URBAN WASTE COLLECTION AID UWC 2.0

ASSOC BUI HOAI THANG

LÊ NHẬT ĐĂNG	2052950
TRẦN KHOA	2052541
TRẦN CÔNG KHÔI	2052138
ĐINH TRỰC TÂM	2053415
LÊ PHAN KỲ TÀI	2053411

Contents

Member list and contribution

No	Full name	Student ID	Contribution	Signature
1	Lê Nhật Đăng	2052950	100%	[illegible] Lê Nhật Đăng
2	Trần Khoa	2052541	100%	Khoa
3	Trần Công Khôi	2052138	100%	*khoi*
4	Đinh Trực Tâm	2053415	100%	Tam
5	Lê Phan Kỳ Tài	2053411	100%	[illegible] Lê Phan Kỳ Tài

1. Task 1: Requirement elicitation

1.1 Domain context of urban waste management in Vietnam.

Waste Management in urban areas has been a hot topic since we experienced the first industrial revolution. Nowadays, identifying, managing and resolving urban solid waste is considered one of the most important missions along with concurrently developing the industry. In 2015, Vietnam was in the list of top five countries that dump more plastic into the oceans than the rest of the world combined. Whereas most plastic waste comes from cities and societies, they were thrown at seas due to lack of management and treatment on land. According to Global Recycling, the five biggest cities in Vietnam including Hanoi, Ho Chi Minh, Hai Phong, Da Nang and Can Tho are responsible for about 70 percent of the total waste generation. A statistics conducted by the Netherlands' Ministry of Foreign Affairs, CREM and Partners for Innovation in 2018 also pointed out that the annual Vietnamese waste production comprised more than 27.8 million tons consisting of 46 percent from municipal sources, the remainder delivered from agriculture and industry. Though, due to lack of modern technology, the method to treat those wastes is still very challenging.

In 2021, according to the statistics conducted by the Department of Natural Resources and Environment of Ho Chi Minh City, the city generated about 9400 tons of domestic waste(not including industrial waste) and that number is still increasing about 10% each year. Currently, according to The Ministry of Finance, the city generates over 4000 tons of solid industrial waste and over 400 tons of hazardous waste including 45 tons of medical waste. The majority of domestic waste is treated by landfill technology(60% of total waste) whereas the rest is treated by burning, fertilizer producing and recycling. There are 4 main steps of treating waste in Ho Chi Minh City. Firstly, domestic waste after being discharged will be collected by local collecting waste companies. Then, garbage is brought to MCPs(Major Collecting Points) by janitors and collectors. Thirdly, the Urban Environment Company of Ho Chi Minh City will collect all the waste and transfer it to the city's garbage treatment areas. Finally, waste will be classified and treated with the above methods. Hence, a system called UWC 1.0 is released which aims to help those companies to control the waste treatment process easier, more convenient and faster. Nevertheless, the first version of UWC 1.0 is not that helpful due to small-scale models, technical errors and old databases.

Realizing the difficulties in managing and treating urban waste, an organization X is contracted to develop an information management system called UWC 2.0 in order to improve efficiency of garbage collection of Service provider Y. The stakeholders consist of the organization X, back officers, collectors and janitors. Back officers will be in charge of operating a central system to create a calendar and coordinate for collectors and janitors. Collectors will drive different types of vehicles from mini to large trucks while janitors will manually collect garbage from Major Collecting Points.

Back officers will have to have an overview of janitors and collectors as well as their work calendar. Moreover, an overview of vehicles and their technical details as well as an overview of all MCPs should be controlled and taken care of by back officers. They also have to assign vehicles to janitors and collectors(trollers for janitors, trucks for collectors) and assign janitors and collectors to MCPs. As a manager, they have to be able to keep in touch with janitors and collectors everywhere and everytime, create an optimized route for each collector in order to save energy as well as travel distance.

For collectors and janitors, they need an overview of their work calendar. They have to understand and memorize their tasks on a daily and weekly basis. Janitors should be able to communicate well with collectors, other janitors and back officers. Therefore, delay less than one second in real-time communication devices is compulsory. They also have to check in and check out tasks everyday and have to send notifications about the MCPs if they are fully loaded.

The UWC 2.0 will bring many benefits to each stakeholder. Back officers will feel easier to maintain, operate and manage the project. It will also be easier for janitors and collectors to communicate with the server which leads to less mistakes and delays. Finally, once the model is scaled up successfully, it will need more workforce to maintain a system which will create more employment opportunities in our society in the future.

1.2 Functional, non-functional requirements and general use-case diagram.

1.2.1 Functional and non-functional requirements

• Functional Requirement:

– As a back officer, I want to see an overview of janitors and collectors and their work calendar.

– As a back officer, I want to have an overview of all MCPs and information about their capacity.

– As a Back officer, I want to assign vehicles to janitors and collectors.

– As a Back officer, I want to assign janitors and collectors to MCPs.

– As a Back officer, I want to create a route for each collector.

– As a Back officer, I want to be able to send messages to collectors and janitors.

– As a janitor and collector, I want to have a detailed view of our tasks on a daily and weekly basis. All important information should be displayed in one view (without scrolling down).

– As a janitor and collector, I want to be able to communicate with collectors, other janitors and back officers.

– As a janitor and collector, I want to be able to check in / check out tasks every day.

– As a janitor and collector, I want to be notified about the MCPs if they are fully loaded.

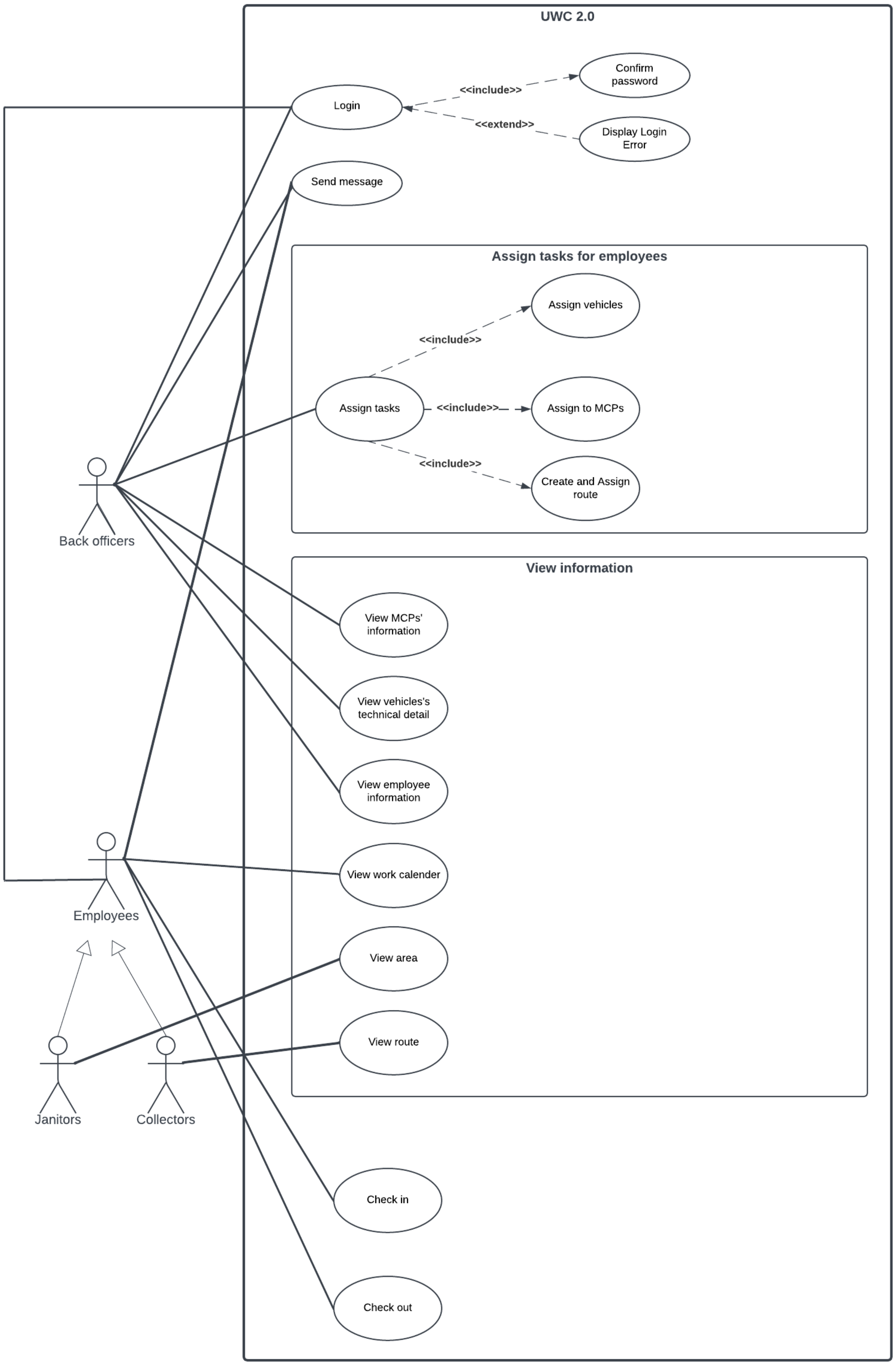
• Non-Functional Requirement:

– As a back officer, I want information on MCPs updated every 15 minutes with the availability of at least 95% of their operating time.

– As a back officer, I want each collector’s assigned route to be optimized in terms of fuel consumption and travel distance.

– As a back officer, janitor or collector, I want the messaging communicated in a real-time manner with delay less than 1 second.

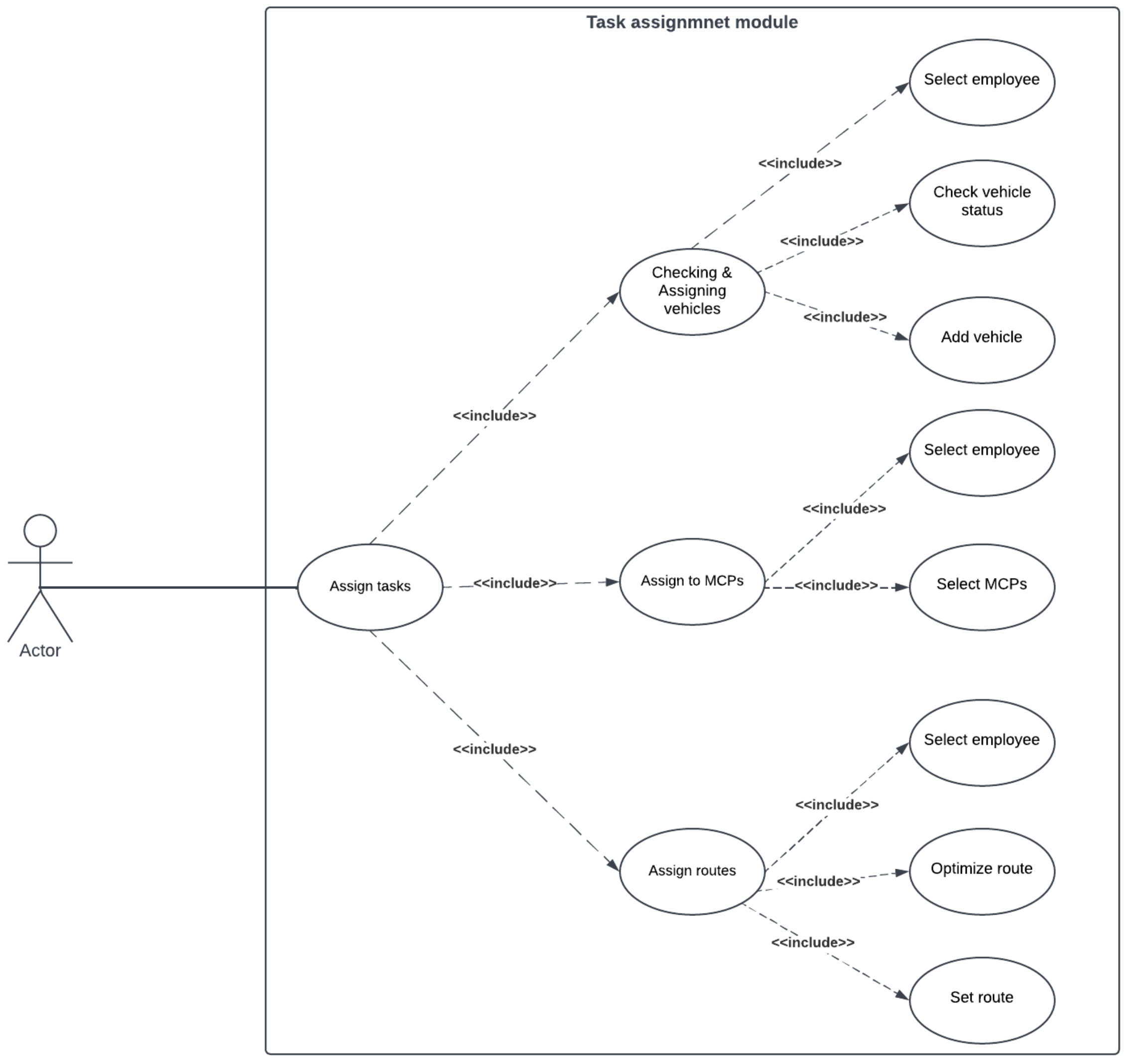
1.2.2 Use-case diagram



1.3 Use-case diagram for task assignment using a table format.

1.3.1 Use case diagram

The use case diagram in this section is similar to the use case diagram in section 1.2.2.



1.3.2 Table format for each use-case

Use-case No.1:

Use-case ID **Use-case name**	1 **Checking & Assigning vehicles**
Use-case overview	Back officers can assign vehicles to collectors and janitors. The vehicles for collectors and janitors are trucks and trollers, respectively.

Actors	Back officers
Preconditions	1. Account type must be a “back officer” account. 2. Back officers have already accessed the “Assign tasks” on the side bar.
Trigger	Back officers click the “Checking & Assigning vehicles” tab.
Normal flow	1. After clicking on the “Checking & Assigning vehicles”, the screen will display a list of workers. 2. Back officers choose workers to assign vehicles. 3. After selecting workers, a list of vehicles will be shown on the screen. 4. Back officers choose specific vehicles for his employees by selecting suitable vehicles for each kind of employee (trucks for collectors or trollers for janitors). 5. After choosing the appropriate vehicle, the screen will display a pop-up message “Assign vehicles successfully”.
Post conditions	Vehicles are assigned to employees afterwards.
Exception flow	Exception 1 at step 3: If an employee is currently having a vehicle, the screen will display a message “This employee has been assigned a vehicle” then returning to step 2. Exception 2 at step 3: If back officers choose employees with different roles, the screen will display a message “Only can choose employees with the same role” then returning to step 2.

Use-case No.2:

Use-case ID **Use-case name**	2 **Assign to MCPs**
Use-case overview	Assign collectors and janitors to suitable Major Collecting Points.
Actors	Back officers

Preconditions	1. Account type must be a “back officer” account. 2. Back officers have already accessed the “Assign tasks” on the side bar.
Trigger	Users click the “Assign to MCPs” button.
Normal flow	1. After clicking on the “Assign to MCPs” button, the screen will display a list of employees. 2. Back officers choose workers to assign to MCPs. 3. After selecting employees, the screen will display all Major Collecting Points that need to be collected. 4. Back officers can choose multiple MCPs for collectors and janitors. 5. After choosing MCPs for employees, the screen will display the message “Send people to MCPs successfully”.
Postconditions	MCPs are assigned to workers afterwards.
Exception flow	None

Use-case No.3:

Use-case ID **Use-case name**	3 **Assign routes**
Use-case overview	Assign optimal routes for the collectors.
Actors	Back officers
Preconditions	1. Account type must be a back officer. 2. Back officers have already accessed the “Assign tasks” on the side bar.
Trigger	Back officers click the “Assign route” button.

Normal flow	1. After clicking on the “Assign route” button, the screen will display a list of employees. 2. Back officers choose employees. 3. After selecting employees, the screen will display the optimal route from employees’ location to MCPs in terms of fuel consumption and travel distance. 4. Back officers select “Set route” in order to assign routes for employees. 5. After that, the screen will display the message “Assign route to employees successfully”.
Postconditions	Routes are assigned to workers afterwards.
Exception flow	Exception 1 at step 3: If the back officers select employees whose role are janitors, the screen will display the message “Only assign and create route for collectors”, return to step 2.

Use-case No.4:

Use-case ID	4
Use-case name	**Assign tasks**
Use-case overview	Back officers can set a working schedule for all collectors and janitors.
Actors	Back officers
Preconditions	Account type must be a back officer.
Trigger	Back officers click on “Assign tasks” button
Normal flow	1. The screen shows a list of tasks that back officers can assign to the employees. 2. Back officers select a specific task to assign. 3. After selecting a task, the system will run that task such as assign vehicles, assign to MCPs or assign routes(described clearly in other use case tables).
Postconditions	
Exception flow	None

2. Task 2: System Modeling

2.1 Activity diagram between systems and the stakeholders in the Task Assignment module.

2.1.1 Use case 1: Assign vehicles

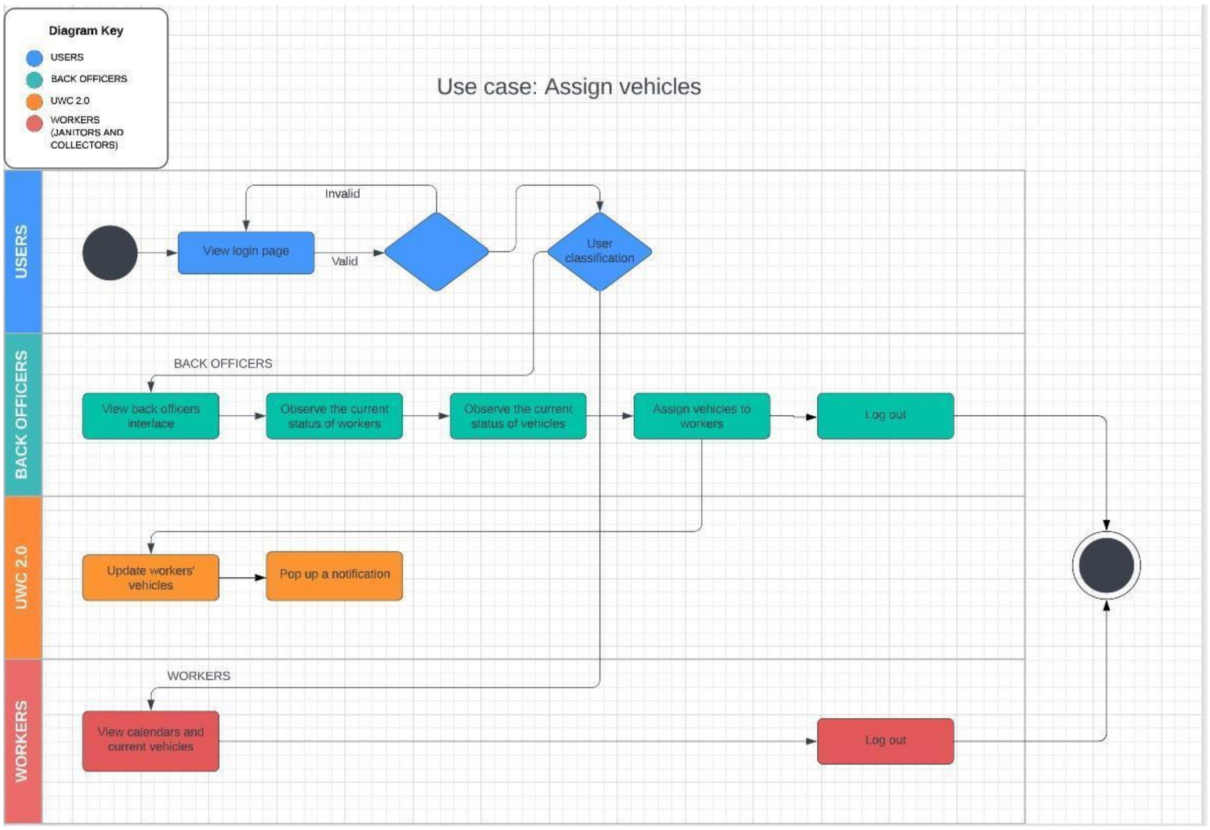


Diagram Key
USERS
BACK OFFICERS
UWC 2.0
WORKERS (JANITORS AND COLLECTORS)
Use case: Assign vehicles
USERS
View login page
Invalid
Valid
User classification
BACK OFFICERS
BACK OFFICERS
View back officers interface
Observe the current status of workers
Observe the current status of vehicles
Assign vehicles to workers
Log out
UWC 2.0
Update workers' vehicles
Pop up a notification
WORKERS
WORKERS
View calendars and current vehicles
Log out

Description:

- All users including back officers, janitors and collectors need to sign in the system. In the login page, after users put in their accounts, the system will verify and come out with a valid or invalid result. If it is invalid, the system asked users to sign in again. If it is valid, the system will automatically classify back officers and workers based on their private code in the personal account.
- After classification, workers will have their interface which shows their schedules and their current vehicles.
- Back officers will be provided their own interface. They will observe the current status of workers as well as the current status of vehicles before assigning those free vehicles to workers.
- The system UWC 2.0 will update the workers' vehicles.
- After the system UWC 2.0 updates the workers' vehicles successfully, the back officers will receive a notification which is "ASSIGN VEHICLES <A> TO WORKER <B> SUCCESSFULLY" and back officers can log out of the system.
- After the system UWC 2.0 updates the workers' vehicles successfully, the workers' devices pop up a notification which is "ASSIGNED <VEHICLES> SUCCESSFULLY" and workers can log out of the system.

2.1.2 Use case 2: Assign people to MCPs

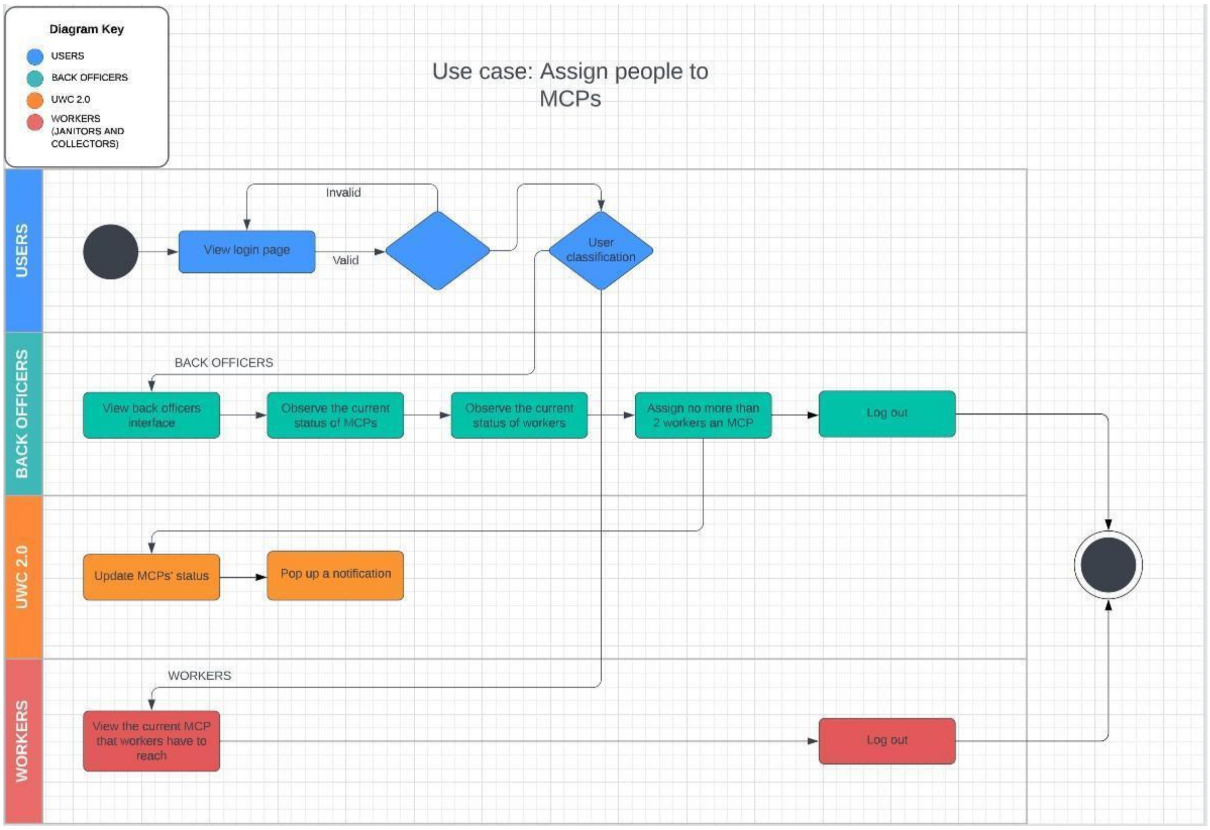


Diagram Key
USERS
BACK OFFICERS
UWC 2.0
WORKERS (JANITORS AND COLLECTORS)
Use case: Assign people to MCPs
USERS
View login page
Invalid
Valid
User classification
BACK OFFICERS
BACK OFFICERS
View back officers interface
Observe the current status of MCPs
Observe the current status of workers
Assign no more than 2 workers an MCP
Log out
UWC 2.0
Update MCPs' status
Pop up a notification
WORKERS
WORKERS
View the current MCP that workers have to reach
Log out

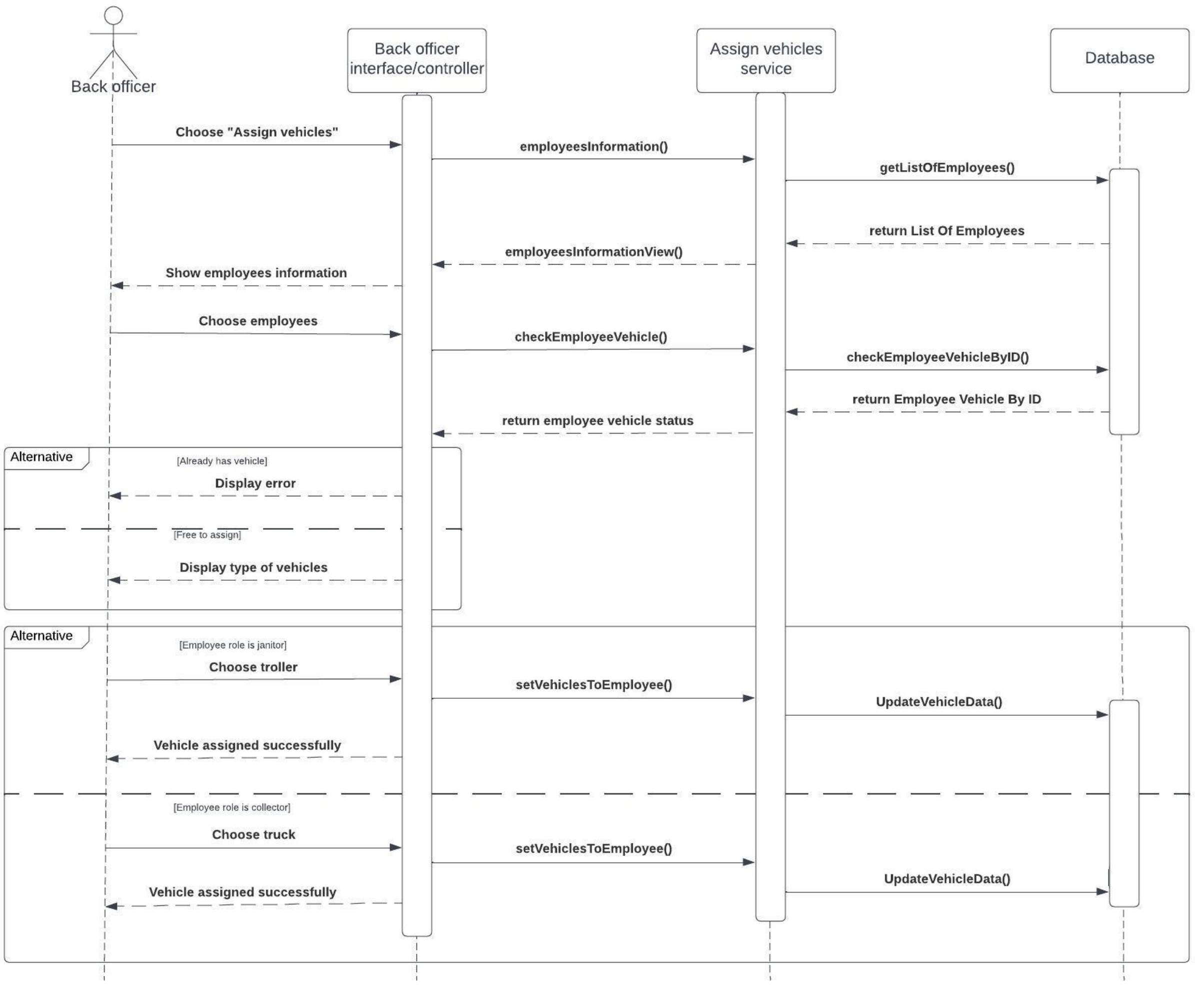
<u>Description:</u>

- All users including back officers, janitors and collectors need to sign in the system. In the login page, after users put in their accounts, the system will verify and come out with a valid or invalid result. If it is invalid, the system asks users to sign in again. If it is valid, the system will automatically classify back officers and workers based on their private code in the personal account.
- After classification, workers will have their interface which shows their destinations(MCP needs to be reached).
- Back officers will be provided their own interface. They will observe the current status of MCPs as well as the current status of workers before assigning those workers to MCPs.
- The system UWC 2.0 will update the MCPs' status.
- After the system UWC 2.0 updates the MCPs' status successfully, the back officers will receive a notification which is "ASSIGN WORKER <A> TO MCP <B> SUCCESSFULLY" and back officers can log out of the system.
- After the system UWC 2.0 updates the MCPs' status successfully, the workers' devices pop up a notification which is "ASSIGNED MCP <B> SUCCESSFULLY" and workers can log out of the system.

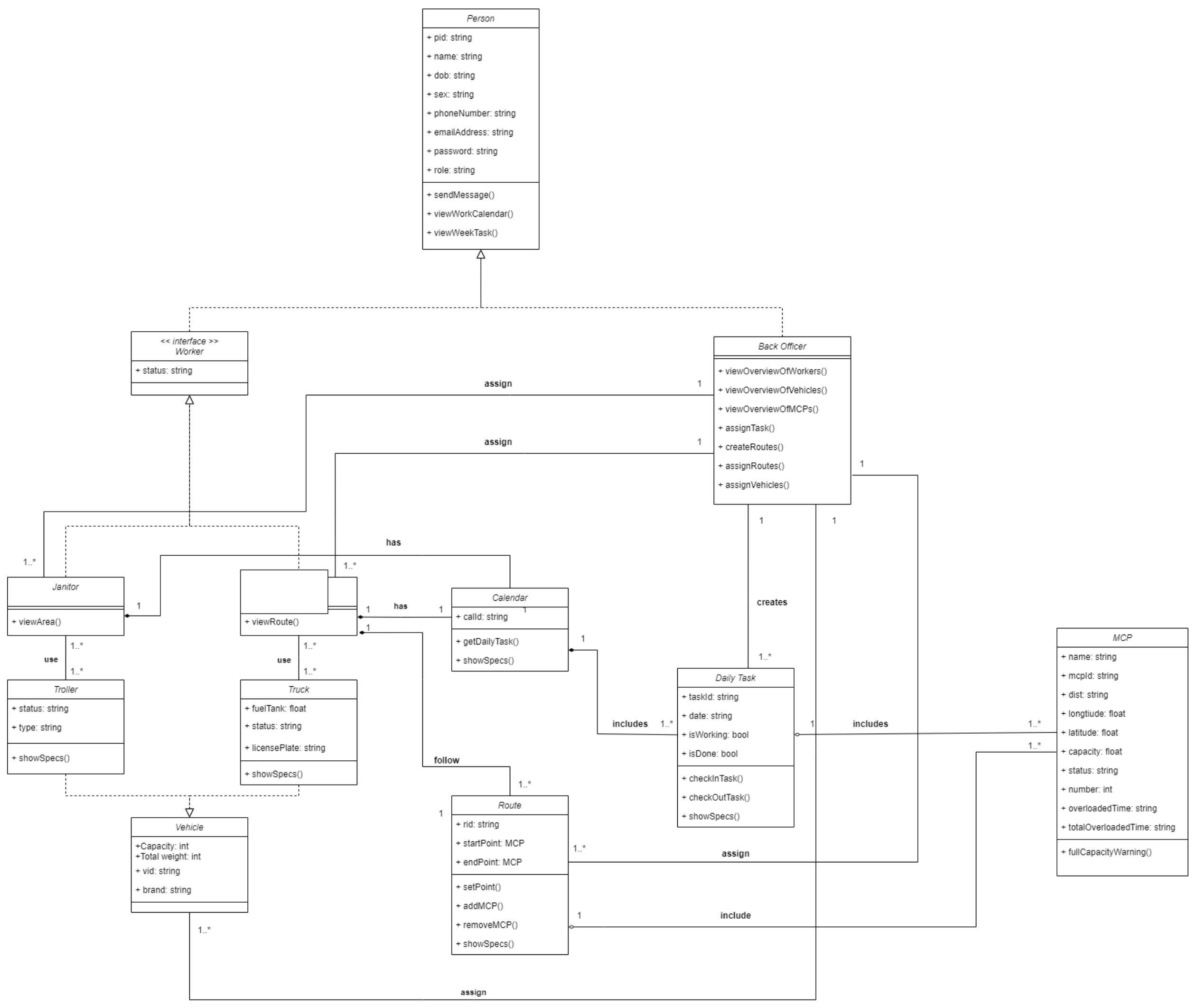
2.2 Sequence diagram illustrating the process that back officers assign vehicles to collectors

<u>Description:</u>

- First of all, back officers click on the “assign vehicles” button. The system retrieves the list of collectors from the database and displays them on the back officers’ screen.
- Next, back officers choose “assign” after the collector's name. The system retrieves the information from the database and checks if the collector already has a vehicle or not.
- If not, the list of vehicles will be displayed on the screen and back officers can assign the vehicles the collector and janitor.
- If the role of an employee is janitor, the back officers select . If the role of an employee is collector, the back officers select trucks. The vehicle will be assigned to a specific employee and the information of the vehicle will be updated in the database.



2.3. Class diagram of the Task Assignment module.

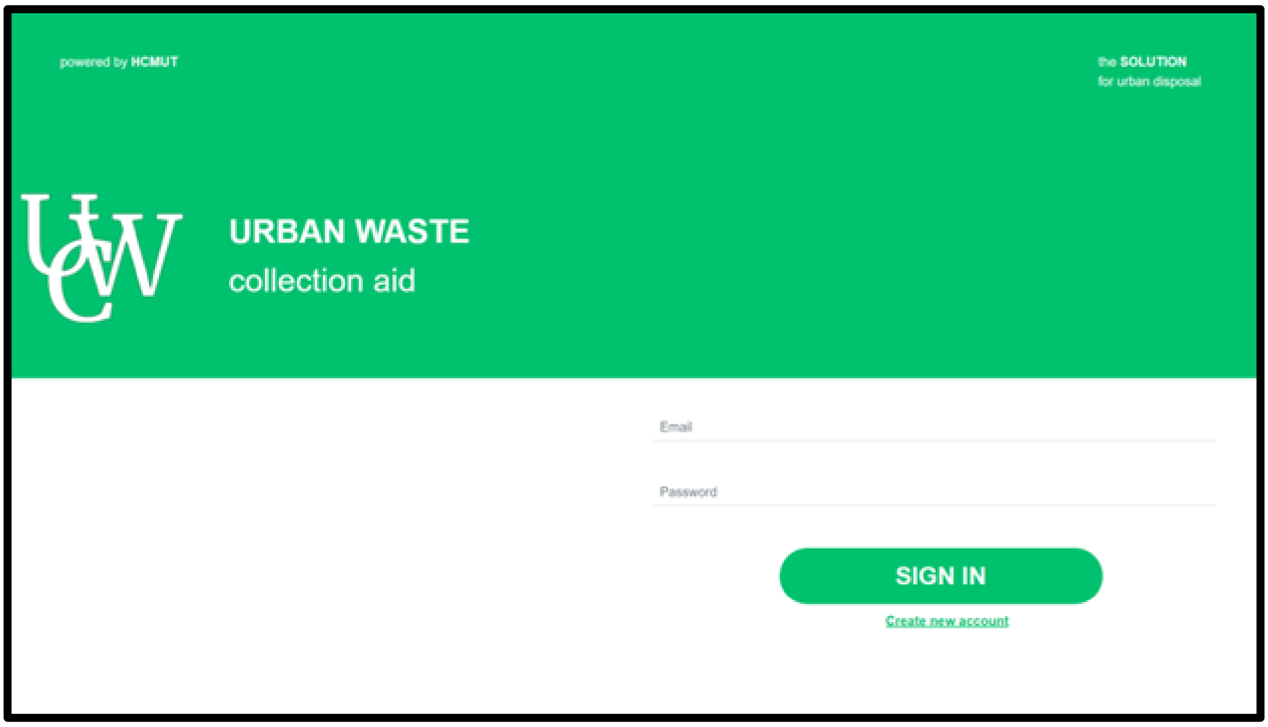


Click here for better inspection

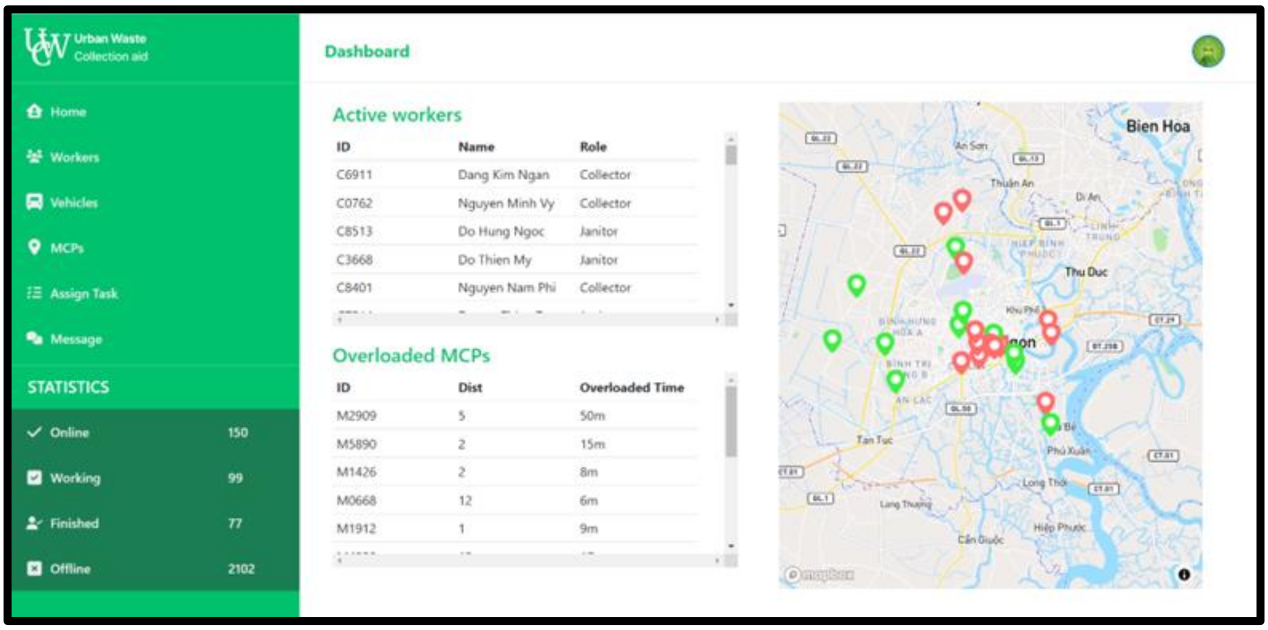
Description:

- The system has two types of users which are normal workers (including Janitors and Collectors) and back officers. Therefore, we perform generalization on **Worker** and **Back Officer** with a parent class **Person** which has some critical instances such as *pid*, *emailAddress*, *password*, *role*, etc. The class Worker acts as an interface for class **Janitor** and **Collector**. These classes have some methods depending on the role of the actor. For example, **Back Officer** has permission to assign tasks to workers so it has the method ***assignTask()*** whereas **Worker** only has permission to view the information. Methods used for viewing are ***viewInfo()***, ***viewWeekTask()*** and ***viewWorkCalendar()***,...
- Because we need to handle information about vehicles and MCPs, we create two classes which are **Vehicle** and **MCP**. **Vehicle** class has instances as follows: *vid*, *weight*, *brand*, *licensePlate* and *capacity*. Two types of vehicles are trucks and trollers, so we create a class **Truck** and a class **Troller** which are derived from the class **Vehicle**. **MCP** class has information about its location, so it contains the instance *dist* as the district it is allocated and information about longitude and latitude on the map. In order for back officers to easily specify and assign MCPs to workers, **MCP** classes must have a number and *name*. Back officers can also know about the status of the MCPs if they are overloaded or not by the instance status. Moreover, if the MCPs are overloaded, there are instances to keep track of the time that they are being overloaded.
- **Janitor** and **Collector** can both view their **Calendar** which includes their **Daily Task**, while **Collector** also needs to know about the **Route** leading to those MCPs.

2.4 Desktop-view central dashboard



Login page



STATISTICS

Online	150
Working	99
Finished	77
Offline	2102

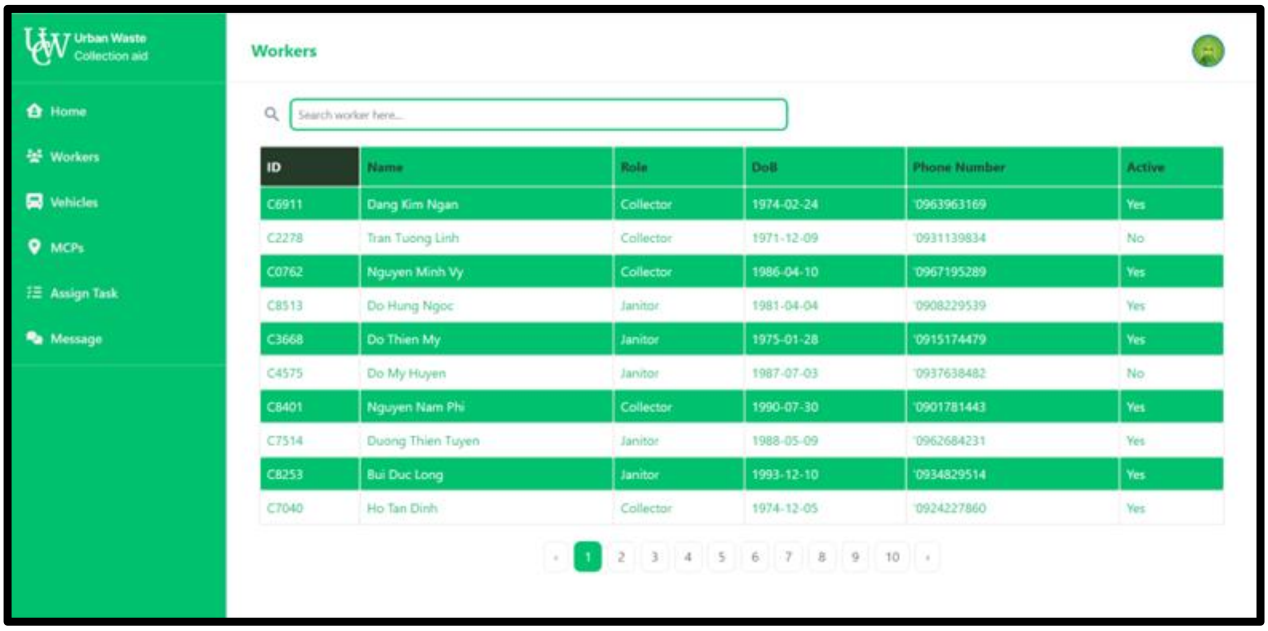
Active workers

ID	Name	Role
C6911	Dang Kim Ngan	Collector
C0762	Nguyen Minh Vy	Collector
C8513	Do Hung Ngoc	Janitor
C3668	Do Thien My	Janitor
C8401	Nguyen Nam Phi	Collector

Overloaded MCPs

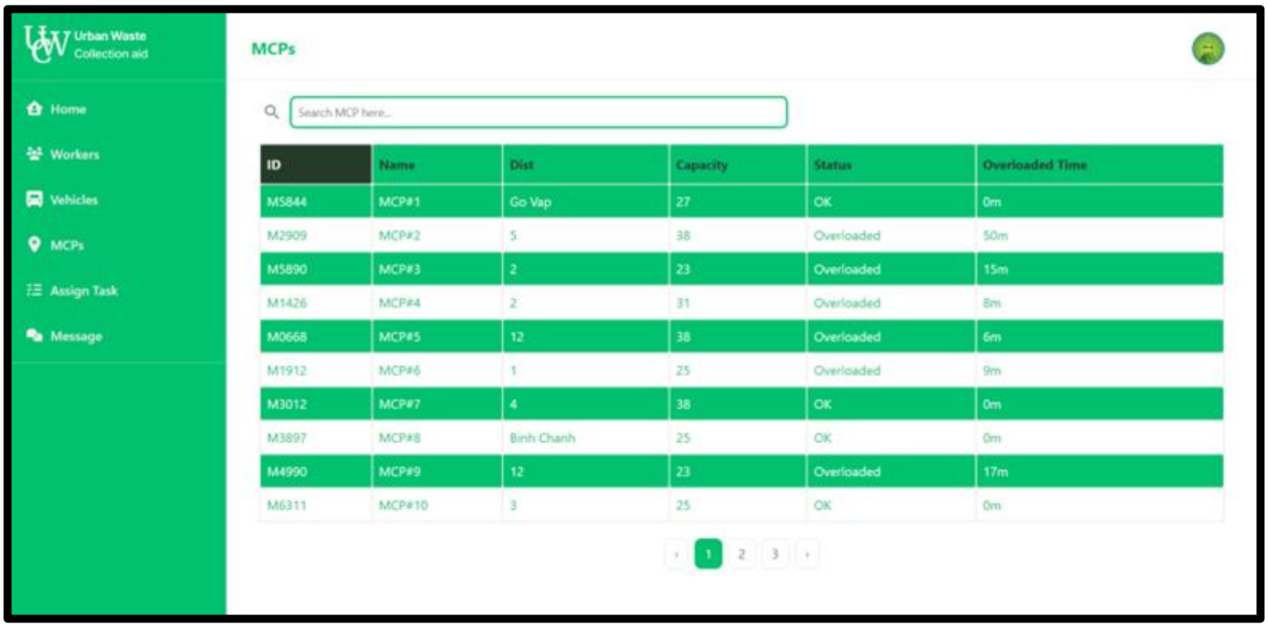
ID	Dist	Overloaded Time
M2909	5	50m
M5890	2	15m
M1426	2	8m
M0668	12	6m
M1912	1	9m

Home page



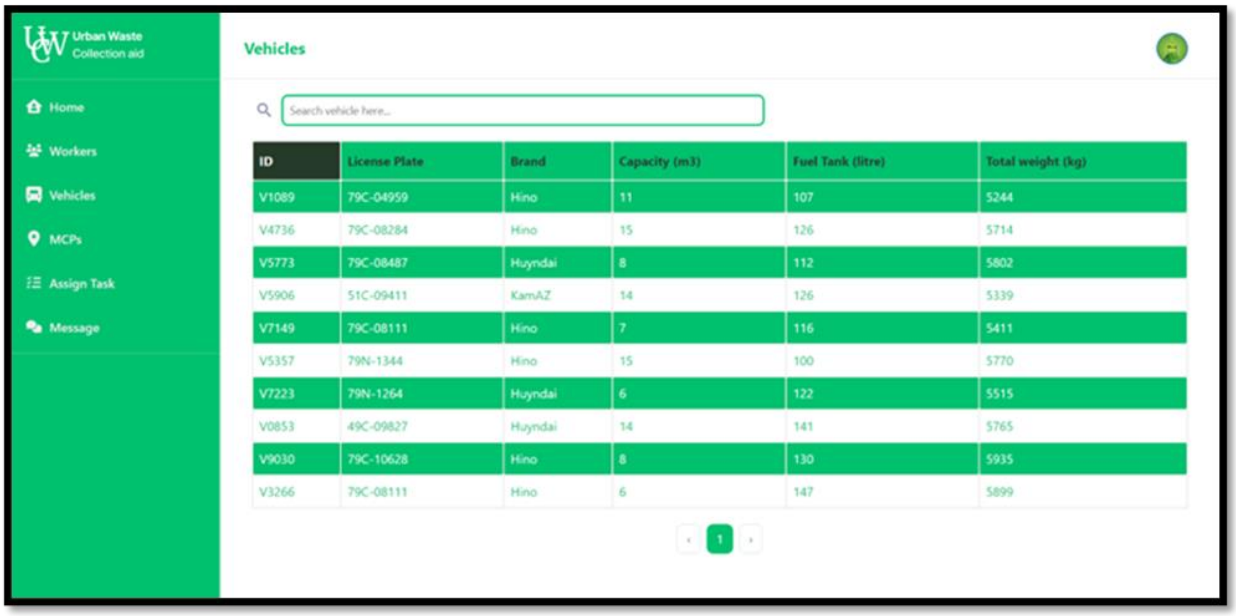
ID	Name	Role	DoB	Phone Number	Active
C6911	Dang Kim Ngan	Collector	1974-02-24	'0963963169	Yes
C2278	Tran Tuong Linh	Collector	1971-12-09	'0931139834	No
C0762	Nguyen Minh Vy	Collector	1986-04-10	'0967195289	Yes
C8513	Do Hung Ngoc	Janitor	1981-04-04	'0908229539	Yes
C3668	Do Thien My	Janitor	1975-01-28	'0915174479	Yes
C4575	Do My Huyen	Janitor	1987-07-03	'0937638482	No
C8401	Nguyen Nam Phi	Collector	1990-07-30	'0901781443	Yes
C7514	Duong Thien Tuyen	Janitor	1988-05-09	'0962684231	Yes
C8253	Bui Duc Long	Janitor	1993-12-10	'0934829514	Yes
C7040	Ho Tan Dinh	Collector	1974-12-05	'0924227860	Yes

Workers information page



ID	Name	Dist	Capacity	Status	Overloaded Time
M5844	MCP#1	Go Vap	27	OK	0m
M2909	MCP#2	5	38	Overloaded	50m
M5890	MCP#3	2	23	Overloaded	15m
M1426	MCP#4	2	31	Overloaded	8m
M0668	MCP#5	12	38	Overloaded	6m
M1912	MCP#6	1	25	Overloaded	9m
M3012	MCP#7	4	38	OK	0m
M3897	MCP#8	Binh Chanh	25	OK	0m
M4990	MCP#9	12	23	Overloaded	17m
M6311	MCP#10	3	25	OK	0m

MCPS information page

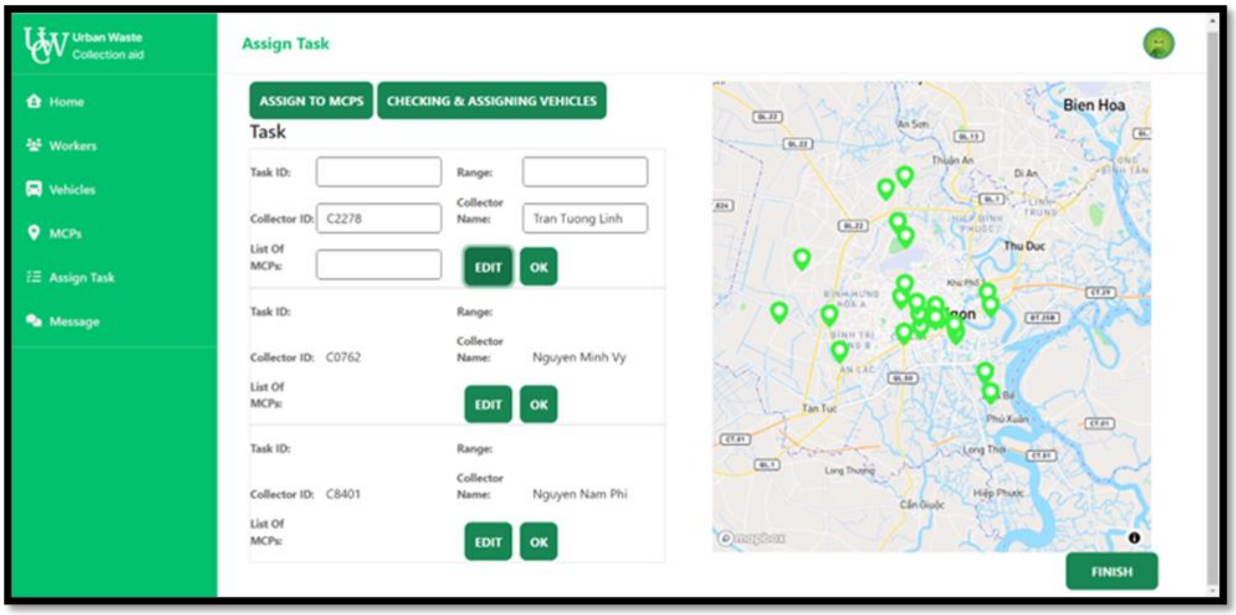


Vehicles

Search vehicle here...

ID	License Plate	Brand	Capacity (m3)	Fuel Tank (litre)	Total weight (kg)
V1089	79C-04959	Hino	11	107	5244
V4736	79C-08284	Hino	15	126	5714
V5773	79C-08487	Huyndai	8	112	5802
V5906	51C-09411	KamAZ	14	126	5339
V7149	79C-08111	Hino	7	116	5411
V5357	79N-1344	Hino	15	100	5770
V7223	79N-1264	Huyndai	6	122	5515
V0853	49C-09827	Huyndai	14	141	5765
V9030	79C-10628	Hino	8	130	5935
V3266	79C-08111	Hino	6	147	5899

Vehicles information page



Assign Task

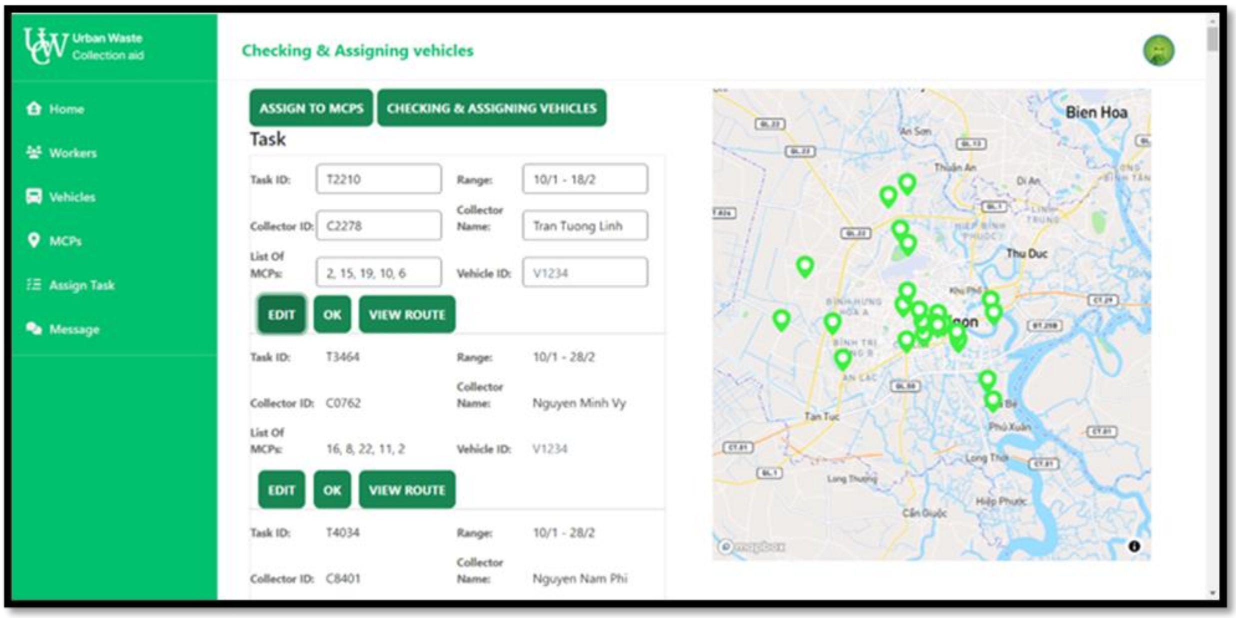
Task

Task ID: | Range:
Collector ID: C2278 | Collector Name: Tran Tuong Linh
List Of MCPs:

Task ID: | Range:
Collector ID: C0762 | Collector Name: Nguyen Minh Vy
List Of MCPs:

Task ID: | Range:
Collector ID: C8401 | Collector Name: Nguyen Nam Phi
List Of MCPs:

Assign to MCPS page

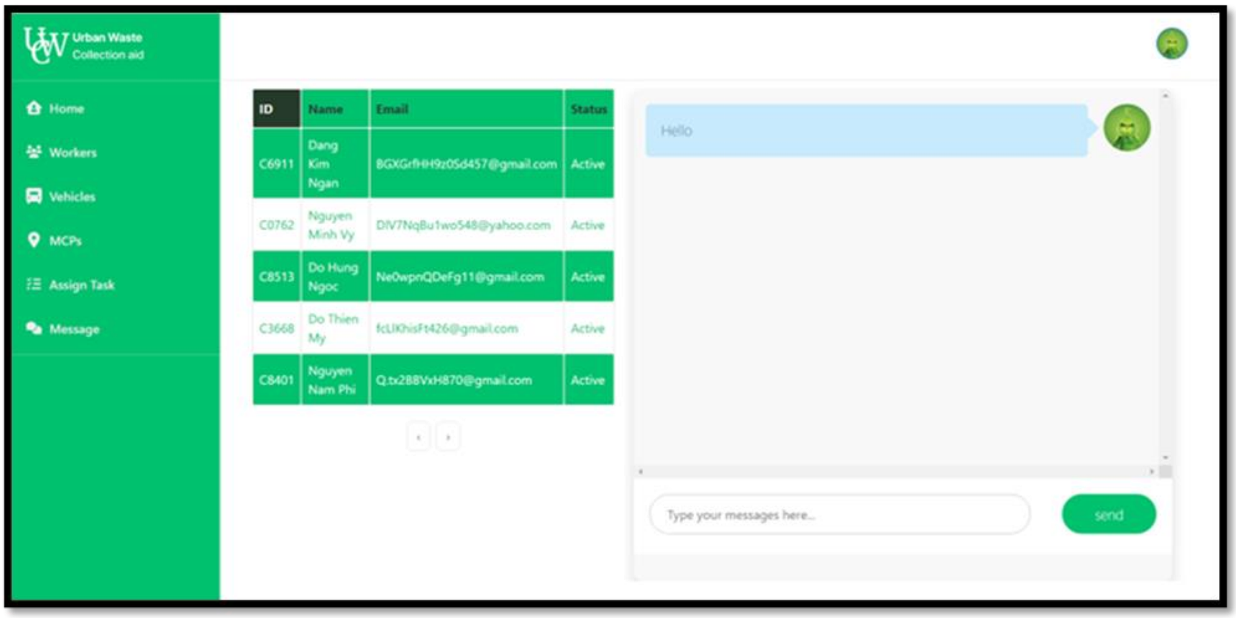


Checking & Assigning vehicles

Task

Task ID:	T2210	Range:	10/1 - 18/2
Collector ID:	C2278	Collector Name:	Tran Tuong Linh
List Of MCPs:	2, 15, 19, 10, 6	Vehicle ID:	V1234
Task ID:	T3464	Range:	10/1 - 28/2
Collector ID:	C0762	Collector Name:	Nguyen Minh Vy
List Of MCPs:	16, 8, 22, 11, 2	Vehicle ID:	V1234
Task ID:	T4034	Range:	10/1 - 28/2
Collector ID:	C8401	Collector Name:	Nguyen Nam Phi

Checking and assigning vehicles page



ID	Name	Email	Status
C6911	Dang Kim Ngan	BGXGrfHH9z0Sd457@gmail.com	Active
C0762	Nguyen Minh Vy	DIV7NqBu1wo548@yahoo.com	Active
C8513	Do Hung Ngoc	NeOwpnQDeFg11@gmail.com	Active
C3668	Do Thien My	fcLIKhisFt426@gmail.com	Active
C8401	Nguyen Nam Phi	Q.tx288VxH870@gmail.com	Active

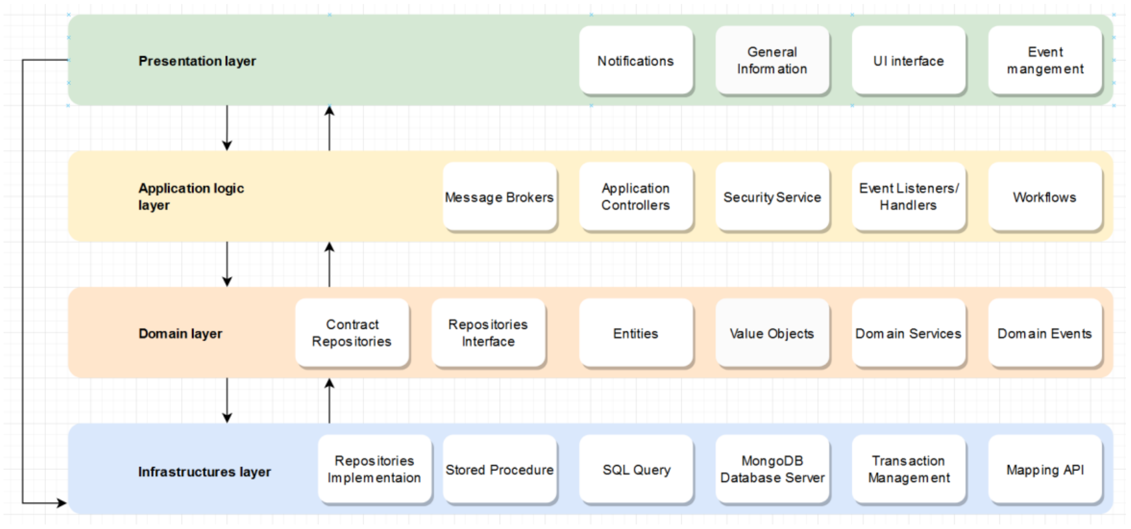
Hello

Type your messages here...

Send message page

3. Task 3: Architecture design

3.1 Layered Architecture design



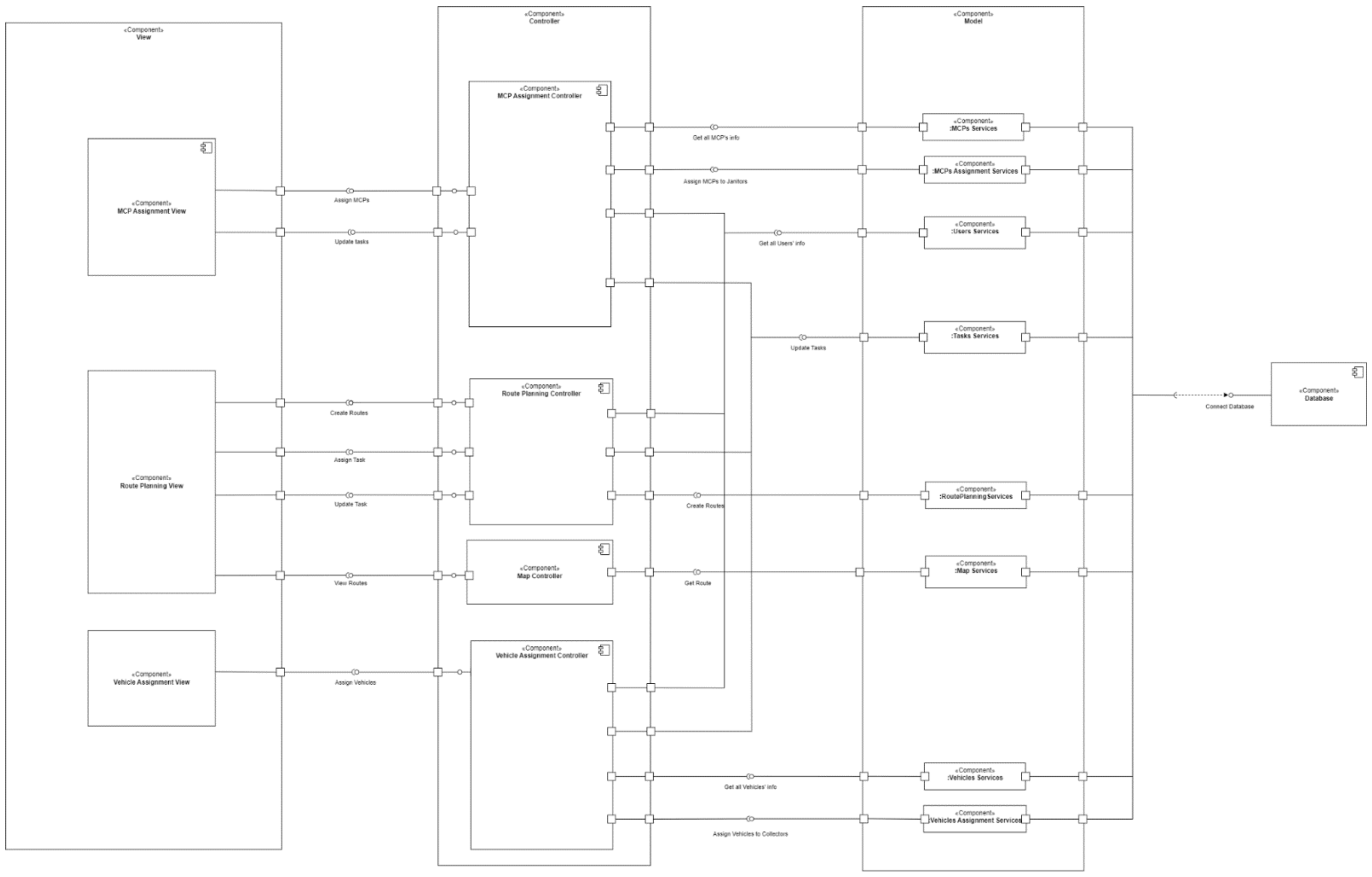
- Presentation layer:
    - The Presentation layer is responsible for managing the user interface and user interactions with the system. This layer includes displaying information to the users such as notifications, general information, menus day, time, buttons, maps as well as handling user inputs through UI interface. The Presentation layer also manages events, such as button-clicking or form submissions and communicates with other layers of the system to process user requests and update the UI accordingly.
    - These components work together so as to provide a smooth and responsive user experience. The Presentation layer separates concerns related to user interaction from the underlying business logic, making it easier to develop and maintain complex applications.

- Application layer:
    - The Application Controller is responsible for coordinating the flow of the application. It receives requests from the user interface and directs them to the appropriate application services for processing. The Application Controller also manages the overall state of the application and handles any error that may occur.
    - An Authentication Service is a component of an application that is responsible for managing user authentication and authorization. In the Application layer, the Authentication Service is implemented as a separate component that interacts with other components in the layer to verify user identity and determine access rights. When the access is authenticated, the application continues to get to the next layer.
    - Message Brokers is used for sending message applications. This broker is a gateway for data to come in or come out of the Application layer.
    - Event Listeners are responsible for responding to events that occur within the system. They can be used to perform tasks such as logging, auditing, or sending notifications. Event Listeners can be triggered by user actions or by changes in the underlying data.

- Domain layer:
    - Entities are objects that have a unique identity and represent a specific concept within the system. They have attributes and behaviors that define their states and behaviors.
    - Value Objects are objects that represent a specific value or a set of values. They do not have a unique identity and they are typically used to represent simple concepts like dates or addresses.
    - Domain Services are responsible for implementing complex business logic that does not fit within a single entity or collection. They provide methods for performing tasks such as calculations or validations.
    - Domain Events are events that occur within the system as a result of changes to the domain model. They can be used to trigger actions or notify other components of changes within the system.
    - Finally, Contract Repositories is a contract repository where contracts are stored.

- Infrastructure layer
    - A database server is a software program that provides database services to other computer programs or computers. In this project, MongoDB is used to store waste business data. It stores and manages data in a structured format allowing efficient storage and retrieval of information.
    - Transaction Management is to ensure the integrity and consistency of the data in the UWC 2.0 database. This is achieved by ensuring that either all the operations in a transaction are completed successfully or none of them are applied at all. In the event of a failure, the transaction can roll back to its initial state, ensuring the database to remain in a consistent state. Transaction management is typically handled by the database management system, which provides mechanisms for defining and managing transactions.
    - Mapping API is a third-party service that provides mapping and location-based services to an application. In the context of the data layer, a mapping API would be integrated as one of the third-party APIs that the data layer interacts with.
    - Stored procedure: including steps to transform data for specific purposes (finding the often overloaded landfills, …).
    - SQL query: used to interact with UWC 2.0 relational database. SQL is a standard language for managing and manipulating data stored in a relational database. It allows users to write queries to retrieve, insert, update, and delete data from the UWC 2.0 database.

3.2 Component Diagram


«Component» View
«Component» MCP Assignment View
«Component» Route Planning View
«Component» Vehicle Assignment View
Assign MCPs
Update tasks
Create Routes
Assign Task
Update Task
View Routes
Assign Vehicles
«Component» Controller
«Component» MCP Assignment Controller
«Component» Route Planning Controller
«Component» Map Controller
«Component» Vehicle Assignment Controller
Get all MCP's info
Assign MCPs to Janitors
Get all Users' info
Update Tasks
Create Routes
Get Route
Get all Vehicles' info
Assign Vehicles to Collectors
«Component» Model
«Component» :MCPs Services
«Component» :MCPs Assignment Services
«Component» :Users Services
«Component» :Tasks Services
«Component» :RoutePlanningServices
«Component» :Map Services
«Component» :Vehicles Services
«Component» :Vehicles Assignment Services
Connect Database
«Component» Database

Click here for better inspection

<u>Description:</u>

- The Task Assignment will be exclusive for Back Officers.
- The Task Assignment Module will include 3 main functions: *MCP Assignment*, *Route Planning* and *Vehicle Assignment*
- Each of these functions will have their own view and controller, which all has access to several **services** - which help them access the database and retrieve/update data they need.
- *MCP Assignment* will be used to create and assign tasks (MCP or list of MCPs) to Collectors and Janitors.
- *Route Planning* can create optimized routes and assign them to Collectors based on their assigned lists of MCPs.
- *Vehicle Assignment* will be in charge of assigning vehicles to Collectors and Janitors.

4. Task 4: Implementation – Sprint 1

4.1 Online repository (github, bitbucket, etc) for version control.

Project's online repository: click here

4.2 Documents, materials and folders for Requirement, System modeling and Architectural design.

For this section, please check out the folder documents and diagrams in the Github repository

4.3 Usability test with the user interface that developed in MVP1.

For this section, please check out the folder documents and diagrams, then select “testing.pdf” for more details.

5. Task 5: Implementation – Sprint 2

5.1 Develop MVP2 with input from Task 2.4 and Task 4.

For this section, this is a short demo of our project: click here

5.2 Demonstrate the whole project from Task 1 to Task 5

For this section, this is our presentation slide: click here