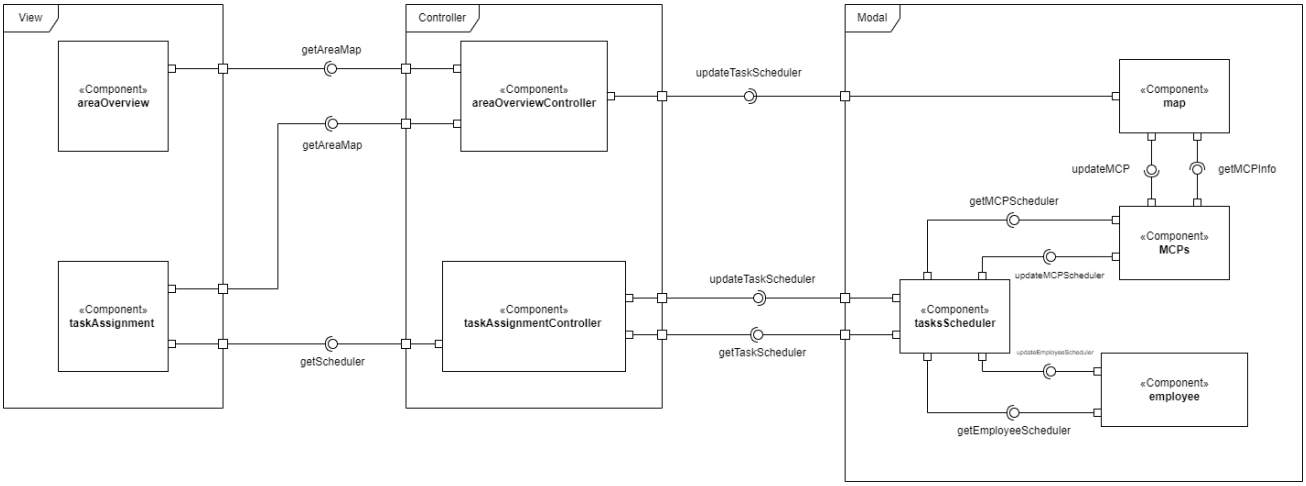
Module	Vehicle
Input	name: string capacity: float fuel: float image: string collectorID: string
Output	vehicle: Vehicle
Function	createVehicle(name, capacity, fuel, collectorID): vehicle getVehicle(id): vehicle deleteVehicle(id): void updateVehicle(id, name, capacity, fuel): void updateCollector(id, collectorID): void

Module	Employee Services
Input	name: string age: int role: “collector” | “janitor” password: string oldPassword: string newPassword: string id: string
Output	employee: Employee collectorSchedule: {area, day, workshift} [] janitorSchedule: {MCP, day, workshift} []
Function	login(name, id, password): collector getEmployeeInfo(id): employee updateInfo(id, name=””, age=0): void updatePassword(id, oldPassword, newPassword): void getOwnSchedule(id, role): collectorSchedule | janitorSchedule

Module	Back Officer
Input	name: string age: int password: string id: string janitorID: int collectorID: int employeeID: int
Output	existingBackOfficer: Back Officer collector: Collector janitor: Janitor areaSchedule: {name, day, workshift, collectorID} [] MCPSchedule: {name, day, workshift, collectorID} []+
Function	login(id, password): existingBackOfficer updateInfo(id, name =””, age = 0): void createEmployee(name, age, role): collector | janitor updateEmployeeInfo(id: collectorID | janitorID , name=””, age=0): void deleteEmployee(id): void sendMessage(employeeID: string, message: string)

Module	Task Assignment
Input	employeeID: int
Output	areaSchedule: {name, day, workshift, employeeID} [] MCPSchedule: {name, day, workshift, employeeID} []
Function	getAreaList(name): Area [] getMCPList(id): MCP [] getCollectorList(): Employee[] getJanitorList(): Employee[] assignJanitorToMCP(name, employeeID): void assignCollectorToArea(name, employeeID): void

3.2. Implementation Diagram for Task Assignment module



Need more explanation!!

TASK 4

4.1. Setting up

The team creates an online repository for version control.

Our GitHub repository: https://github.com/khoatruong19/UWC2.0

4.2. Adding documents, materials and folders

Using the selected version control system to report the changes to the files.

Due to the excessive amount of commits, we will show only a decent amount of commits that actually made important changes to our application. For further information, you can look into our repository in the link above.

```
commit 1b1530a962fccf86b6a05365c1479e45eb3fcb06
Author: khoatruong19 <thpthv.truonghuynhdangkhoa@gmail.com>
Date:   Tue Nov 15 11:59:04 2022 +0700

    refactoring

commit af9badbf2eafc0978e157d41e2e294dc04a1a8c4
Author: Chasu Hak <72014568+hieptth@users.noreply.github.com>
Date:   Tue Nov 15 11:51:10 2022 +0700

    Delete note.txt

commit cc35dca602e04573eb6c5397d25fa440f9d57e7a
Author: Chasu Hak <72014568+hieptth@users.noreply.github.com>
Date:   Tue Nov 15 11:50:54 2022 +0700

    Add files via upload

commit f9c86f7cd1e3a477234118e8c02f4196f99c368f
Author: Chasu Hak <72014568+hieptth@users.noreply.github.com>
Date:   Tue Nov 15 11:50:04 2022 +0700

    Create note.txt

commit f55d5a96b180ec8ca94a017f395027bcb258c9e3
Author: Chasu Hak <72014568+hieptth@users.noreply.github.com>
Date:   Tue Nov 15 11:46:45 2022 +0700

    Create README.md

commit 060620a58655ca043b319b80003cd16d793419d3
Author: khoatruong19 <thpthv.truonghuynhdangkhoa@gmail.com>
Date:   Mon Nov 14 17:12:07 2022 +0700

    set up and done sidebar
```

```
commit e428242ae650496864643a683b2bbdb075814439
Author: Chasu Hak <72014568+hieptth@users.noreply.github.com>
Date:   Mon Nov 21 19:08:56 2022 +0700

    Create note.txt

commit f27a4f980c9bd9108358b09b10af1e56a73d48ce
Author: nqhuy137 <nquochuy137@gmail.com>
Date:   Sun Nov 20 13:41:27 2022 +0700

    Reinstalled package.json

commit e1024fcfaafb732f6ddf8ee450f24d05f9a70e0f
Merge: 65e245a 425aa24
Author: khoatruong19 <thpthv.truonghuynhdangkhoa@gmail.com>
Date:   Sun Nov 20 13:33:07 2022 +0700

    Merge branch 'main' of https://github.com/khoatruong19/UWC2.0

commit 65e245a94145d056b6156a08f483b32508960fd7
Author: khoatruong19 <thpthv.truonghuynhdangkhoa@gmail.com>
Date:   Sun Nov 20 13:32:35 2022 +0700

    setup map

commit 425aa24f368bc9a0f779462fcaea6b50f9018c4c
Author: Chasu Hak <72014568+hieptth@users.noreply.github.com>
Date:   Tue Nov 15 16:21:47 2022 +0700

    Add files via upload

    Update Task 4
```

```
commit 50c67e158ccfa9c3b6c1de76038dd9cb00c3e74d
Merge: 03c01b7 2238298
Author: khoatruong19 <thpthv.truonghuynhdangkhoa@gmail.com>
Date:   Wed Nov 23 21:38:54 2022 +0700

    Merge branch 'main' of https://github.com/khoatruong19/UWC2.0

commit 03c01b77a91f8495a52d48eadd5937048c010a7c
Author: khoatruong19 <thpthv.truonghuynhdangkhoa@gmail.com>
Date:   Wed Nov 23 21:35:15 2022 +0700

    fixed map bug

commit 223829889a9da16a449104d0b64646bb755194e4
Merge: 8334dda 1d361ac
Author: hieptth <hieptth71@gmail.com>
Date:   Wed Nov 23 20:25:15 2022 +0700

    Merge branch 'main' of https://github.com/khoatruong19/UWC2.0

commit 8334dda0b8e164f9a19f63756548f86569dd2add
Author: hieptth <hieptth71@gmail.com>
Date:   Wed Nov 23 20:24:58 2022 +0700

    add ChooseRoute.js
```

```
commit 881721d26c89e764465cebcae64e51b2f2d4284f
Author: nqhuy137 <nquochuy137@gmail.com>
Date:   Fri Nov 25 19:33:24 2022 +0700

    deleted unused MCPInfoSidebar draft

commit b85facb1e3b133cc37ed5d6ee8917514281a5033
Author: nqhuy137 <nquochuy137@gmail.com>
Date:   Fri Nov 25 19:29:28 2022 +0700

    Added janitor assignmen popup

commit 668e418874c63f2735411de7ac78582faf37f9aa
Merge: e762c9d 5f84770
Author: nqhuy137 <nquochuy137@gmail.com>
Date:   Fri Nov 25 19:09:08 2022 +0700

    commit after pull

commit e762c9d7f6f78faf73f2ae2be789b609659993ce
Author: nqhuy137 <nquochuy137@gmail.com>
Date:   Fri Nov 25 18:53:42 2022 +0700

    commit before pull

commit 7a22b37ac395b8f2b51b7aed459ced6563f4db9d
Author: nqhuy137 <nquochuy137@gmail.com>
Date:   Fri Nov 25 18:14:59 2022 +0700

    Added MCP modal popup when clicking MCP icon
```

```
commit 2e5f371a6eaf421a7d5a76dc7e74c3988d87672a
Author: nqhuy137 <nquochuy137@gmail.com>
Date:   Sat Nov 26 19:52:15 2022 +0700

    added MCP info modal

commit b4189e2e4ae76941b290b2989b0d152ea34d7c62
Author: blackcat22521 <blackcat22521@gmail.com>
Date:   Sat Nov 26 16:37:29 2022 +0700

    updateLoc

    component assignCollector

commit 973eb76ed69d9511843261b9f5fafb97462e42be
Author: hieptth <hieptth71@gmail.com>
Date:   Fri Nov 25 22:43:23 2022 +0700

    add ChooseRouteModal

commit c49572cbe109766d463e685ad16f5d7306b2c5e0
Author: hieptth <hieptth71@gmail.com>
Date:   Fri Nov 25 20:52:20 2022 +0700

    clean-up

commit 71c42137be6c59d5d2b21f017402894f54e7aeed
Author: hieptth <hieptth71@gmail.com>
Date:   Fri Nov 25 20:46:26 2022 +0700

    add react-icons
```

```
commit 37461b8d34b47da1607a586a3303ecd7a27438a0
Author: nqhuy137 <nquochuy137@gmail.com>
Date:   Tue Nov 29 19:54:44 2022 +0700

    implemented Calendar.js

commit 92f96a3a17bafbf423057b20ed6dbecb801a2b90
Author: nqhuy137 <nquochuy137@gmail.com>
Date:   Tue Nov 29 19:53:55 2022 +0700

    deleted unused files

commit 3c1ccee8e967a4f5d46c6c3f8a1f530c29b7e73b
Author: hieptth <hieptth71@gmail.com>
Date:   Sat Nov 26 20:18:47 2022 +0700

    update ChooseRouteModal
```

```
commit e56d688fb68703347cdbd3c77feb6040e023c466
Merge: 7d8c4df 546e873
Author: blackcat22521 <blackcat22521@gmail.com>
Date:   Thu Dec 1 17:59:19 2022 +0700

    add collector table

commit 7d8c4dfe4872b2c3869b94a0bd78b851a8023b36
Author: blackcat22521 <blackcat22521@gmail.com>
Date:   Thu Dec 1 17:50:39 2022 +0700

    add collector assign table

commit 546e87314bb40dd4756b80e8f1842af29a3c5246
Author: hieptth <hieptth71@gmail.com>
Date:   Thu Dec 1 11:23:24 2022 +0700

    add feature to ChooseRouteModal

commit a05bf55ae073261f7eaa364f08493bd28661c4bc
Author: Chasu Hak <72014568+hieptth@users.noreply.github.com>
Date:   Thu Dec 1 08:02:17 2022 +0700

    Delete node_modules directory

commit ae378a818d4e37fe9be211a7349709892db71c4a
Author: Chasu Hak <72014568+hieptth@users.noreply.github.com>
Date:   Thu Dec 1 08:01:34 2022 +0700

    Update .gitignore

commit f3ad1a1b1ba15f387afa10401f031d39b10a58c8
Author: hieptth <hieptth71@gmail.com>
Date:   Thu Dec 1 08:00:18 2022 +0700

    fix local
```

```
commit f6f0c7666ff51b042603718487b280f57e34390d
Author: khoatruong19 <thpthv.truonghuynhdangkhoa@gmail.com>
Date:   Sat Dec 3 20:28:45 2022 +0700

    map done

commit c6a632d40693f864dcf59d394a65b59739086679
Author: em-dzu-github <vu.nguyen1472@hcmut.edu.vn>
Date:   Sat Dec 3 11:33:39 2022 +0700

    Update Task assignment page

commit 4b925cffd293d28f562063bb4538839efb18b8b1
Author: nqhuy137 <nquochuy137@gmail.com>
Date:   Fri Dec 2 20:33:06 2022 +0700

    fixed modal

commit 6f04c925ca11058d30a3dcb408391f3bc9f221e5
Author: nqhuy137 <nquochuy137@gmail.com>
Date:   Fri Dec 2 20:32:48 2022 +0700

    erase "Workers" info

commit 5f9082d69c46d32641cbb1e8992885f5a1a1b045
Author: nqhuy137 <nquochuy137@gmail.com>
Date:   Fri Dec 2 20:31:56 2022 +0700

    validate submit on "new shift" field

commit 827350d4b5056d941ec87ad2b353717adf709030
Author: nqhuy137 <nquochuy137@gmail.com>
Date:   Fri Dec 2 19:44:33 2022 +0700

    integrated collector table
```

```
commit 4599df21dae7221133f2761210fa2aa0dfeb39e6
Author: nqhuy137 <nquochuy137@gmail.com>
Date:   Mon Dec 5 00:24:55 2022 +0700

    adjust style of EmployeeList page

commit ef52185802c194d54be18ec6c2cd444ec9e3e47c
Author: nqhuy137 <nquochuy137@gmail.com>
Date:   Sun Dec 4 23:10:24 2022 +0700

    fixed duplicate

commit a11614a43570df2c921785c67ef1fdfd99a71a8a
Merge: a8077a5 da2be00
Author: nqhuy137 <nquochuy137@gmail.com>
Date:   Sun Dec 4 23:01:53 2022 +0700

    Merge branch 'main' of github.com:khoatruong19/UWC2.0

commit a8077a57bc3f019ce518ca98512b00811af22a7c
Author: nqhuy137 <nquochuy137@gmail.com>
Date:   Sun Dec 4 23:00:03 2022 +0700

    added EmployeeList page

commit 8b767e1435dd3c9731b74056c72a8e6c738fbe8d
Author: nqhuy137 <nquochuy137@gmail.com>
Date:   Sun Dec 4 22:59:45 2022 +0700

    added AreaMap
```

```
commit a0e603b128106fae956eae8b60b6d36fb7e84e79
Author: Loc <blackcat22521@gmail.com>
Date:   Mon Dec 5 19:01:41 2022 +0700

    add janitor table

commit 53355e5c388a79881a4d1563e0b831844644448f
Author: hieptth <hieptth71@gmail.com>
Date:   Mon Dec 5 17:20:10 2022 +0700

    minor change

commit 1d1aad3d610999f8a2e38e3c8f963a382ff5ceb4
Author: Loc <blackcat22521@gmail.com>
Date:   Mon Dec 5 17:09:34 2022 +0700

    fix table

commit 1d638885f9727bb9aeab210c305bfdf0361b95b5
Merge: 91d8482 471942b
Author: hieptth <hieptth71@gmail.com>
Date:   Mon Dec 5 13:53:24 2022 +0700

    Merge branch 'main' of https://github.com/khoatruong19/UWC2.0

commit 91d84826f2dd0a82f0f01f2eb58604536d47cd44
Author: hieptth <hieptth71@gmail.com>
Date:   Mon Dec 5 13:51:56 2022 +0700

    change color scheme ChooseRoute

commit 471942baccd6602824bb37013748de9ed94f5cba
Author: Chasu Hak <72014568+hieptth@users.noreply.github.com>
Date:   Mon Dec 5 13:51:15 2022 +0700

    Update note.txt
```

```
commit 39127a246760d93b7fbfaa65e87d1e1d90c29452
Author: Loc <blackcat22521@gmail.com>
Date:   Mon Dec 5 19:53:01 2022 +0700

    color

commit 299e66f30e8ed91f83050f07635a1c4d8dc282bf
Author: nqhuy137 <nquochuy137@gmail.com>
Date:   Mon Dec 5 19:27:39 2022 +0700

    changed text color in EmployeeList

commit 921185a95277655c05df989d38c02a8a989eed77
Merge: cb7b50d bffa4eb
Author: hieptth <hieptth71@gmail.com>
Date:   Mon Dec 5 19:24:47 2022 +0700

    Merge branch 'main' of https://github.com/khoatruong19/UWC2.0

commit bffa4ebc66c977b54f80c25843c748078662b3a7
Author: nqhuy137 <nquochuy137@gmail.com>
Date:   Mon Dec 5 19:23:52 2022 +0700

    rebased and rename
```

```
commit c39b599df88bdd15344c82b6be3314ea4c30a53c (HEAD -> main, origin/main, origin/HEAD)
Author: Chasu Hak <72014568+hieptth@users.noreply.github.com>
Date:   Tue Dec 6 11:40:42 2022 +0700

    Finishing touch

commit a0767a2f78d38f5adc326f673df9b967247e07b2
Author: khoatruong19 <thpthv.truonghuynhdangkhoa@gmail.com>
Date:   Mon Dec 5 23:35:47 2022 +0700

    login done
```

4.3. Implement MVP1

Design an interface of either a Desktop-view central dashboard for Task Management for back-officers.

Click Here to access our interactive Figma implementation or browse through the following images.

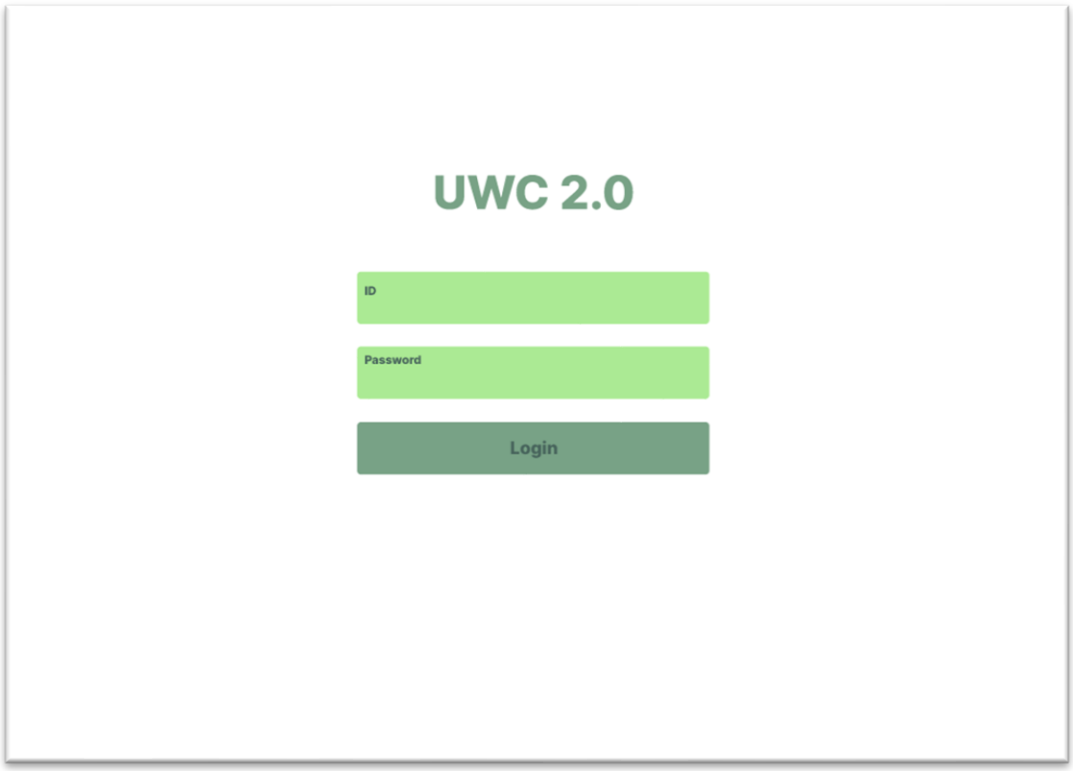


Figure 1. Login page

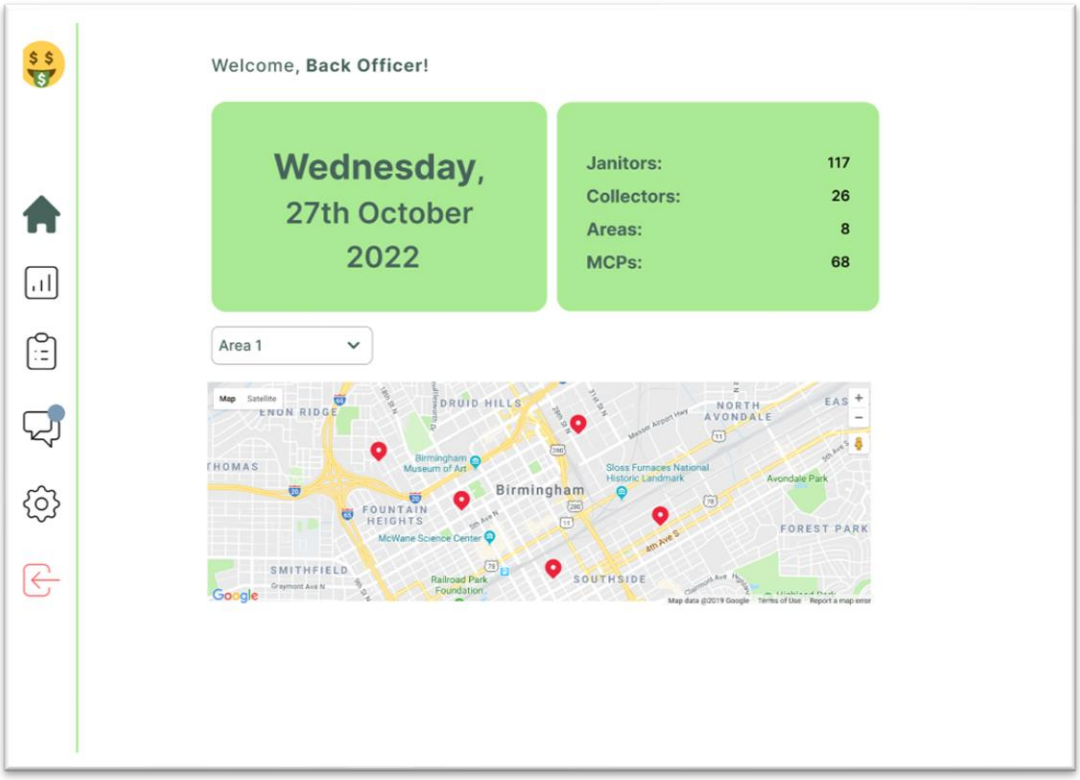


Figure 2. Main menu

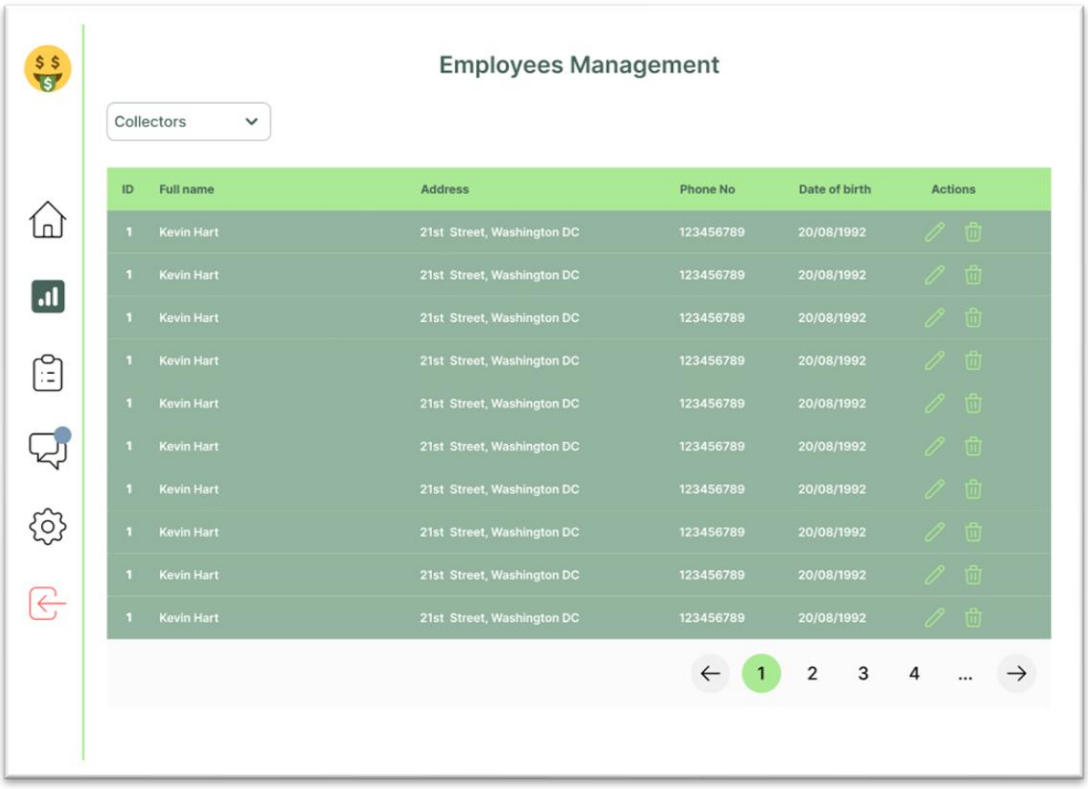


Figure 3. Employee Management Menu

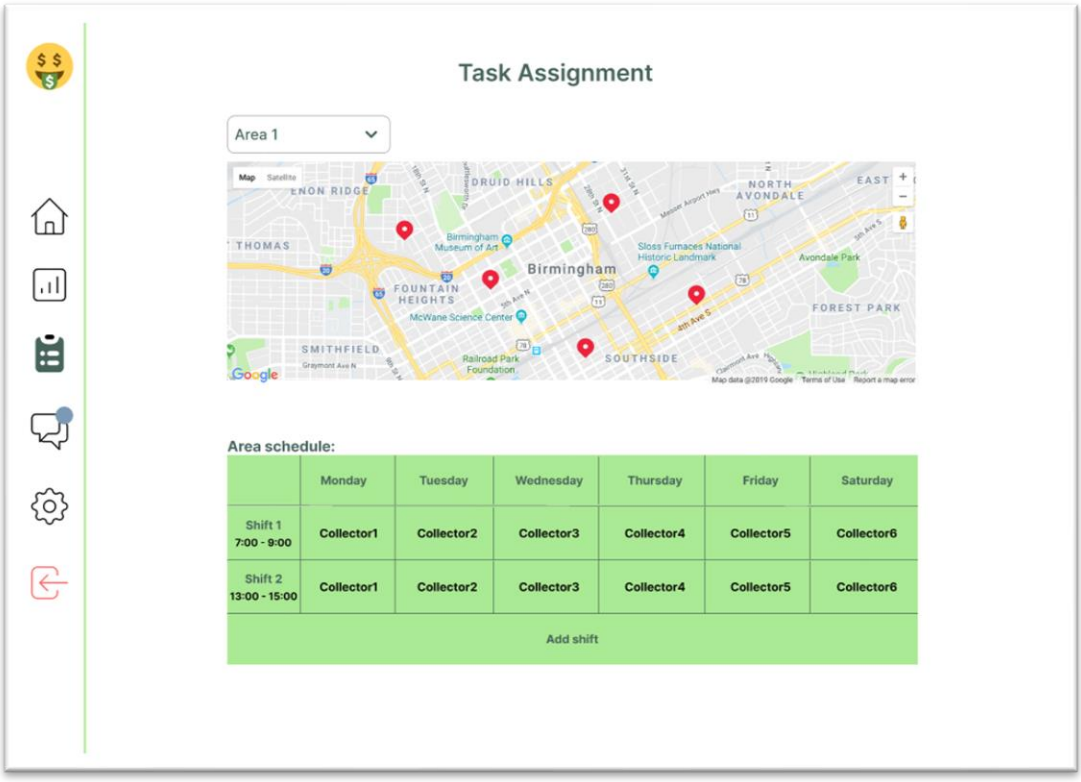


Figure 4. Task Assignment Menu

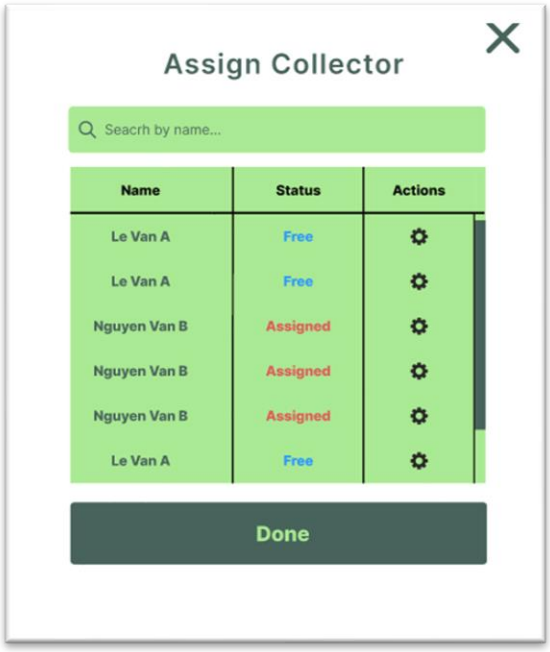


Figure 5. Assign Collector Overlay

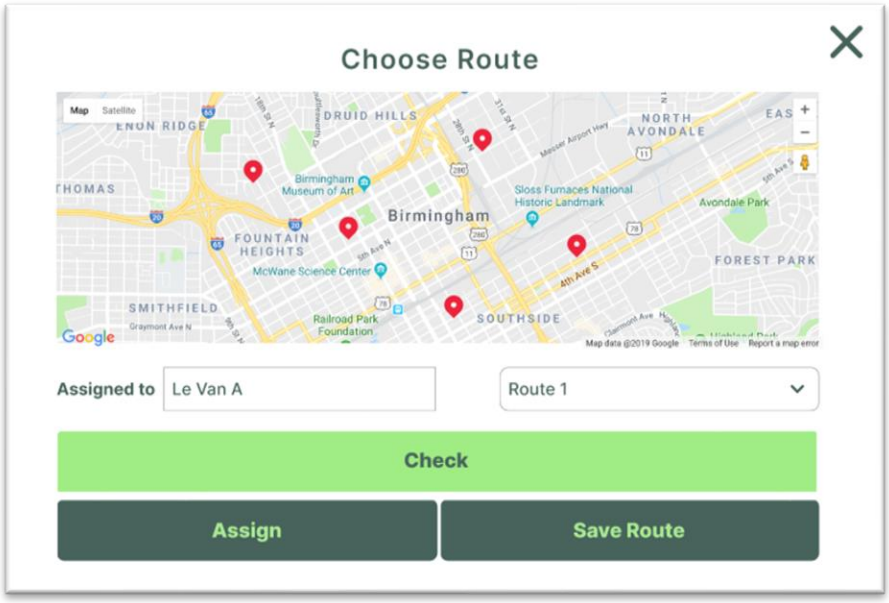


Figure 6. Choose Route Overlay

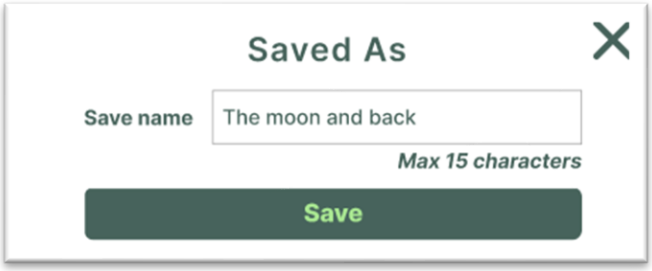


Figure 7. Save Route Overlay

Figure 8. MCP Info Overlay

Figure 9. MCP Schedule Overlay

Figure 10. Assign Janitor Overlay

TASK 5

Note that after pulling the code from our Github repository, you may want to run “npm install” if you don’t have the prerequisite packages available. Then, change the directory to the client folder before running “npm start” to start our web application.

We will be redirected to the following login page.

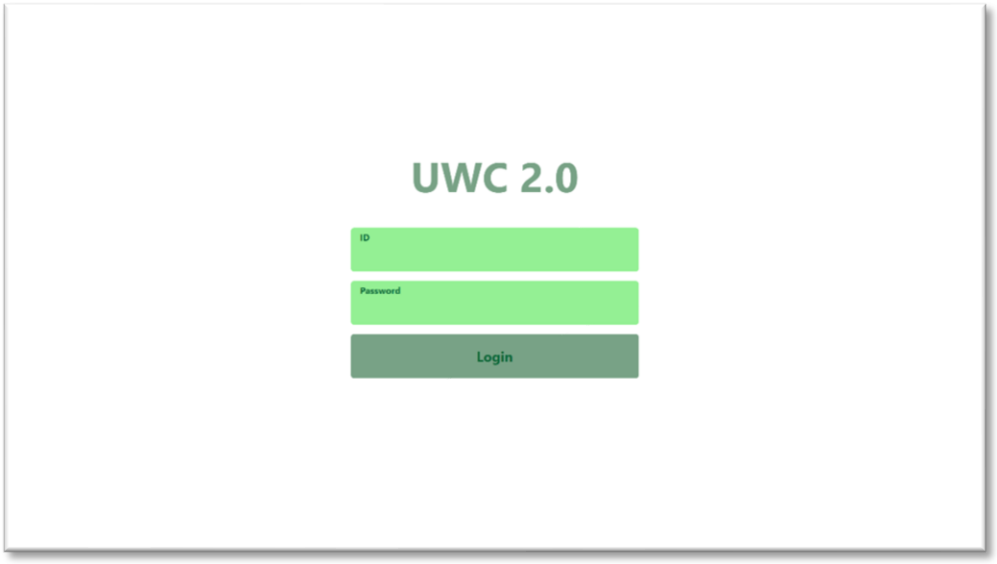


Figure 11. Login page.

After logging in, our default destination will be the Home page. In the Home page, there are information about the Current Date, the number of Janitors, Collectors, Areas, MCPs and an integrated map.

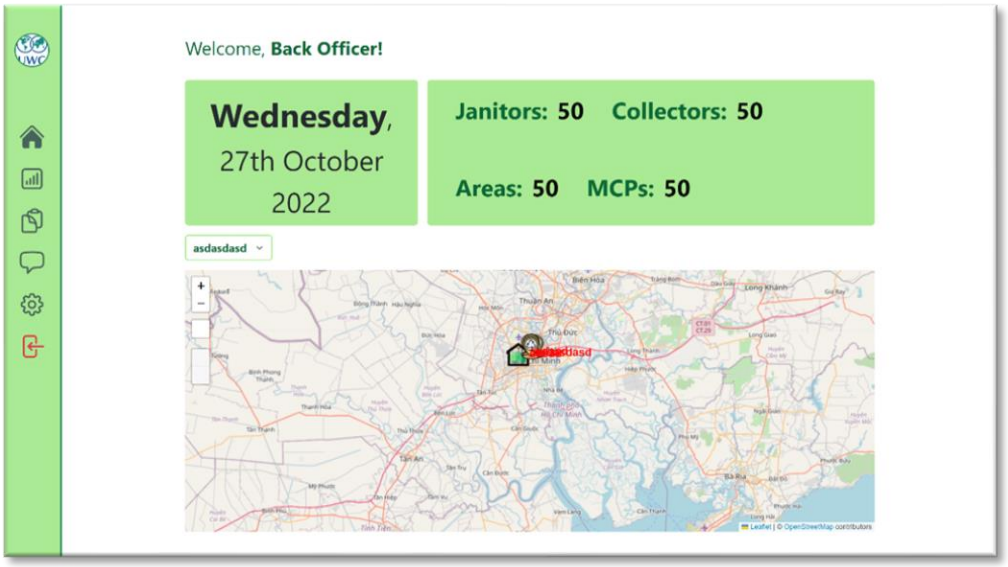


Figure 12. Home page.

From the Home page, we can access to the Employee List, Task assignment page, Chat system, Setting configuration and Logout. But this task only focus on Task assignment so we won't go into other details.

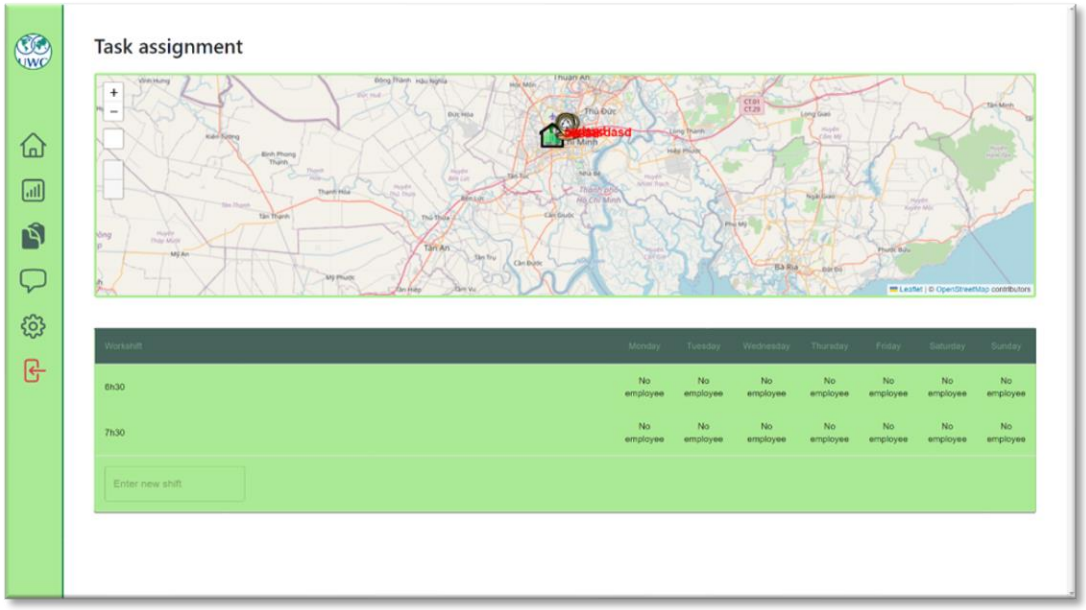


Figure 13. Task assignment page.

From this point on, there will be two scenarios where we want to assign janitors to any MCPs or assign collectors to any workshift of the current week.

First, we will go into details the process of assigning a collector to any workshift.

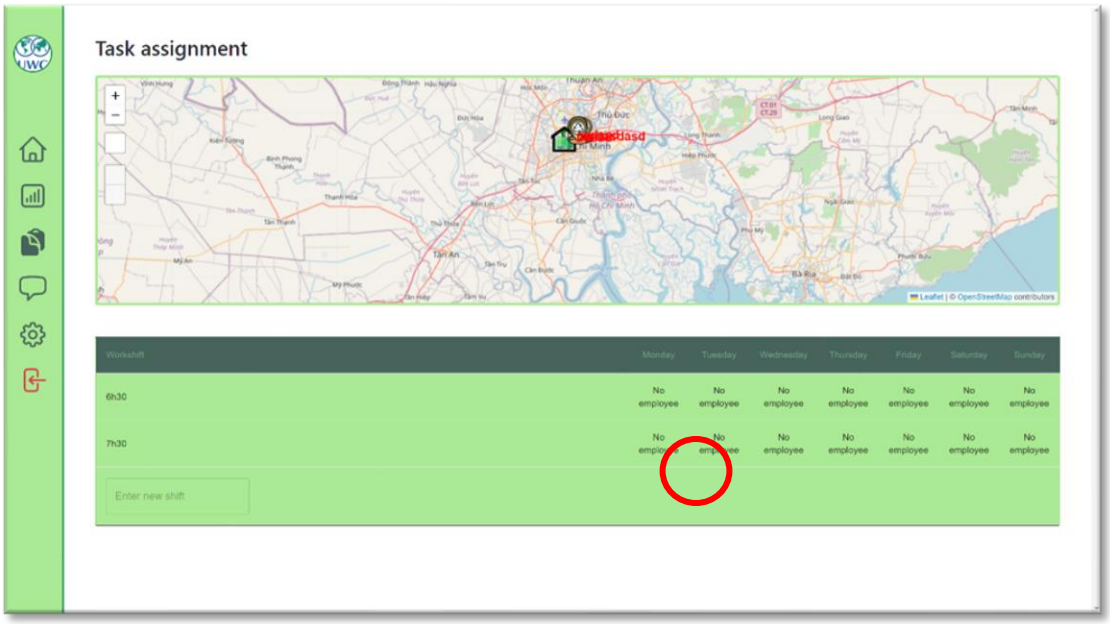


Figure 14. Choose any workshift.

After picking a workshift, the Assign Collector overlay with pop up for us to choose a collector from a Collector List with his/ her brief information such as Name, Status and Action (we will go into this shortly).

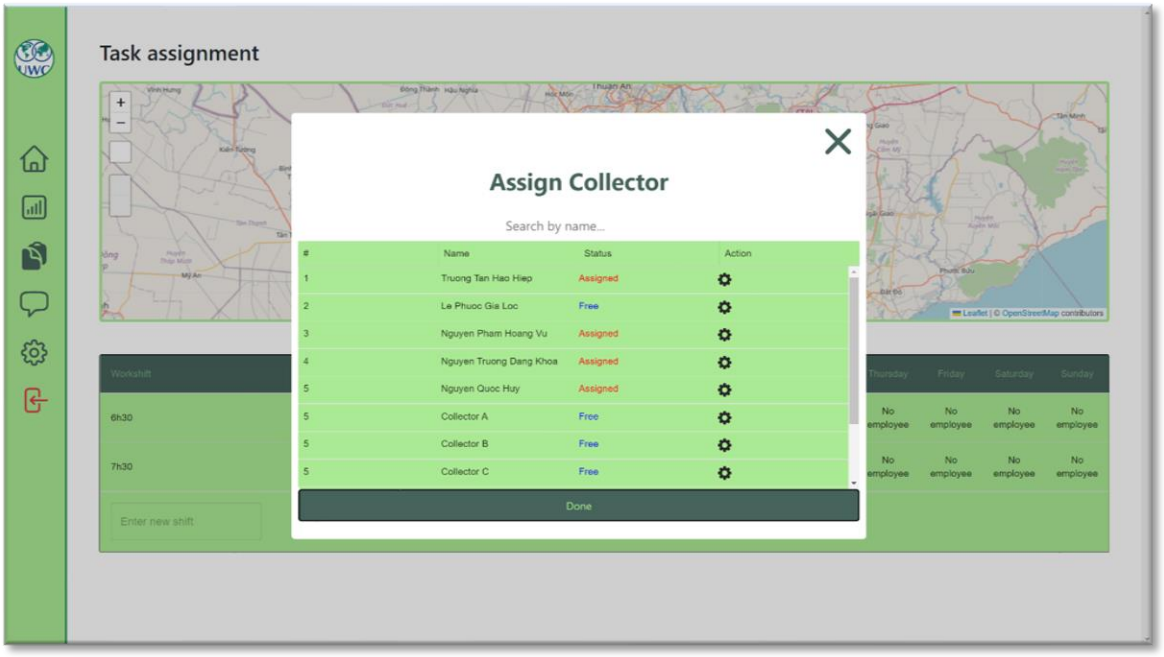


Figure 15. Assign Collector overlay.

If the collector haven't been assigned a route yet, we can click on the Action icon to manually assign a route for the chosen collector in the Choose Route overlay.

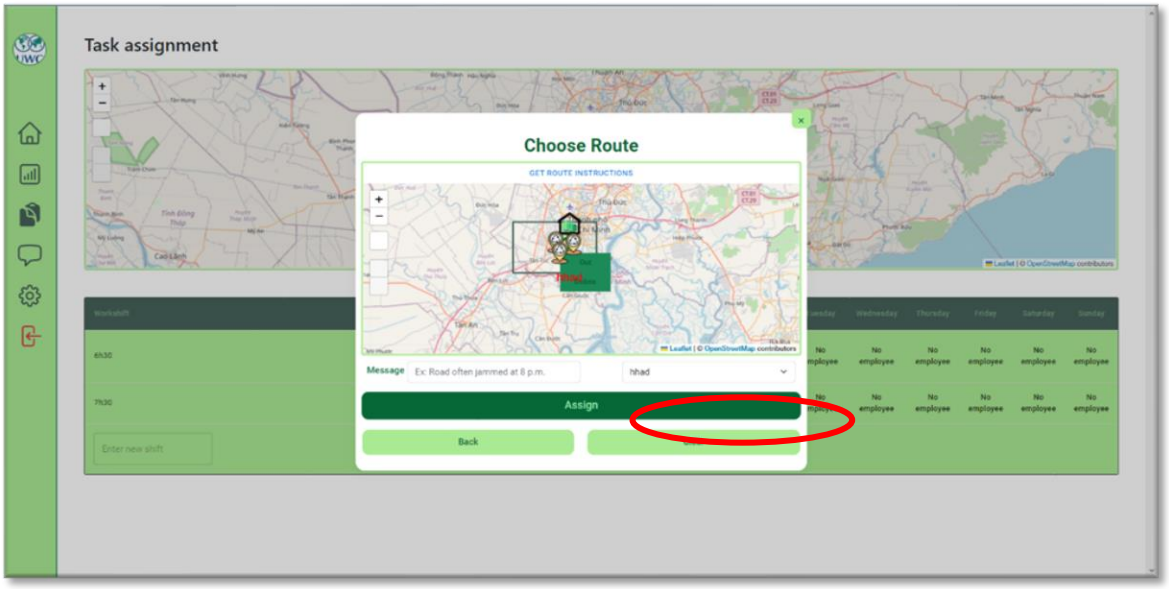


Figure 16. Choose Route overlay.

In the Choose Route overlay, we will only focus on the chosen Area (can be changed using the drop down highlighted in red). There are various operation to begin with; right click on

any MCP and there will be an option to opt out it in our current route, hit the clear button to revert the map back to its original state and the Get route instructions will show us step to step instruction for the chosen route. After completing the operation, hit the Assign button to assign the optimized route to the collector.

Alternatively, in the Task assignment page, we can click on any MCP in the provided map to access the MCP Info overlay to start assigning a janitor to an MCP.

Figure 17. MCP Info overlay.

After picking a workshift, the Assign Janitor overlay with pop up for us to choose a janitor from a Janitor List with his/ her brief information such as Name, Status and Action.

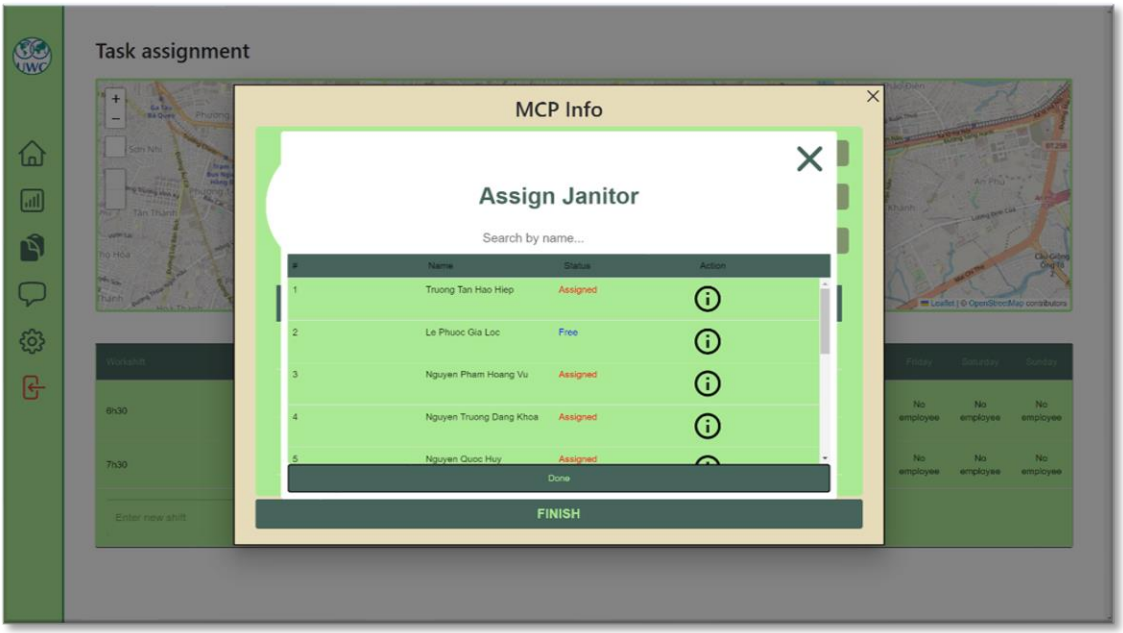


Figure 18. Assign Janitor overlay.

Picking a janitor will assign him/ her to the chosen workshift. In addition, we can check the collector personal information by clicking on the Action button which will redirect us to the Employee List.

Figure 19. Employee List page.

END OF REPORT!