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

SOFTWARE ENGINEERING

Project

Urban waste collection aid 2.0

Instructor:	Truong Tuan Anh
Students:	Le Nguyen Gia Nghi- 1952868
	Tran Do Dang Khoa - 2053142
	Tran Gia Han - 2052464
	Pham Quynh Tran - 2053518
	Do Truong Thinh - 2052725

Contents

1 Task 1: Requirement elicitation

1.1 Overview context

1.1.1 Project's context

In recent years, life quality in rural areas has been significantly improved in most parts of the world. Thanks to the development of the industries, most services like Food and Beverage (FB), healthcare, transportation and commuting, technology, etc., have been better than ever, which leads to a noticeable rise in the economy. This also comes with a downside: the more products we make, the more wastes we create. If the amount of actual wastes exceed the amount of wastes we can process, the remaining unprocess wastes will then start to damage our living environment. Without some proper actions to counter it, our ecosystem will soon be in danger. The huge increment in the amount of trash demands a stronger waste managing system to handle it.

This project is carried out for that reason, its main aim is to upgrade the waste collecting management system from the old version of Urban Waste Collection aid (UWC 1.0) by optimizing the waste collecting process and reducing the cost from using and maintaining it.

1.1.2 Relevant stakeholders

Back officers:

Back officers are the head of the system, who deals with managing work. They are responsible for creating working schedule for janitors and collectors, managing vehicles, and overlook the MCPs.

Janitors:

Janitors' job is to collect wastes from assigned locations and bring them back to their MCP by a troller for collectors to collect.

Collectors:

After janitors have gathered garbage to the MCPs, it is collectors' job to drive a waste containing vehicle and pick those up to the waste processing unit. With the help of UWC 2.0, collectors may gain access to maps, up-to-date information about their routes and MCP and have better knowledge of the surroundings and conditions.

1.1.3 Current needs

Back officers:

Back officer need a system which gives them the ability to create a working schedule for janitors and collectors, view the information of the vehicles before distributing them with the corresponding staff, and know the status of all MCPs with their capacity.

Janitors:

To complete their task, janitors need to access to their work calendar and the details of their task, including which MCPs they will work with, which collectors and janitors are assigned to that MCP. They also need a good communication channel to get notified when needed and to report their problems when necessary.

Collectors:

Collectors need to know the details of their vehicle, such as its status, its maintenance history, its capacity, etc. Because this job demands collectors to travel through long routes, they also need to plan optimized routes to save time and fuel and have an overview of the MCPs.

1.1.4 Current problems

Back officers:

The expansion of the garbage collection system which the current UWC 1.0 cannot handle could cause many problems to back officers as well as janitors and collectors when managing the tasks. The performance of the current systems could also prevent back officers to create and assign tasks more efficiently. Back officer also need the basic knowledge to understand the system thoroughly and be able to deal with risks that may come when the system is broken down.

Janitors:

Janitors could have been struggling with staying connected with their coworkers as they are traveling from place to place and may not have sustainable connections. They are not familiar with using an e-schedule to view their tasks for checking in and out.

Collectors:

Collectors' efficiency depends greatly on their route planning. Their current problem could be the lack of information on route overview with vehicles' technical details taken into account for route planning which affects their workflow.

1.1.5 Benefits of UWC 2.0

UWC 2.0 will be an overhaul of the old UWC 1.0 system. The new system is capable of handling real-time data from at least 1000 MCPs in the initial phase and could be expanded up to 10000 in 5 years to cope with the expansion of the garbage collection system.

With the route optimization feature, collectors won't have to take time for planning as the system is now able to make computation based on fuel consumption and travel distance and output the optimized route.

UWC 2.0 will see a renovation in user interfaces and experiences in order to organize information in a better way for users to interact with ease and improve the efficiency of the system.

Text-messages communication is also available for users to stay connected during work with delay less than 1 second for the best efficiency in communication between workers.

1.2 Functional and non-functional requirements

1.2.1 Functional requirements

Back officers:

- Can sign in the system as back officers.
- Have an overview of janitors and collectors, their work calendar.
- Have an overview of the number of workers who can work on each day.
- Have an overview of vehicles and their technical details (weight, capacity, fuel consumption, etc).

- Have an overview of all MCPs and information about their capacity.
- Assign vehicles to janitors and collectors.
- Assign janitors and collectors to MCPs (task).
- Create a route for each collector. Assigned route is optimized in terms of fuel consumption and travel distance.
- Be able to send message to collectors and janitors.

Collectors and janitors:

- Can sign in the system as collectors/ janitors.
- Have an overview of their work calendar.
- Have a detailed view of their task on a daily and weekly basic (calendar, tasks, vehicles, routes). All important information should be displayed in one view (without scrolling down).
- Be able to communicate with collectors, other janitors and back officers.
- Check in / check out tasks every day.
- Be notified about the MCPs if they are fully loaded.

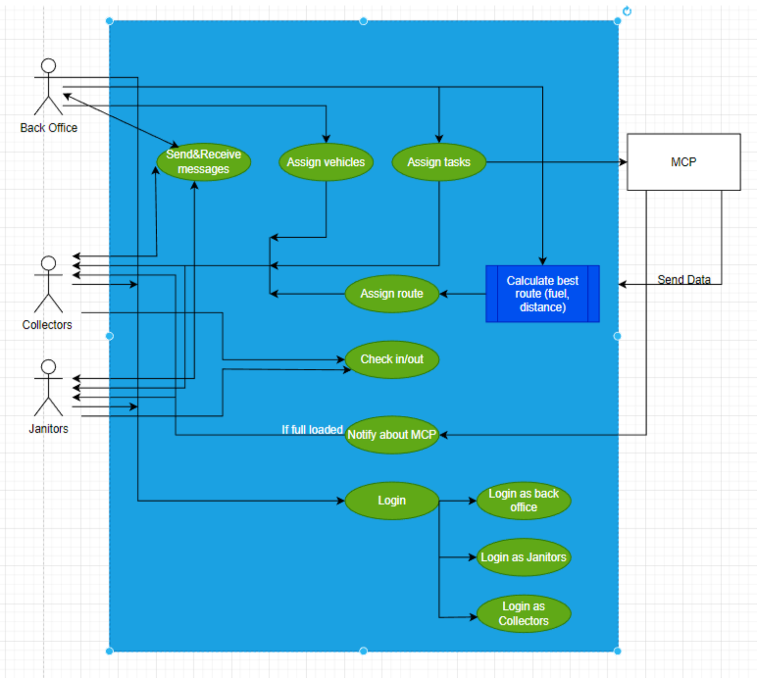
Some other requirements for the system:

- Store information about each user's account.
- Be able to delegate access to different users.
- Record the information of each working day.
- System interfaces should be in Vietnamese, with an opportunity to switch to English in the future.

1.2.2 Non-Functional requirements

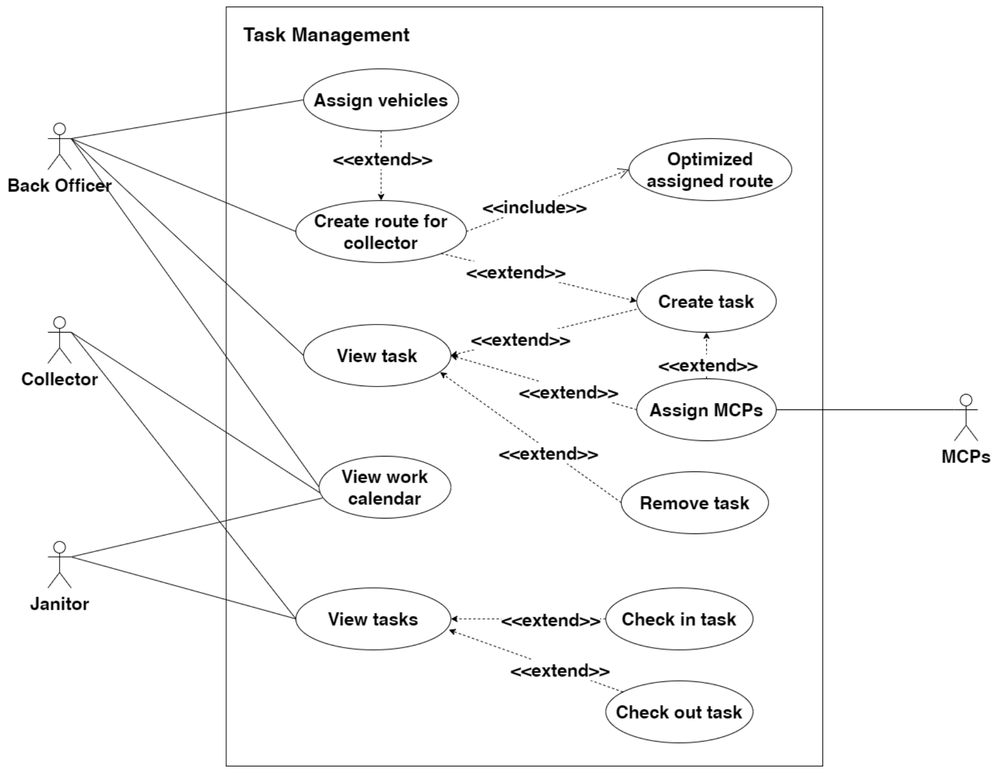
Performance	-Web pages load no more than 2 seconds. -The process of sending data should not exceed 3 seconds. -Response to remaining operations should be less than 1 second. -Be able to handle real-time data from at least 1000 MCPs. -Information should be updated from MCPs every 15 minutes with the availability of at least 95% of their operating time.
Reliability	-Ready to work everyday. -Small error rate.
Security	-Ensure compliance with the laws and regulations of the host country. -Admin accounts are allowed to be entered incorrectly up to 5 times. -Warn if another server IP logged in.
Scalability	-Be able to handle real-time data from at least 10.000 MCPs in five years. -Both English and Vietnamese support in the future.
Backup	-Save data every working day.

1.2.3 Use-case diagram for the whole system



1.3 Task assignment

1.3.1 Use-case diagram



1.3.2 Use-case using a table format

Use-case name	Assign route for collector
Primary Actor	Back officer
Description	The back officer assigns a route for the collector by specifying the destination.
Trigger	Back officer clicks on assign route button
Preconditions	User has logged in as back officer
Postconditions	An optimized route is sent to the collector
Normal flow	Assigning route for collector 1. Back officer specifies the desired destination 2. System looks up the collector's assigned vehicle 3. System create route optimized in term of fuel consumption and travel distance of the vehicle 4. Return the route to back officers and to the assigned collector
Alternative flow	Collector has not been assigned a vehicle. 1. System notifies the back officer and brings up the vehicle view. 2. Back officer selects and assigns a vehicle for the collector 3. System create route optimized in term of fuel consumption and travel distance of the vehicle 4. System looks up the collector's assigned vehicle 5. System create route optimized in term of fuel consumption and travel distance of the vehicle 6. Return the route to back officers and to the assigned collector
Exception	No optimized route available

Use-case name	View task
Primary Actor	Back officer
Description	The back officer views, creates, or modifies the tasks in the task management module.
Trigger	Back officer enters the task management module
Preconditions	User has logged in as back officer
Postconditions	Tasks is viewed or modified
Normal flow	Create task 1. Back officer creates a new task 2. Back officer specifies the task information, assign the MCP, janitor, and collector to the task 3. System creates the task and notifies relating individuals Remove task 1. Back officer selects an existing task 2. Back officer clicks on the remove option 3. System removes the task and notifies relating individuals
Alternative flow	Duplicated and conflicting tasks assigned to janitors, collectors, and MCPs. 1. System notifies the back officer about the conflicts 2. Back officer is prompted to assign the task again
Exception	Duplicated and conflicting tasks

Use-case name	View task
Primary Actor	Janitor and Collector
Description	The janitor and collector views, check-in, and check-out tasks in the task management module.
Trigger	Janitor or collector enters the task management module
Preconditions	User has logged in as janitor or collector
Postconditions	Tasks is viewed or modified
Normal flow	Check-in task 1. Janitor or collector select an existing task assigned to them 2. Janitor or collector check-in the task to start working on it. 3. System updates task's status to in-progress and notifies the back officer Check-out task 1. Janitor or collector select an undergoing task assigned to them 2. Janitor or collector check-out after completing to task 3. System updates task's status to completed and notifies the back officer
Alternative flow	Work is suspended due to incident 1. Janitor or collector select the task and update the status. 2. System updates task's status to on-halt and notifies the back officer
Exception	Incidents occur during work

2 Task 2: System modelling

2.1 Activity diagram of Task assignment module

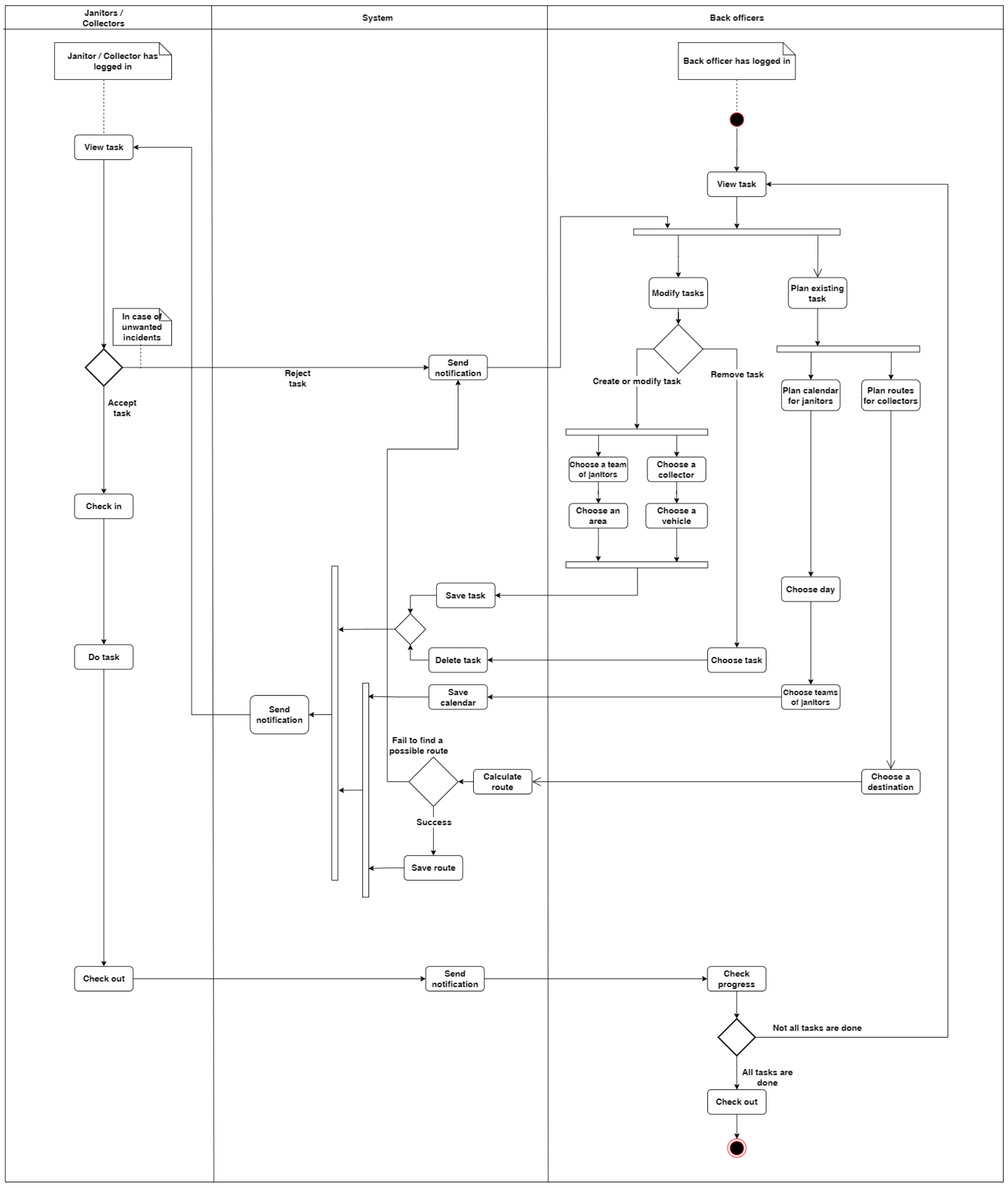


Figure 1: Activity diagram capturing the business process in Task Assignment module

Description

- To start assigning tasks, back officers must have logged in to the system
- After the back officers have check in, they can view a the tasks before starting an action. There are two main types of action: planing tasks for later assignment or assigning tasks.
- There are two actions back officers can do when planing tasks. Firstly, they can plan a work calendar for janitors so janitors can know which day they have to work and who they will work with. Secondly, back officers can plan vehicles and routes for collectors. This action does not require choosing collectors, it only needs the back officers to create a route based on the gathered information and choose a sufficient vehicle to go on with that route.
- For assigning tasks, back officers can also do two actions. First is to create a task. Creating a task for a team of janitors in a specific day requires back officers to choose an area where they will clean up and gather the wastes, while creating a task for a collector requires back officers to choose a collector for each pre-designed route. A notification will then be sent to collectors and janitors once an assignment is made. Collectors and janitors is then asked to accept the task or reject the task. If for some reason a task is rejected due to unwanted incidents (even in mid-operation), back officers may delete a task and create a new one. If a task is accepted, collector will then do the task and check out once the work have been done.
- When all tasks have been completed, the back officers can then check out and end their day work.

2.2 Conceptual solution for Route planning task

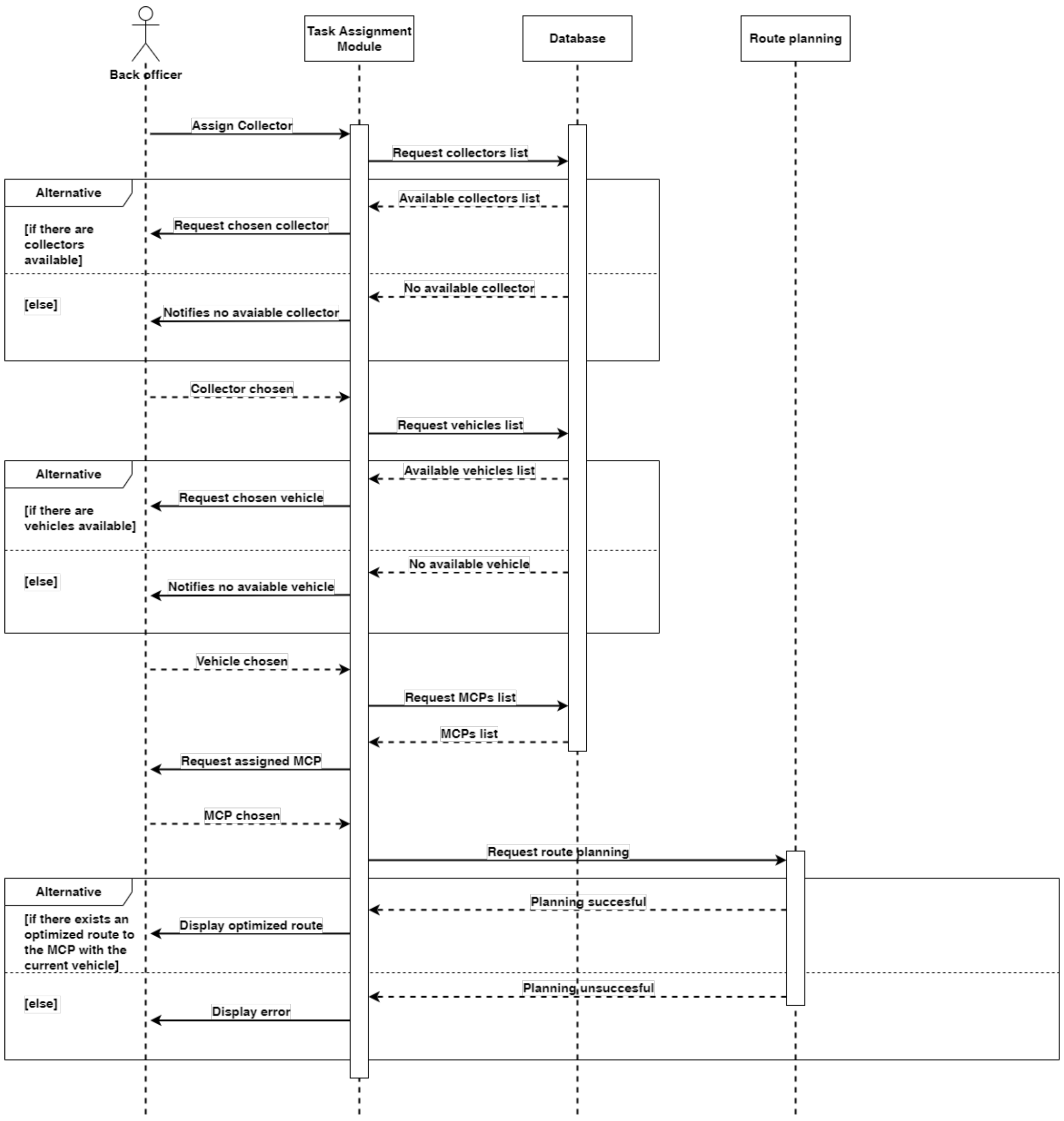


Figure 2: Sequence diagram of the route planning task

- Prior to the route planning task, the task assignment module will first prompt the back officer to assign a collector to an MCP with a chosen vehicle. If there is no available collectors or vehicles for the task, the module will notify the back officer about the situation.
- If the assignment is satisfied, the route planning task will begin. The task assignment module request a route planning from the route planning module with the vehicle and MCP assigned to the task.
- The route planning module performs some algorithms to calculate the optimized route

based on the constraints provided by the task management module. If an optimized route is determined, the module should response with a success message and the optimized route, the task assignment module will then display the route for the task. On the other hand, if the route can not be specified (e.g. the vehicle is not capable of reaching the destination, there's no route available), the route planning module return the error message which will be displayed to the user.

2.3 Class diagram of Task assignment module

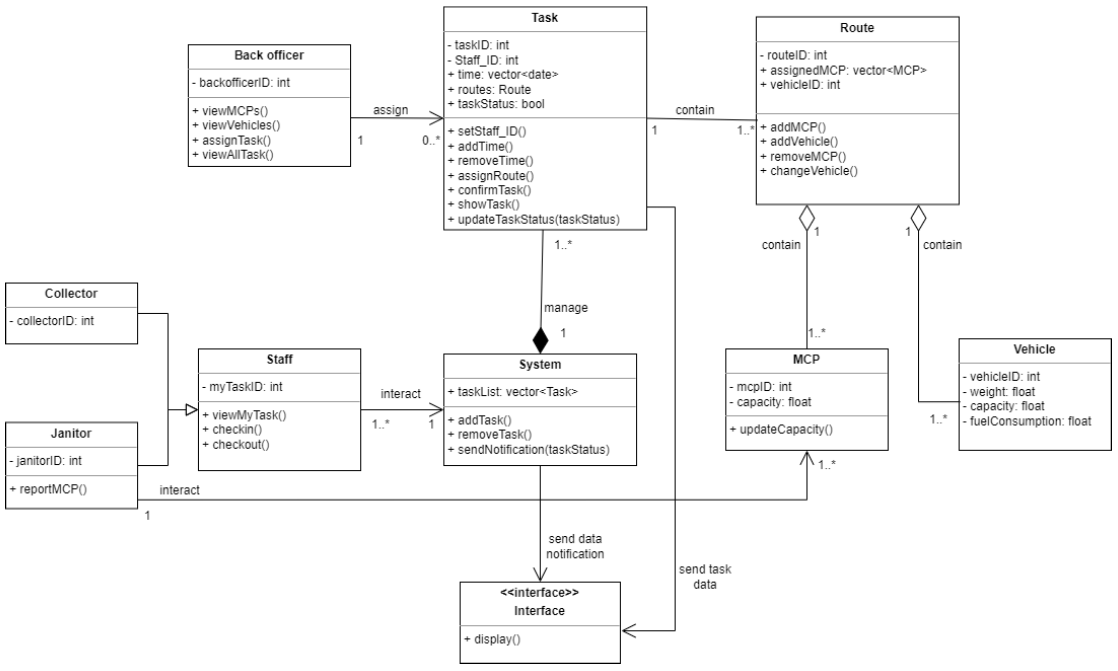


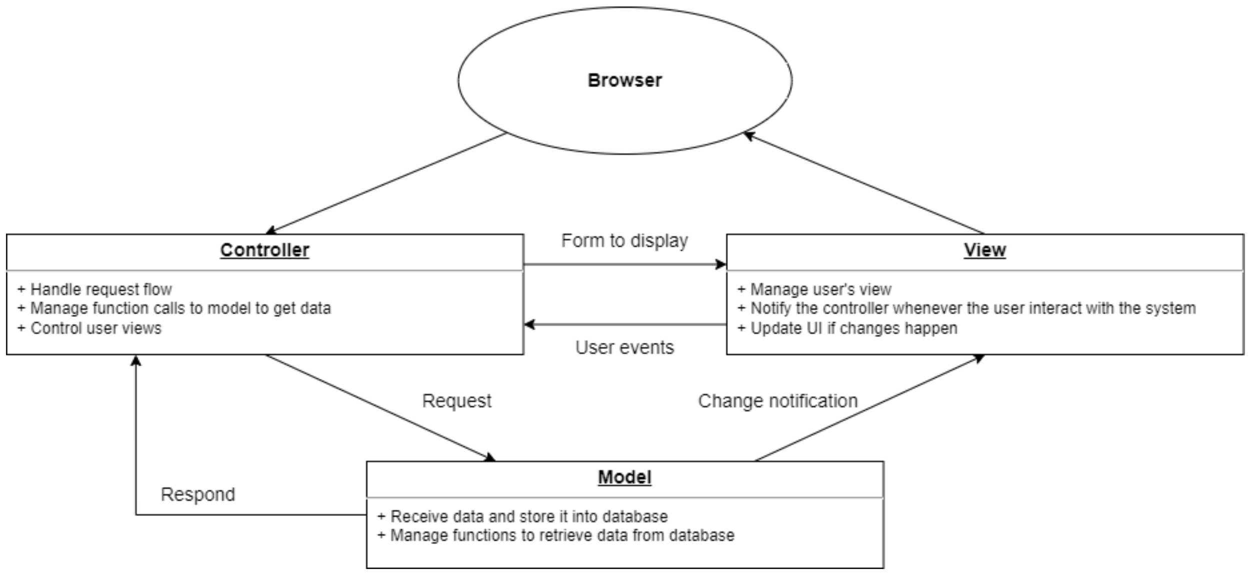
Figure 3: Class diagram of Task Assignment module

Description

- *Back officer* assigns tasks to *Collector* and *Janitor* by deciding calendar and routes. *Back officer* can have a overview of vehicles and all MCPs so that routes can be optimized in term of fuel consumption and travel distance.
- *Staff* (including *Collector* and *Janitor*) can view their tasks and check in through the interface, then the task status (*taskStatus*) is *False*. When work is finished, *Staff* checks out and *taskStatus* switches to *True*.
- *Janitor* reports to the system if a *MCP* is full and the capacity of *MCPs* are updated every 15 minutes.

3 Task 3: Architecture Design

3.1 Architectural approach and system modules of the system


Browser
Controller
+ Handle request flow
+ Manage function calls to model to get data
+ Control user views
Form to display
User events
View
+ Manage user's view
+ Notify the controller whenever the user interact with the system
+ Update UI if changes happen
Request
Change notification
Respond
Model
+ Receive data and store it into database
+ Manage functions to retrieve data from database

MVC pattern explanation:

View	- Manage user's view: • Manage UI • Switch between pages - Notify the controller whenever the user interact with the system: • Record user interaction • Send request to the controller to perform action - Update UI: • Update if receive response from the controller
Controller	- Handle request flow: • Receive request from view • Handle request queue - Manage function calls to model to get data: • Call the proper modules for the request • Call the functions in model corresponding to the request - Control user views • Send response to view for presentation
Model	- Receive data and store it into database: • Get input data from user (if needed) • Store the data - Manage functions to retrieve data from database: • Receive signal from controller for functions to operate • Execute the function to get the data

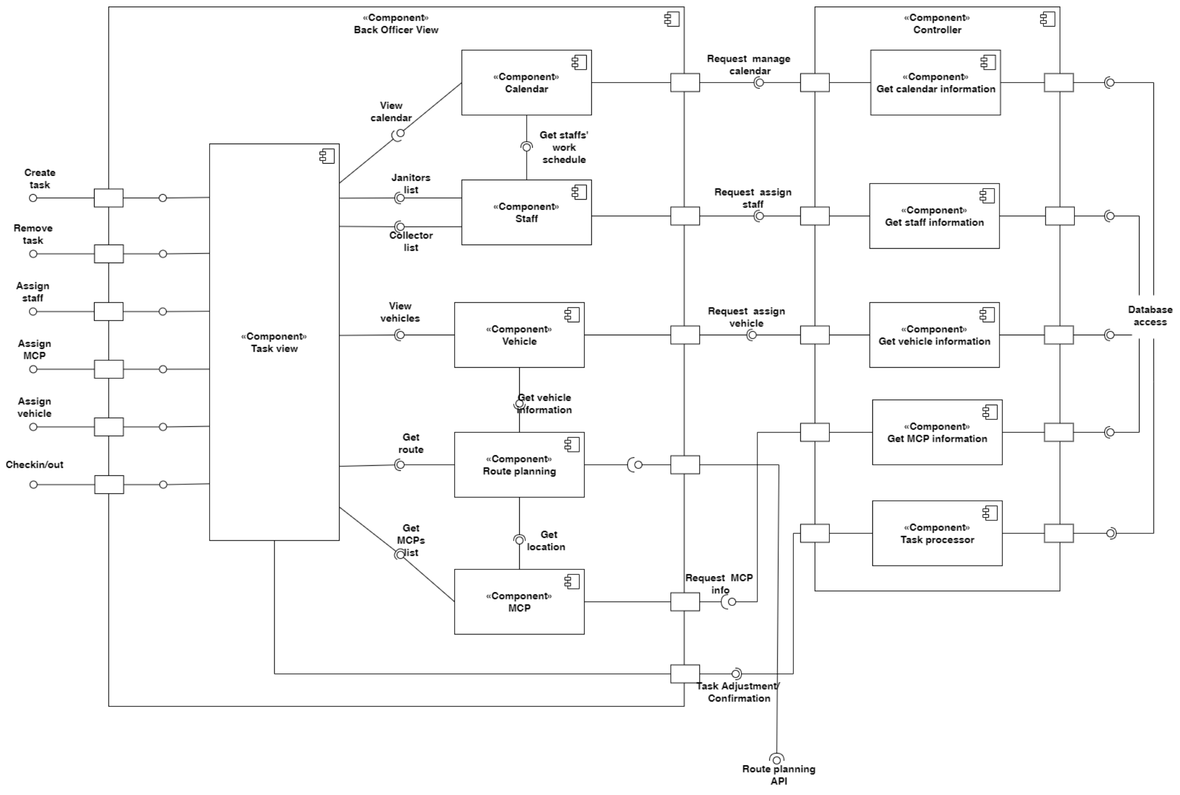
Module Description:
In our UWC 2.0 model, we plan to build 8 modules:

Module name	Description	Input	Output
1. Interface Display	+ Manage the UI, user's view at real time + Receive data from other modules for presentation	+ Data for presentation + UI files	The UI displaying on the screen
2. Interaction Controller	+ Receive user input and record interaction from user + Connect with other modules to receive the response for the request + Connect with Interface Display module to update user's view when changes happen	+ Data for inner functions + Interactions from users	Signals sent to other modules for operating
3. Authentication Management	+ Support the Interface Display module to present proper user's view + Receive and store information about the user identity (username, password, work function,...) whenever a new user sign up + Control the login and logout phase + Connect with the database to manage data	User data (username, password, work position,...)	+ Response to user authentication request (log in, log out, sign up) + Data for storing into database
4. Communication Module	+ Managing chats between users (janitors, collectors with back officers) + Control notifications from the system to the user (work time notification, update notifications)	+ Messages from user + Notifications from the system	+ Messages and notification for the user

5. Task Management	+ Let janitors view the work calendar, work position, information about their tasks + Let collectors view the work calendar, work position, information about their tasks, information about their vehicle and routes + Let back officers create calendar, create tasks, assign them and control the workflow of the day + Let the user check in and check out of their day work + Connect with the database to manage data	+ Request from the Interface Display module + Current user identity for corresponding services + Information about routes, vehicles,... from back officers	+ Notifications for the Communication Module + Data for storing into database
6. Information Management	+ Let users receive and record information other than tasks (collector's alternated route, janitor locations, MCP capacity...) for reference + Connect with the Interface Display module to present the external information + Connect with the database to manage data	Data from other sources	Data for storing into database for later reference
7. Data Formatter	+ Communicate between the database and other modules to extract the suitable data format for use and preprocess data before storing into the database	+ Raw data from the database when extracting data + Input data from other modules when storing data	+ Formatted data ready to be use + Preprocessed data ready to be stored into the database

8. Database	+ Store all the information for use (users, routes, vehicles, work calendar,...) + Retrieve data when needed	Data received from other modules	Raw data

3.2 Implementation diagram of Task Assignment module



4 Member Workload

No.	Full Name	Student ID	Problem	Evaluation
1	Le Nguyen Gia Nghi	1952868	- Architecture design and modules description	100%
2	Tran Do Dang Khoa	2053142	- Architecture design and modules description	100%
3	Tran Gia Han	2052464	- Draw component diagram of the system architecture	100%
4	Pham Quynh Tran	2053518	- Draw component diagram of the system architecture	100%
5	Do Truong Thinh	2052725	- Draw component diagram of the system architecture	100%

HO CHI MINH CITY, October 2022