HO CHI MINH UNIVERSITY OF TECHNOLOGY
COMPUTER SCIENCE AND ENGINEERING FACULTY

REPORT TASK 1-2-3

SOFTWARE ENGINEERING

Group name: **Nontech**

UWC 2.0

Teacher: Lê Đình Thuận

Họ và tên	**Mã số sinh viên**
Lê Duy Hào	2011142
Đỗ Thành Minh	2011610
Trần Minh Tuấn	2012360
Phạm Quang Sơn	1914973
Đặng Văn Vĩnh	1915962
Nguyễn Duy Hà	2011130

Mục lục

I. Introduction

In today's metropolitan cities, millions of people live packed together in densely populated areas and they all contribute to the tons of waste that end up in dumpsters daily. If someone tried to manually manage all that garbage, they would be absolutely overwhelmed; but with the help of municipal waste management software, they could collect the garbage with a fraction of the time and effort!

In order to avoid the previously mentioned scenarios, the responsible party for waste collection must adopt an effective method. The most efficient way to do this would be to turn to smart waste management and using municipal waste management software. Thanks to modern technology, the people responsible for waste collection are able to monitor and manage their devices, billing, inventory, and regulate safety standards. Additionally, by using municipal solid waste software, they could save lots of time, garbage truck fuel consumption, and other resources by measuring containers' fullness levels and not going to empty containers; pick the most optimal route for collecting waste around the city; determine the demand (for example, a country with warmer weather would require more frequent waste collection to avoid odor); as well as preventing visual, odor, noise, and water pollution.

Municipal waste management software lends a helping hand to the company that uses it in terms of reducing losses and increasing profits, and to the residents of a metropolitan area by making their lives easier. So, in its essence, waste collection software is more than just business: It is a service for a company, the people, and the environment!

Who are relevant stakeholders?

External stakeholders: Citizens, governments and organizations.

Internal stakeholders: Back officers, collectors and janitors.

What are their current needs?

- Governments and organizations: Find the best solutions to solve the problem of urban waste management.
- Citizens: Waste should be treated as quickly as possible.
- Back officers: There is a way to manage the garbage collection system effectively and quickly than the previous manual management.
- Collectors and janitors: It is possible to save time moving between garbage collection points, as well as to know which area of garbage is full to avoid the effort without gathering garbage..

What could be their current problem?

- Collectors and janitors:
- The optimal plan has not been found to be able to solve the problem of waste collection and management effectively.
- Spending a lot on garbage collection but not effectively.
- Citizens: Loping waste affects health, causes discomfort.
- Back officers:
- There is no optimal solution to the problem of assigning work schedules to collectors and janitors.
- It is not possible to optimize the movement of garbage trucks and it is impossible to know when the MCPs will be full.
- Collectors and janitors: Always working but efficiency in work is very low.

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

- Government and organizations:
- Can manage and operate the collection in the most detailed and reasonable way.
- Save more resources than before (from employees to money).
- Citizen: Improve people's lives through improving the quality of the surrounding living environment.
- Officer in charge:
- Can be more proactive in division of work for collectors and sanitation workers in a detailed and clear manner.
- Get an overview of all MCPs and information about their capacity.
- It is possible to create separate routes for each collector, in order to optimize fuel and travel distance.
- Collectors and janitors:
- Only collect at that MCP when it knows it's full, saving time traveling to check.
- It is possible to know your detailed tasks in a day, and even beyond a week.

II. Task 1

1. *Functions*

This section defines the functions that each position (back officer, collector, janitor) and the general system must be qualified to perform.

Role	Functional requirement
General System	- Import and use the database from the current system UWC 1.0, but make sure to contain these information below for task specification - Store database to manage information of employees: + General information (name, age, gender, address,..) + Work information: work calender, assigned vehicle, assigned MCPs and area, daily check in / check out information - Store database to manage vehicles : + Technical details: weight, capacity, fuel consumptions,.. + Assigned route
Back officer	- Have an overview of janitors and collectors, their work calendar - Have an overview of vehicles and their technical details - Have an overview of all MCPs and information about their capacity, with the information is being updated continuously - Assign vehicles to janitors and collectors - Assign janitors and collectors to MCPs - Create a route for each collector. Assigned route is optimized in term of fuel consumption and travel distance - Be able to send message to collectors and janitors
Collector and Janitor	- Have an overview of their work calendar - Have a detail view of their task on a daily and weekly basic - Be able to communicate with collectors, other janitors and back officers - Check in / check out task every day - Be notified about the MCPs if they are fully loaded

2. Non-Functions

This section defines the attributes or constraints that indicates how the system operates. They make our application or software run more efficiently and illustrates the system's quality.

These are several aspects of requirement:

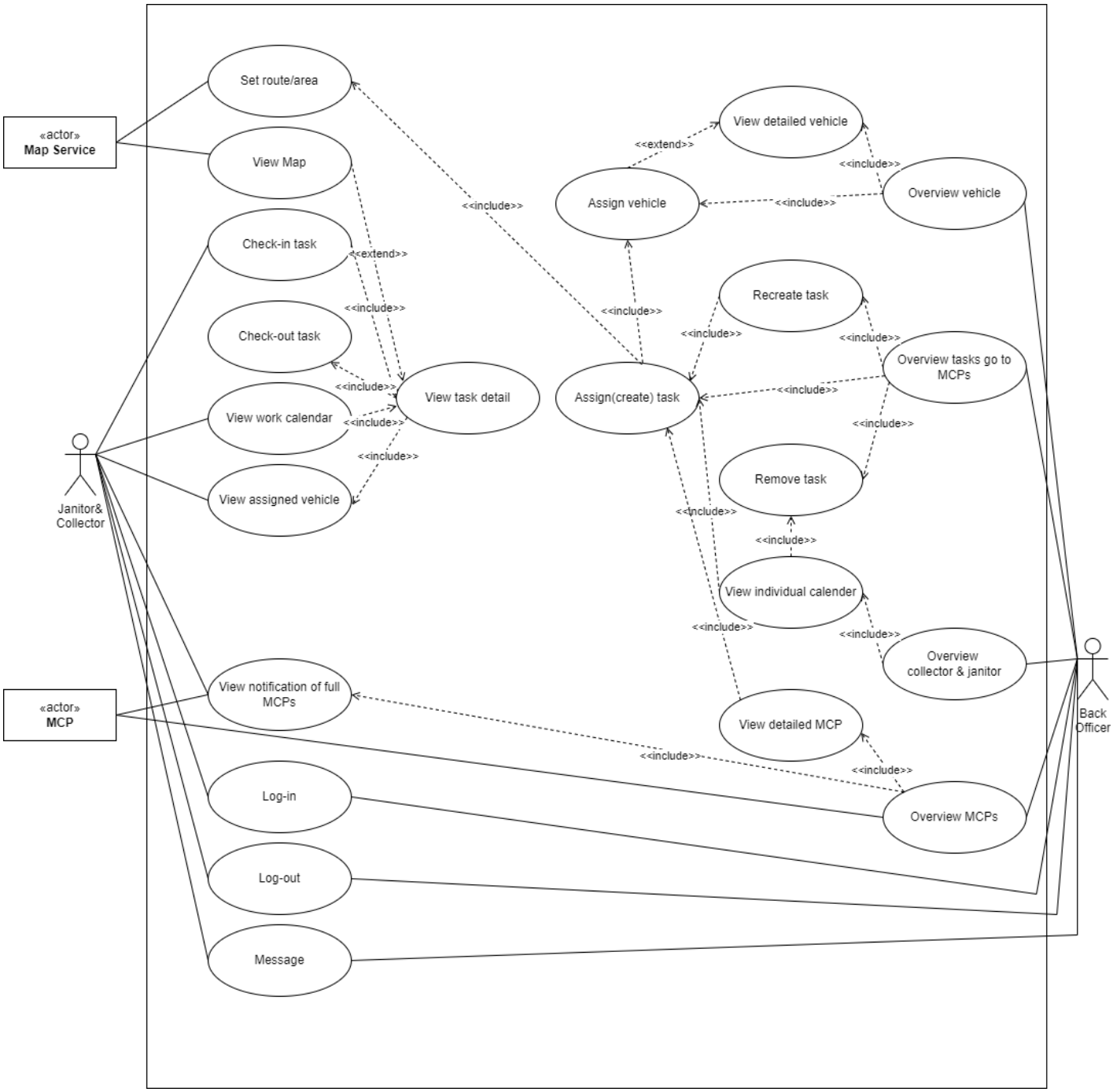
Criteria	Non-functional requirement
Reliability	- When problems occrur (such as system crash or power failure), the system must restore the previous state. - Record the description of problem: + Type of problem + Number of times occur + The time when problem occurs
Security	- Any users must have an account to log in - With the back officers, their power is bigger, so the security is more strict + Require stronger password when create account + After a certain number of fail login attempts, the system will lock this account + Only allow user to log in by one device of company office - With the janitors and collectors, whose knowledge about tech may be not good and the power is not important, the security requirement is easier
Speed	- Information of MCPs should be updated every 15 minutes with the availability of at least 95% of their operating time - Messenger engine among users makes sure delay less than 1 second - The notification to janitors when MCPs is full makes sure delay less than 1 second
Portability	- UWC 2.0 is expected to import and to use the existing data from UWC 1.0 - The Task Management is expected to be inter-operable with the UWC 1.0 as much as possible.
Usability	- When collectors and janitors view their task on a daily and weekly basic, all important information should be displayed in one view (without scrolling down) - The user interface (UI) should be simple and friendly as much as possible for user who is not familiar to tech can easily get used to.
Capacity	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
Localization	The system interfaces should be in Vietnamese, with an opportunity to switch to English in the future

3. Use case diagram

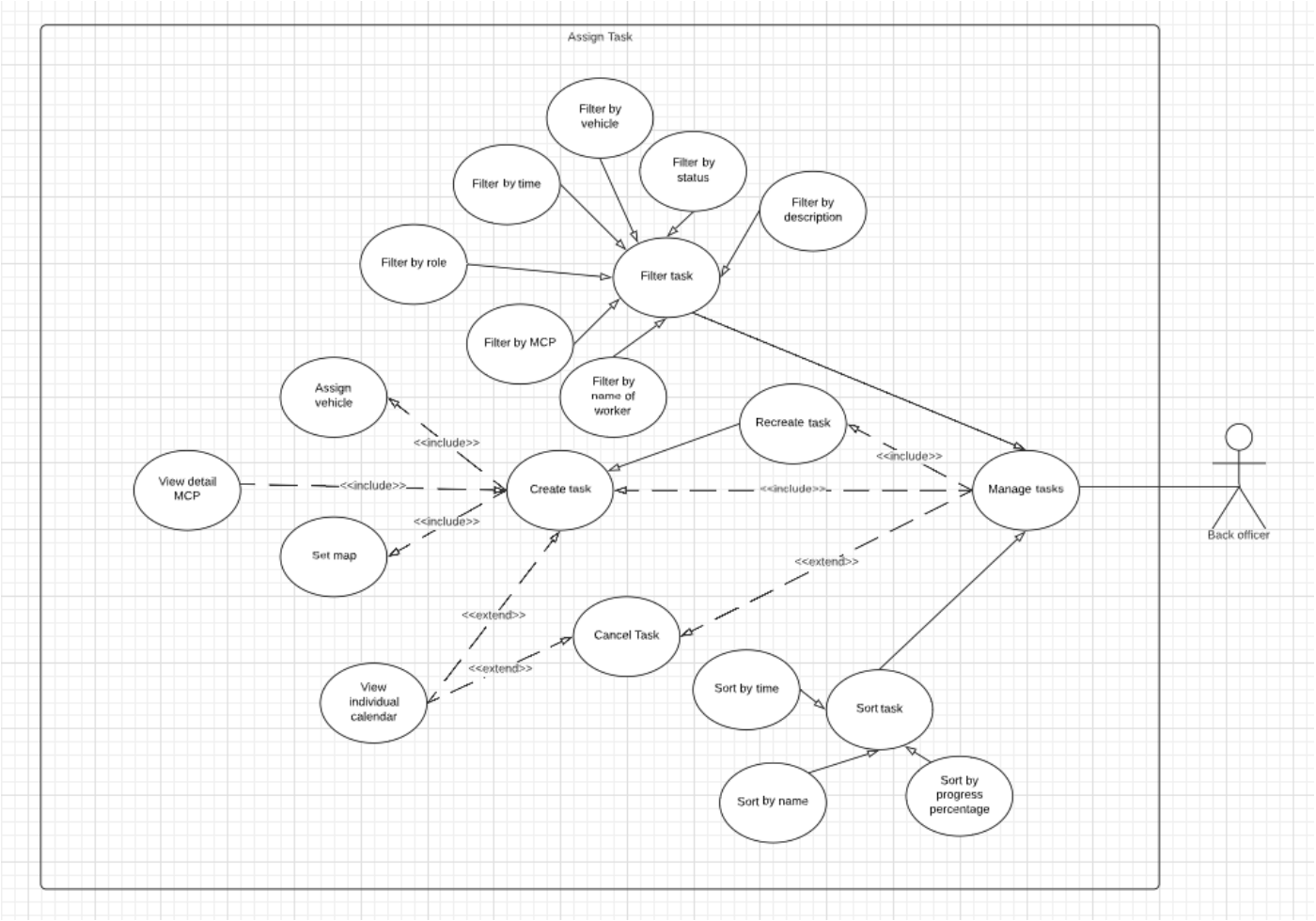
Table describes main use cases of the system

Use case name	General description
Log-in	Log in account registered on the system

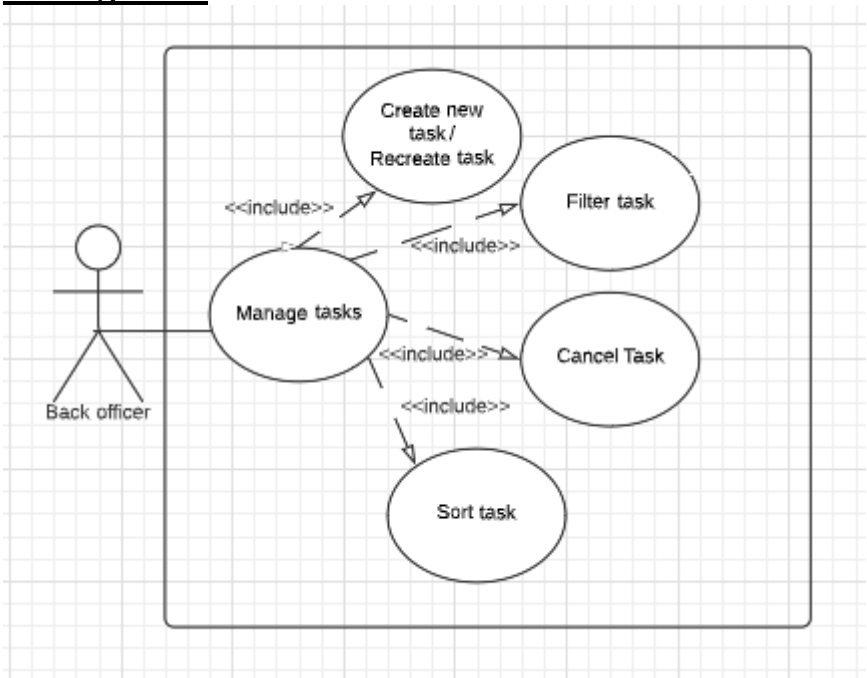
Log-out	Log out account registered on the system
Message	Communicate among the staff
Overview vehicle	View vehicles' detail and assign them for the collector/janitor
Overview task go to MCPs	- Overview calendar of all tasks - Create a new task, includes assign staff, vehicle and route - Cancel a task
Overview janitor/collector	Overview list of all janitors/collectors, their work calendars and assign task for them
Overview MCPs	Overview list of all MCPs and their details include storage status, position on map
View work calendar	View the schedule and detail of task has been assigned
View assigned vehicle	View which vehicle has been assigned to use for the tasks
View notification of full MCPs	Be notified when their assigned MCPs was full
View map	View all MCPs on the maps
Set route/area	Set a route for each collector
Check-in task	Check-in function for employee before work
Check-out task	Check-out function for employee after work
View task detail	Show detail information of employee's task
View detail vehicle	Show detail information of vehicle
Assign (create) task	Assign daily and weekly task to do for employee
Assign vehicle	Assign vehicle to use on duty to collector
Recreate task	Replace previously assigned task from employee with different task
Remove task	Remove task from employee
View individual	Show information of each employee


«actor»
Map Service
Set route/area
View Map
Check-in task
Check-out task
View work calendar
View assigned vehicle
View task detail
Janitor&
Collector
«actor»
MCP
View notification of full
MCPs
Log-in
Log-out
Message
View detailed vehicle
Assign vehicle
Overview vehicle
Recreate task
Assign(create) task
Overview tasks go to
MCPs
Remove task
View individual calender
Overview
collector & janitor
View detailed MCP
Overview MCPs
Back
Officer
<<include>>
<<extend>>

4. *Use case description Task assignment module*



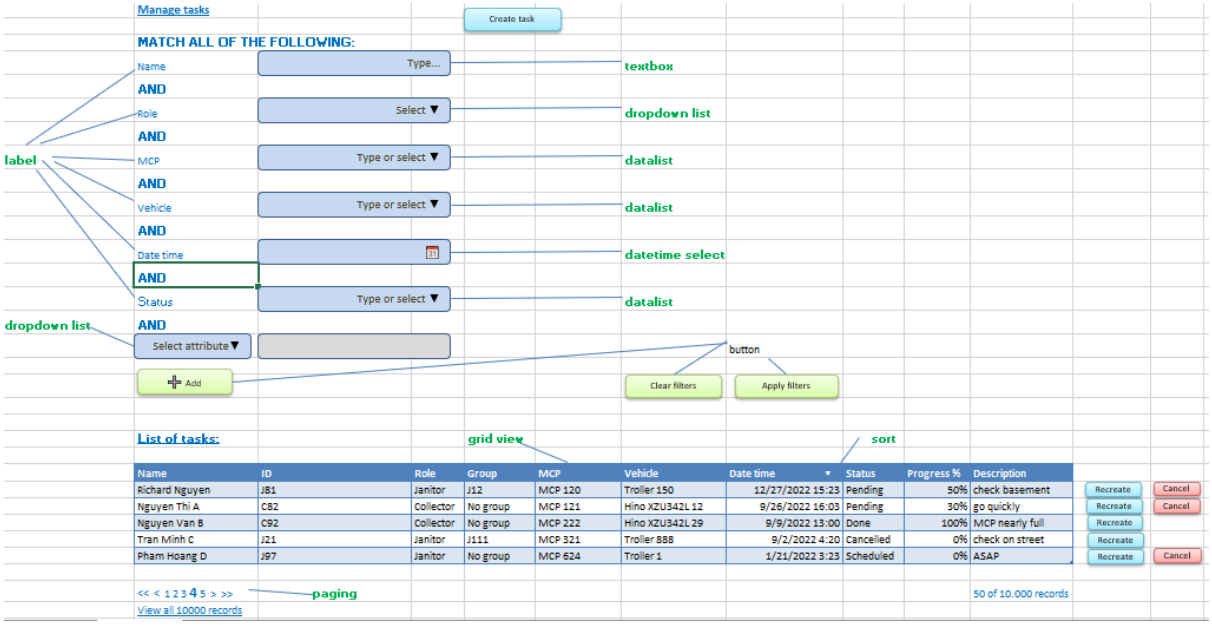
Manage task

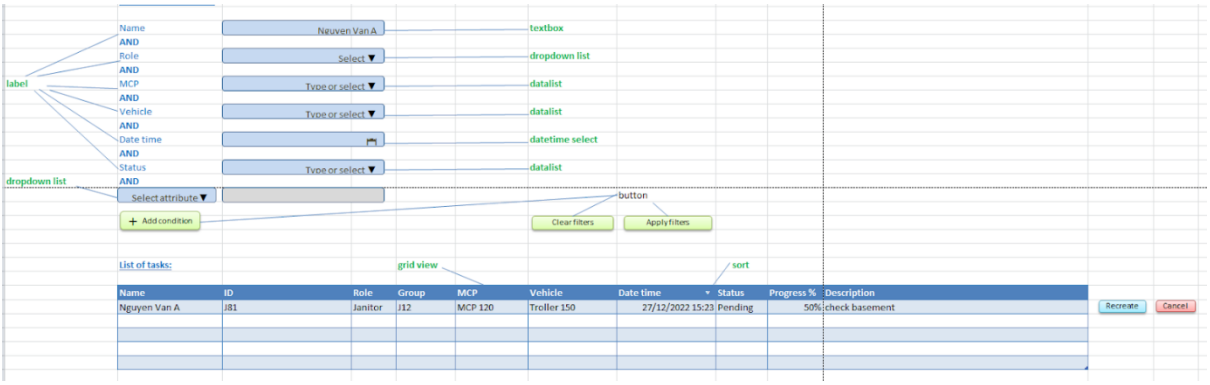


Use-case name	**Manage tasks**
Actor	Back officers

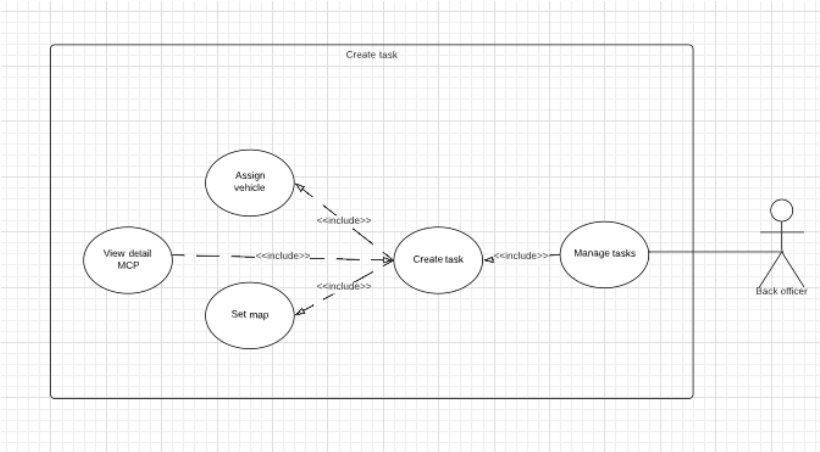
Description	Back officers overview all created tasks, be able to add, cancel, sort in the list of tasks
Trigger	Back officers click on the button “Task” to access the UI
Preconditions	Back officers login validly into the system with manager permission account to access back officers UI.
Normal Flow	1. UI displays tasks in spreadsheet, each task displays its detail 2. Back officers from the UI can perform functions such as add, cancel, filter, sort task to edit: a. Back officers click button “Create task” or button “Recreate” from existing task to perform action create a new task, the UI transform to ‘Create new task’ UI b. Back officers have option to “Cancel” with tasks have status ‘Scheduled” or ‘Pending’, UI pop-ups to confirm cancelation, after confirm turn back to manage task UI c. Back officers click button “Apply filters” to filter task d. In the list of tasks Back officers can sort task according to each column
Alternative Flows	Alternative flow 1: at step 2c - Back officers click “Clear filters”, task list turn back to sheet displays all tasks

Wireframe:





3.3.2 Create Task

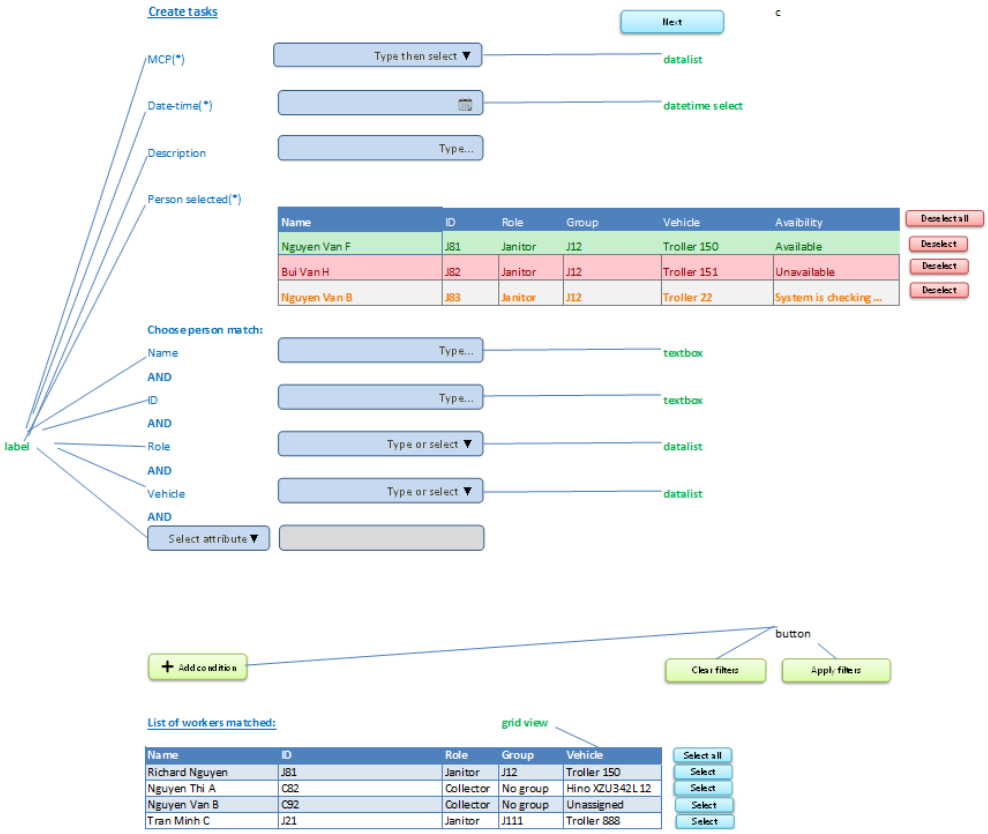


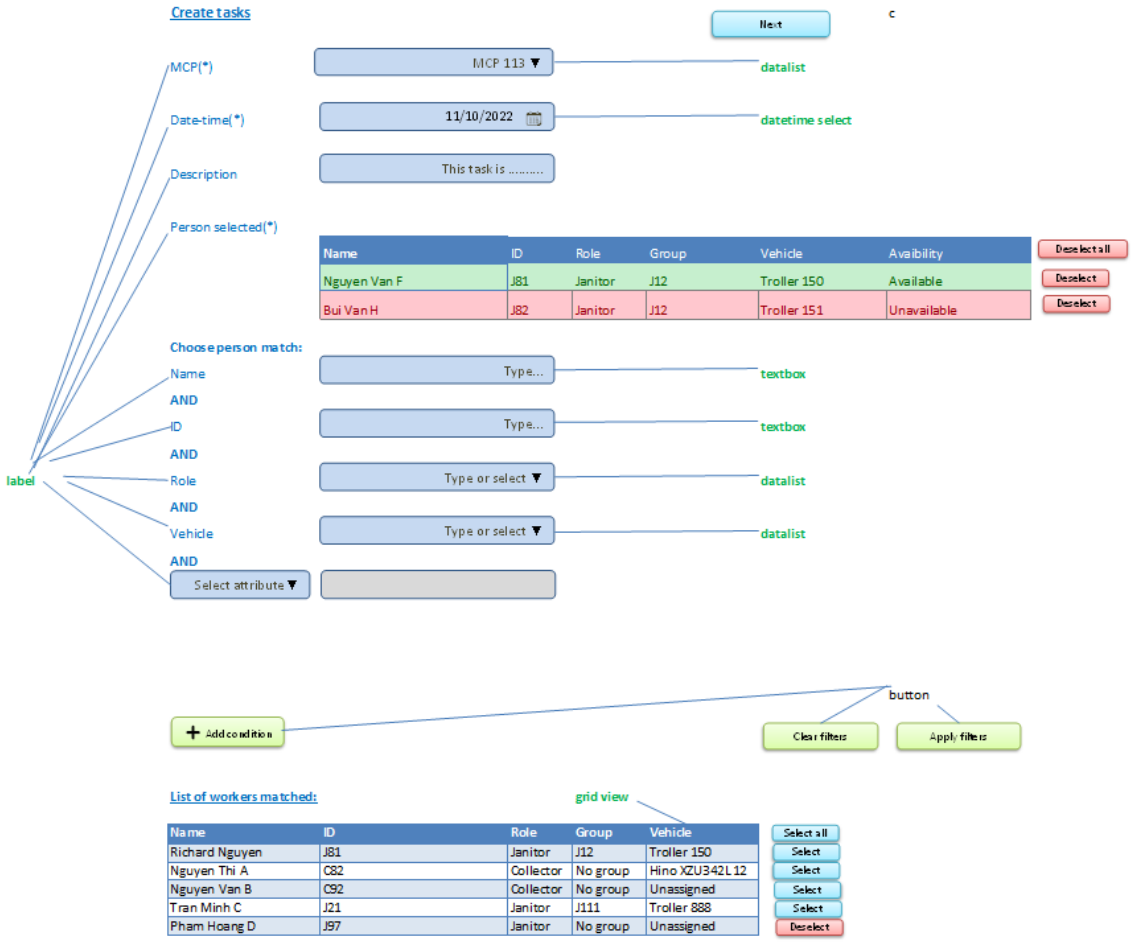
Use case name	**Create tasks**
Actor	Back officers
Description	Back officers can create a new task on the system.
Preconditions	Back officers must log in to the system with an account with management rights.
Trigger	Back officers click on “Create task” button on the Manage interface
Postconditions	A new task is created according to the information entered in the fields, the new task is displayed on the Manage interface.
Normal flow	1. The system opens the "Create task" interface. 2. Back officers fill in the information of the task that you want to create in the fields: MCP, Date-time, Description 3. The system displays a table “Person selected(*)” containing information about selected employees 4. The system has a filter to select employees with the following attributes text “Choose person match”. 5. Back officers fill in the information that you want to filter for employees in the fields: Name, ID, Role, Vehicle. 6. Back officers select properties to filter to find employees. 7. Back officers click on the "Apply filters" button to filter employees, the interface displays the employee table with the right attributes to filter. 8. Back officers click on “Select” to select the appropriate employee or “Select all” to select all, then the employee will be displayed on the “Person

	selected” panel, click on “Deselect” to remove the employee from the “List” panel. of workers matched”. 9. Back officers click on "Clear filters", the interface clears all information being filtered. 10. Back officers click on “Next” to complete task creation and switch to “Assign Vehicle” interface.
Exceptions	- Exceptions 1: step 2 If Back officers enter MCP that does not exist, the system displays “not found” under the field “MCP(*)” - Exceptions 2: step 8 If Back officers enter a role that does not exist, the system displays “not found” under the “Role” field. - Exceptions 3: step 9 If Back officers enter the MCP that does not exist, the system displays “not found” under the “Vehicle” field. - Exceptions 4: step 14 If Back officers enter missing information, when clicking "Next", the system displays "Insufficient information" at the bottom of the interface.
Alternative flows	No

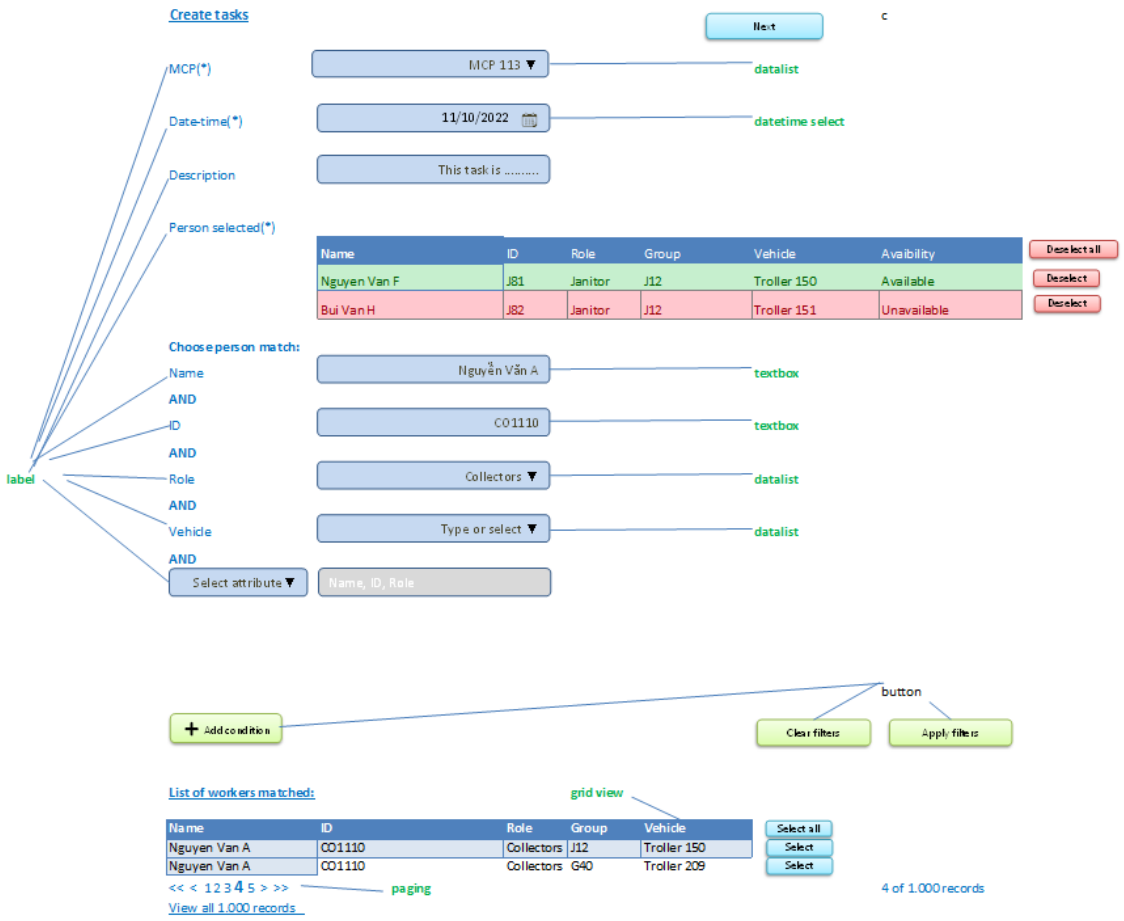
User story: As a Back officer, after receiving a task from my superior, I have a need to create a task to assign to employees. Therefore, I click on "Create task", fill in all the information in accordance with the actual task requirements, complete the system to record the new task and display it on the Manager interface.

Wireframe:


Create tasks
Next
c
MCP(*)
Type then select
datalist
Date-time(*)
datetime select
Description
Type...
Person selected(*)
Name
ID
Role
Group
Vehicle
Availability
Deselect all
Nguyen Van F
J81
Janitor
J12
Troller 150
Available
Deselect
Bui Van H
J82
Janitor
J12
Troller 151
Unavailable
Deselect
Nguyen Van B
J83
Janitor
J12
Troller 22
System is checking ...
Deselect
Choose person match:
label
Name
Type...
textbox
AND
ID
Type...
textbox
AND
Role
Type or select
datalist
AND
Vehicle
Type or select
datalist
AND
Select attribute
Add condition
button
Clear filters
Apply filters
List of workers matched:
grid view
Name
ID
Role
Group
Vehicle
Select all
Richard Nguyen
J81
Janitor
J12
Troller 150
Select
Nguyen Thi A
C82
Collector
No group
Hino XZU342L12
Select
Nguyen Van B
C92
Collector
No group
Unassigned
Select
Tran Minh C
J21
Janitor
J111
Troller 888
Select

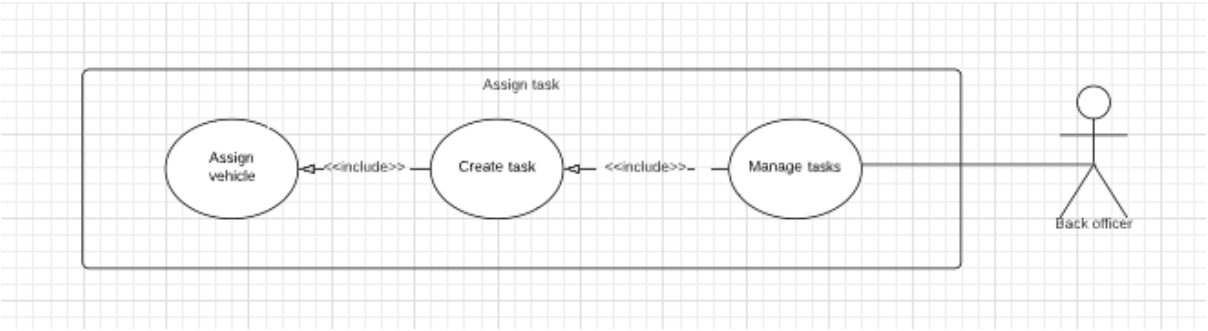


Deselect employee



Filter employee

3.3 Assign vehicle



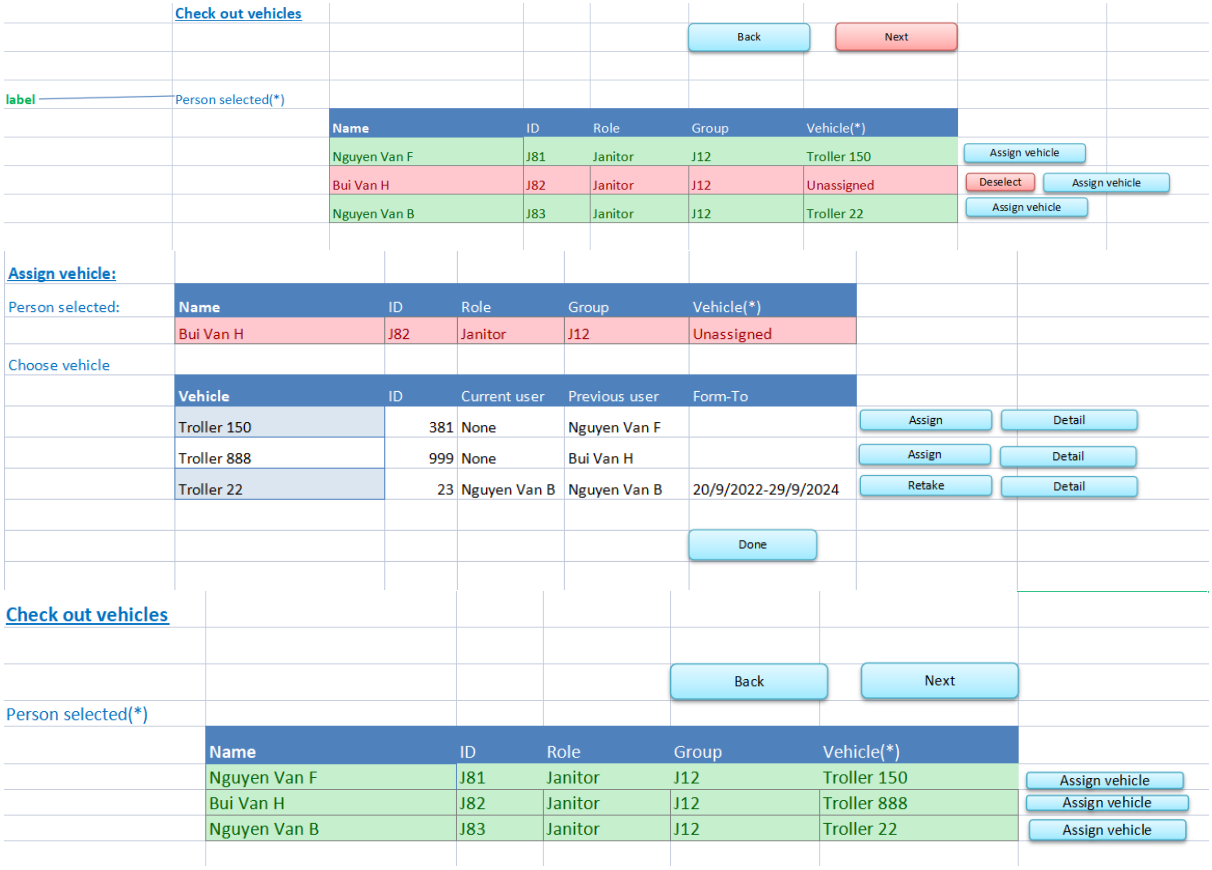
Use-case name	**Assign vehicle**
Actor	Back officers
Description	Back officers check assignment of vehicles to employees.
Trigger	Back officers click “Next” on the first page to move to the 2nd page: “Assign vehicle”.

Preconditions	Complete the content on the first page
Normal Flow	1. The interface switches to the 2nd page: “Assign vehicle”. Display the list of selected employee and their vehicles 2. Back officers click “Assign vehicle” next to the employee they want to assign. 3. Back officers select the vehicle they want to assign in the list of vehicles’ information appeared and press the “Assign” button to assign that vehicle to the employee they had chosen. 4. Press the “Done” button to complete the vehicle assignment. 5. The system updates changes and returns the employee list interface. 6. Back officers click “Next” to switch to the 3rd page: “Set map”.
Alternative Flows	- Alternative flow 1: at step 1 If Back officers do not want to continue assigning vehicles, click “Cancel” to return to the task list interface. - Alternative flow 2: at step 2 Back officers have not completed the assignment of vehicles (vehicle is “Unassigned”), then when clicking “Next”, the interface will display “No vehicle specified” next to the names of employees.

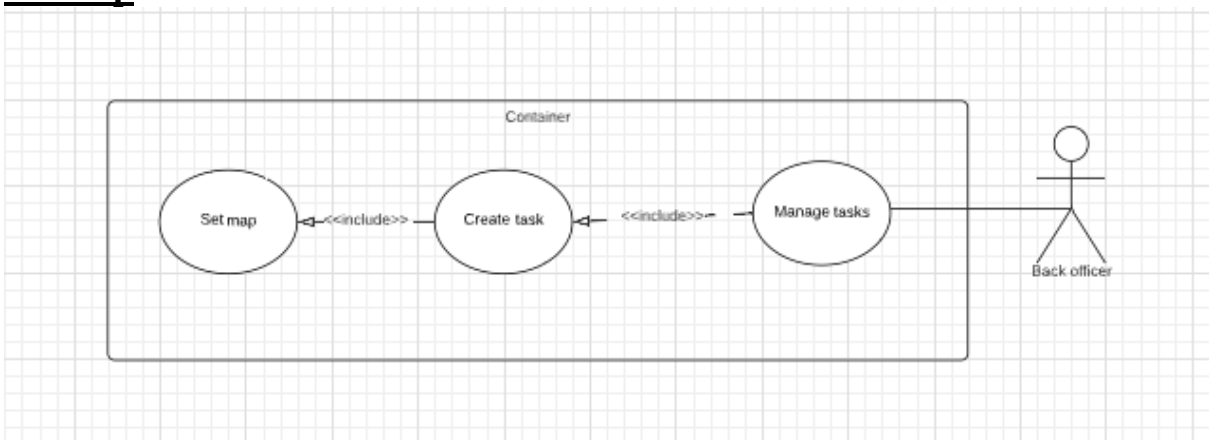
User story:

As a Back office, I want to assign vehicles to them after getting a list of my employees to manage. Therefore, I switched from the “Create task” page to the “Assign vehicle” page, selected the name of the employee I wanted to assign, and a list of existing vehicles appeared on the screen for me to choose from. After I made the selection, the system updated the employee's vehicle to the interface.

Wireframe:



Set map



Use-case name	**Set map**
Actor	Back officers
Description	Back officers set route for Collectors or choose area work for Janitors
Trigger	Back officers press the button “Next” at step 2: “Assign vehicle”.
Preconditions	The form in step 2 has been filled out all required information
Normal Flow	1. System goes to page step 3: “Set map”, there are 2 cases: a. If this task is set for Janitors, the page displays nearby areas to choose.

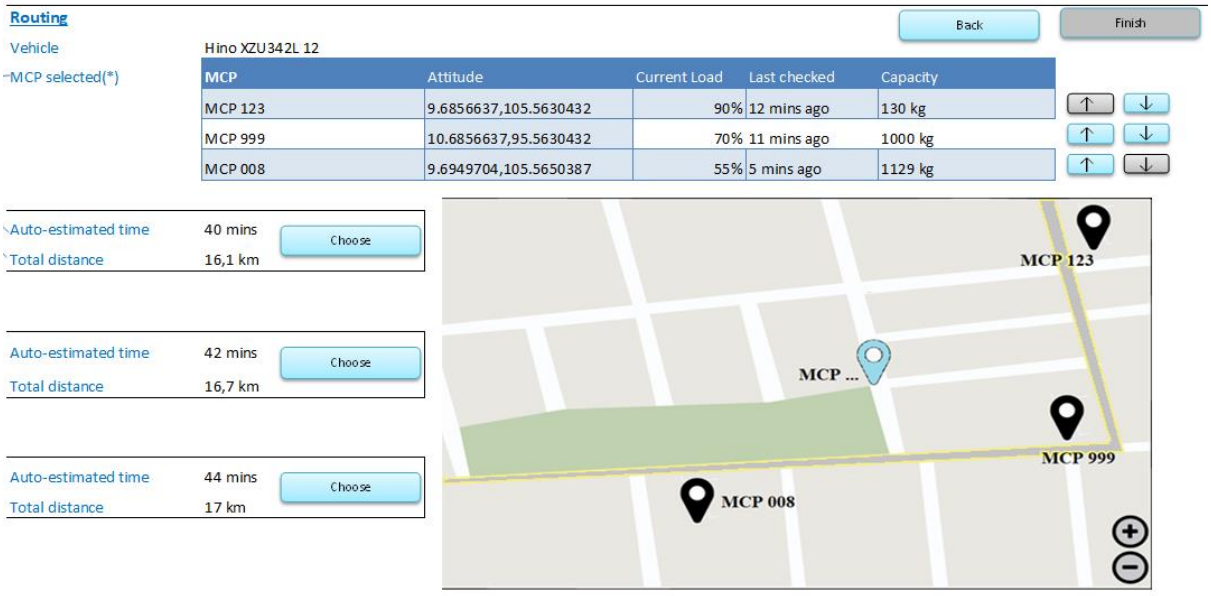
	b. If this task is set for Collectors, the page displays the list of MCPs selected and some predetermined routes with estimated time and distance. 2. Finish setting map, proceed to create task
Alternative Flows	- Alternative flow 1: at step 1 Back officer press Back, the app go back to step 2 “Set vehicle”

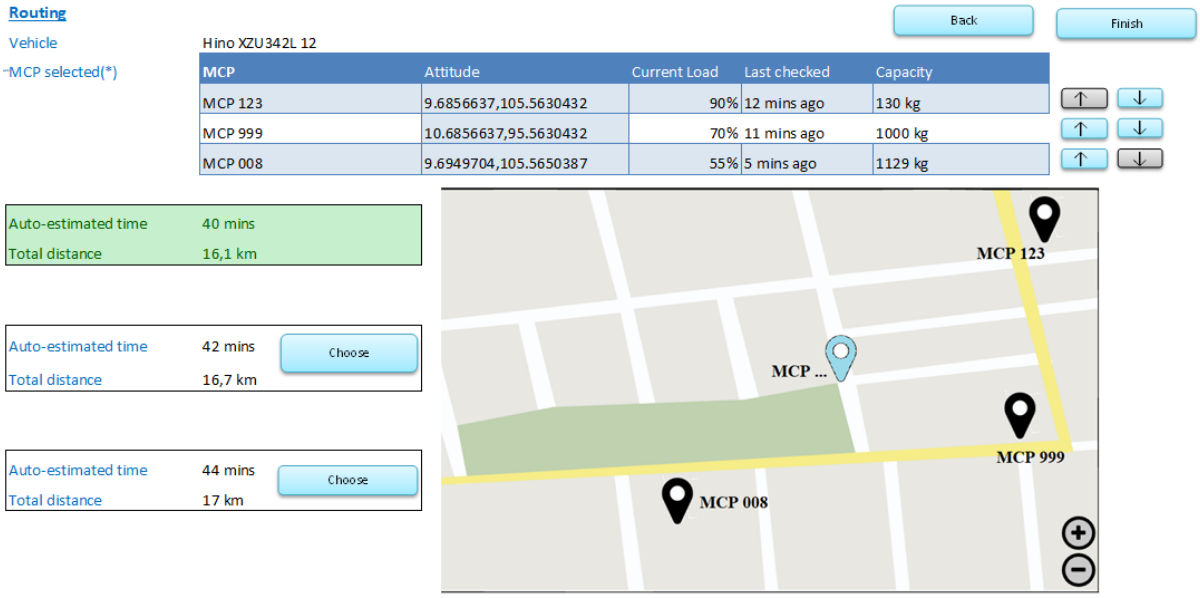
User story:

As a Back office, after getting a list of my employees to manage, I want to assign vehicles to them. Therefore, I switched from the “Create task” page to the “Assign vehicle” page, selected the name of the employee I wanted to assign, and a list of existing vehicles appeared on the screen for me to choose from. After I made the selection, the system updated the employee's vehicle to the interface.

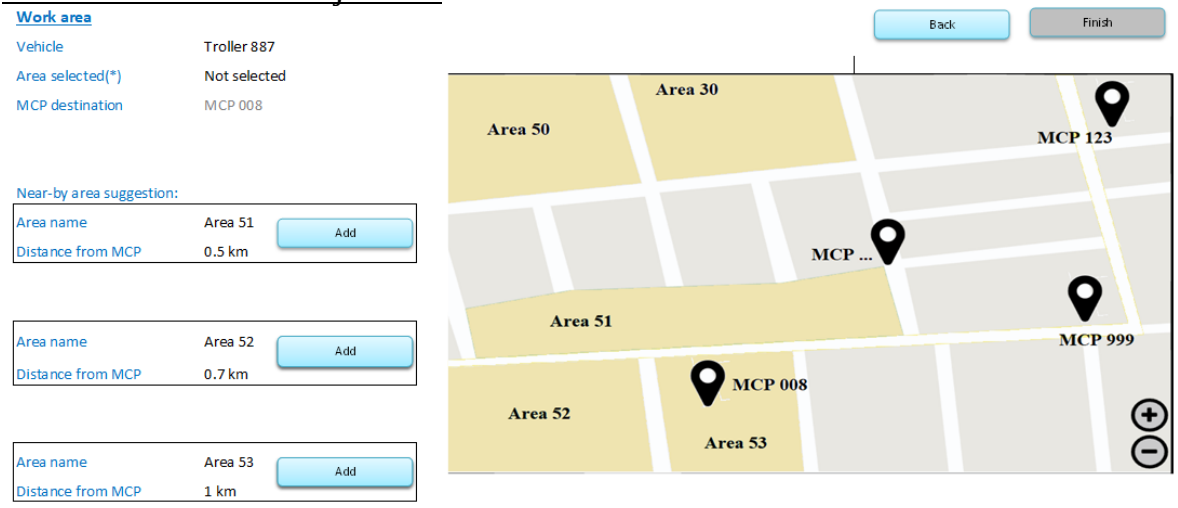
Wireframe:

Routing for collector:

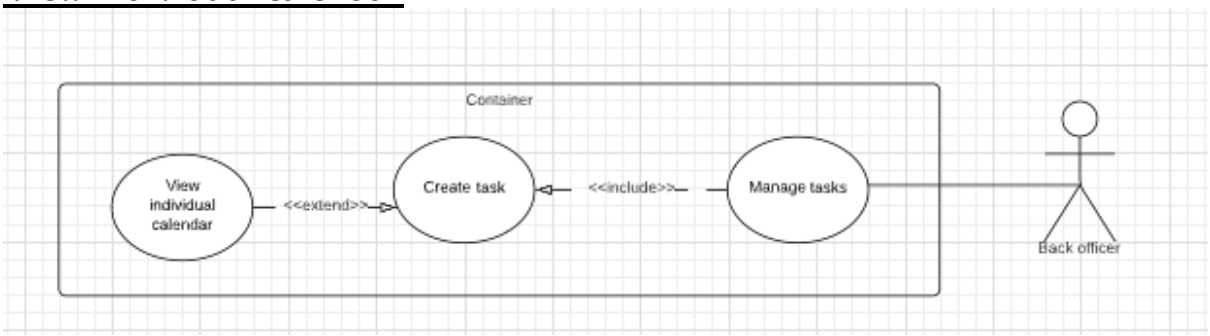




Choose work area for janitor:



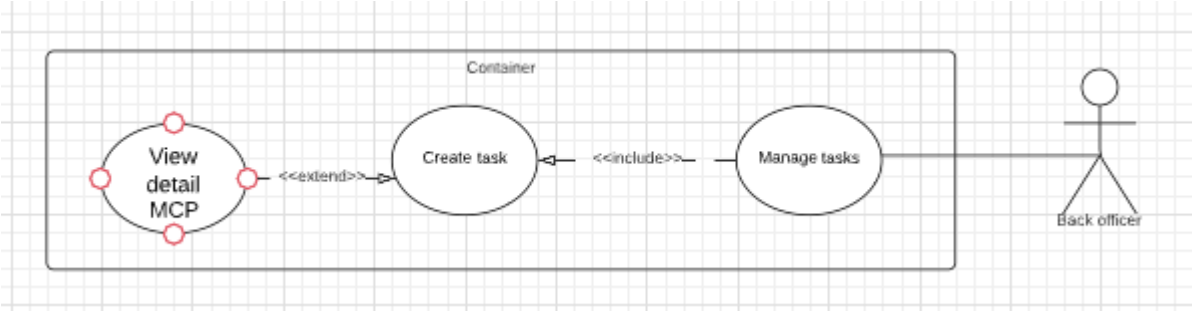
View individual calendar



Use-case name	**View individual calendar**
Actor	Back officers
Description	Back officers view the work calendar of a specific collector/janitor
Trigger	Back officers press button “View individual calendar” in the “Manage Collector/Janitor” page
Preconditions	No
Normal Flow	1. The app display the individual calendar in one view 2. Back officers press button “Create task” to create task for this individual 3. System proceed to go to “Create task”, the interface of create task is already filled the name of the individual. Back officers finish creating task and the system go back to the individual calendar. 4. At the individual calendar, back officers choose a specific task and press “Cancel task” to cancel the task assigned to that individual, after canceling, the system go back to the individual calendar
Alternative Flows	- Alternative flow 1: at step 3 Back officers press “Cancel”, the app goes back to the individual calendar

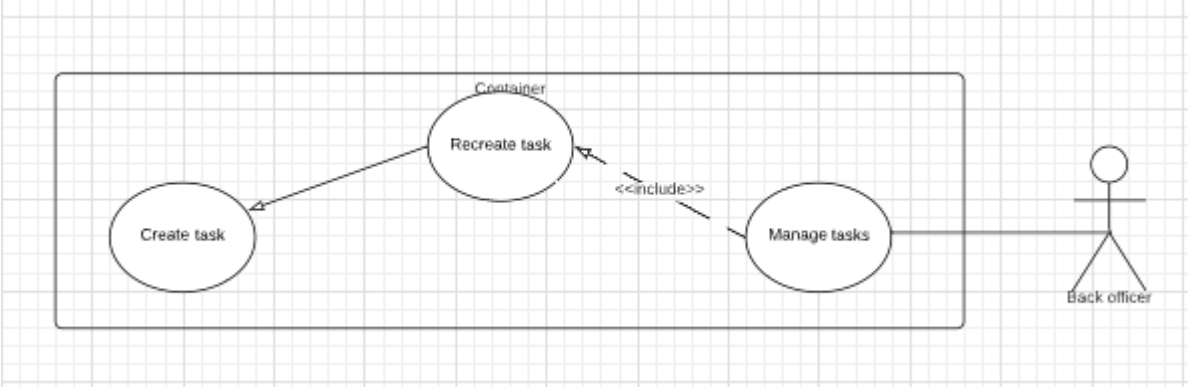
Wireframe:

View detail MCP



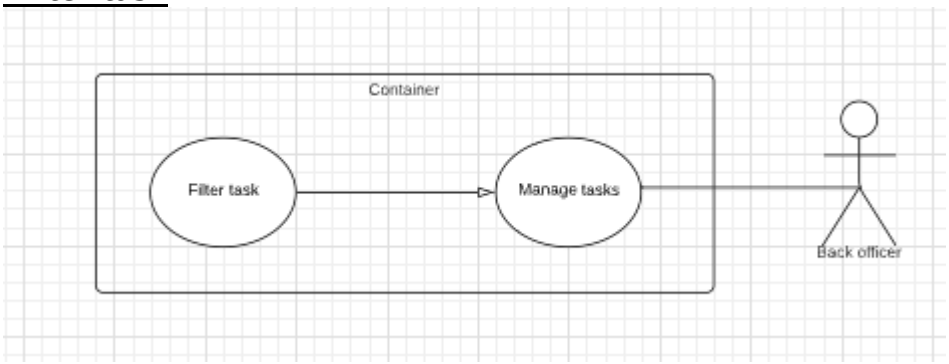
Use-case name	**View detail MCP**
Actor	Back officers
Description	Back officers assign work to a specific MCP
Trigger	Back officers press “MCPs” button
Preconditions	None
Normal Flow	1. The system lists all MCPs on user display. 2. Back officers select the search bar, type the needed MCP, then the system will come up with the result. 3. Click on MCP, the user interface will show the location of that MCP on the maps. 4. Back officers select “Create task”, the user interface display a new task whose MCP field have been filled in automatically. 5. To finish creating task, select “Finish”, then the user interface will go back to the maps with the location of the current MCP.
Alternative Flows	- Alternative flow 1: at step 2 Back officers clear the search bar, the user interface go back to the list of all MCP. - Alternative flow 2: at step 3 Back officers select “Cancel”, stop creating new task, the user interface go back to the maps with the location of current MCP

Recreate task



Use-case name	**Recreate task**
Actor	Back officers
Description	Back officers choose to recreate a task using the same information extracted from the task selected
Trigger	Back officers press “Recreate” button
Preconditions	Task is exist in the system database
Normal Flow	1. The interface displays the “Create task” interface with all completed information fields similar to the selected task recreate. 2. Back officers make information corrections as desired. 3. Click “Finish” to finish creating a new task, the interface returns to the task list.
Alternative Flows	- Alternative flow 1: at step 1 Back officers press “Cancel” button, stop creating new tasks, and return to the task list interface task.

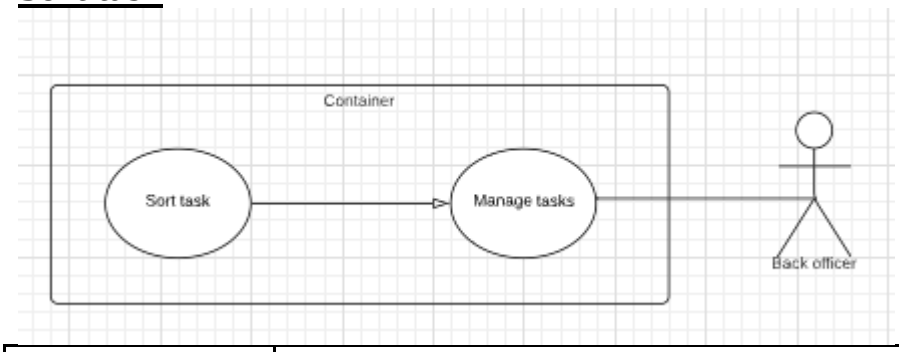
Filter task



Use-case name	**Filter task**
Actor	Back officers
Description	Back officers filter tasks with information to find
Trigger	Back officers press “Filter” button
Preconditions	No
Normal Flow	1. The interface displays a list of filterable fields, including: a. Description b. Status c. Vehicle d. Time

	e. Role f. MCP g. Name of worker 2. Back officers select one of the fields, enter the information to be filtered, click “Filter”. 3. The system filters on demand and the interface displays a list of tasks with information Back officers need to find.
Alternative Flows	- Alternative flow 1: at step 3 Back officers delete filtered content, press “filter” to return to the filterable fields list interface. - Alternative flow 2: at step 3 Back officers enter the filter content that does not exist, the interface does not display any task, shows the text “No task”.

Sort task

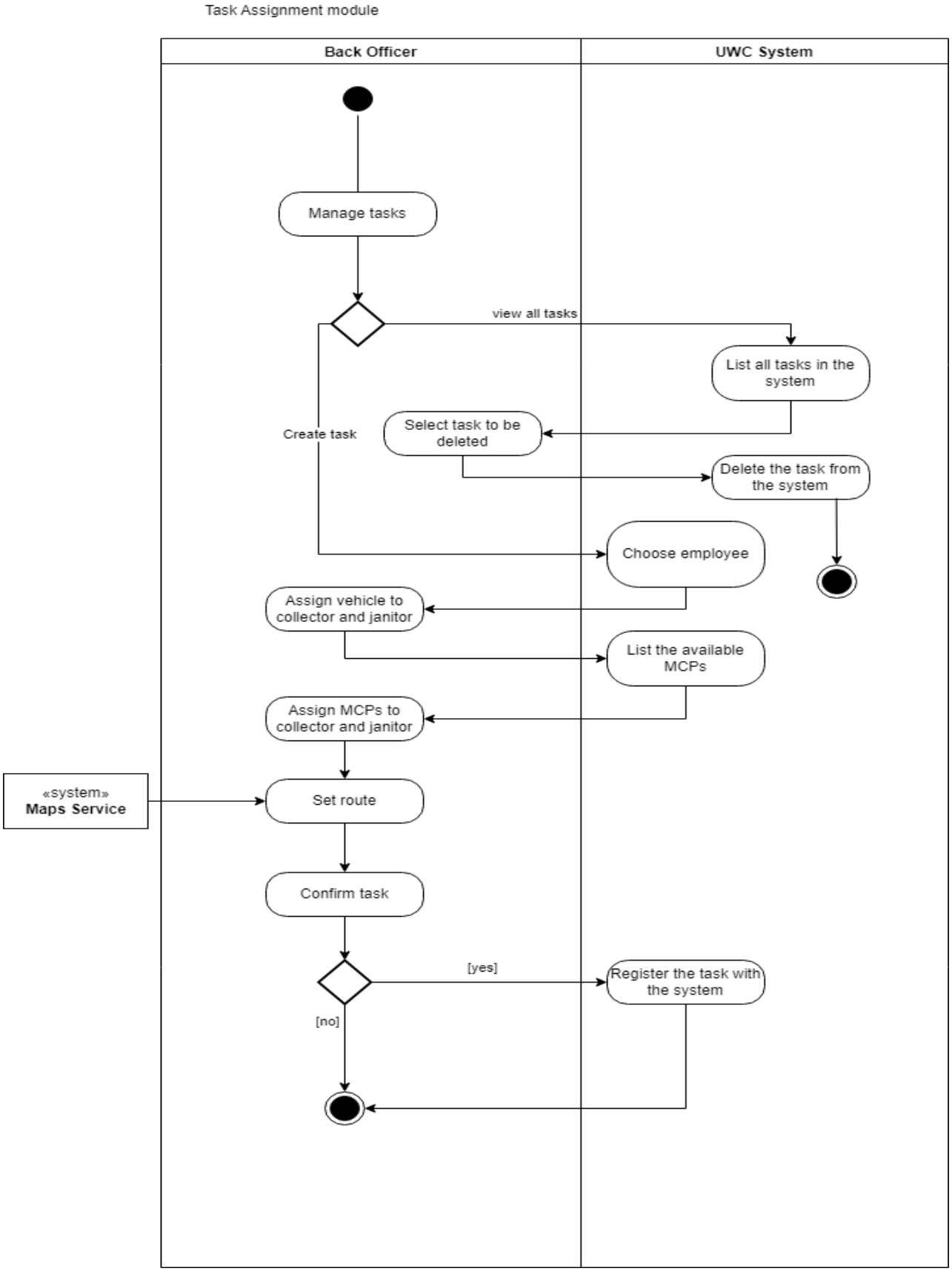


Use-case name	**Sort task**
Actor	Back officers
Description	Back officers want to organize tasks that have the information they need to find
Trigger	Back officers click “Sort”
Preconditions	No
Normal Flow	1. The interface shows a list of fields that can be sorted, including: a. Time b. Name c. Progress percentage 2. Back officers select one of the fields, enter the information to be filtered, and then click the “Sort” button. 3. The interface displays all tasks sorted by the field selected by Back officers. 4. Back officers arrange the jobs in either ascending or descending

	order by pressing "A-Z" or "Z-A."
Alternative Flows	- Alternative flow 1: at step 3 Back officers delete the filtered content and then click “Filter” to return to the interface of the list of sortable fields.

III. Task 2

2.1.Activity diagram:

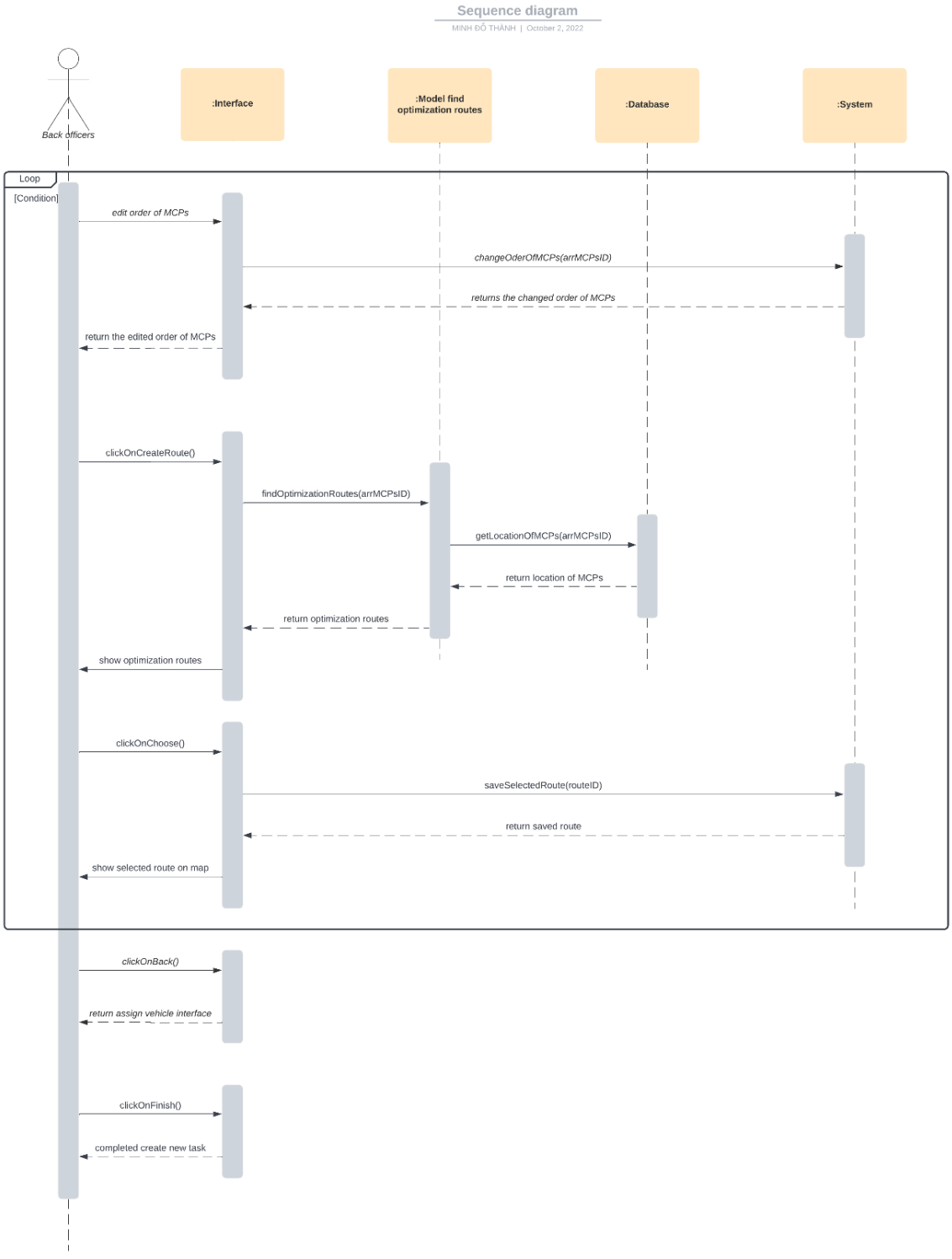


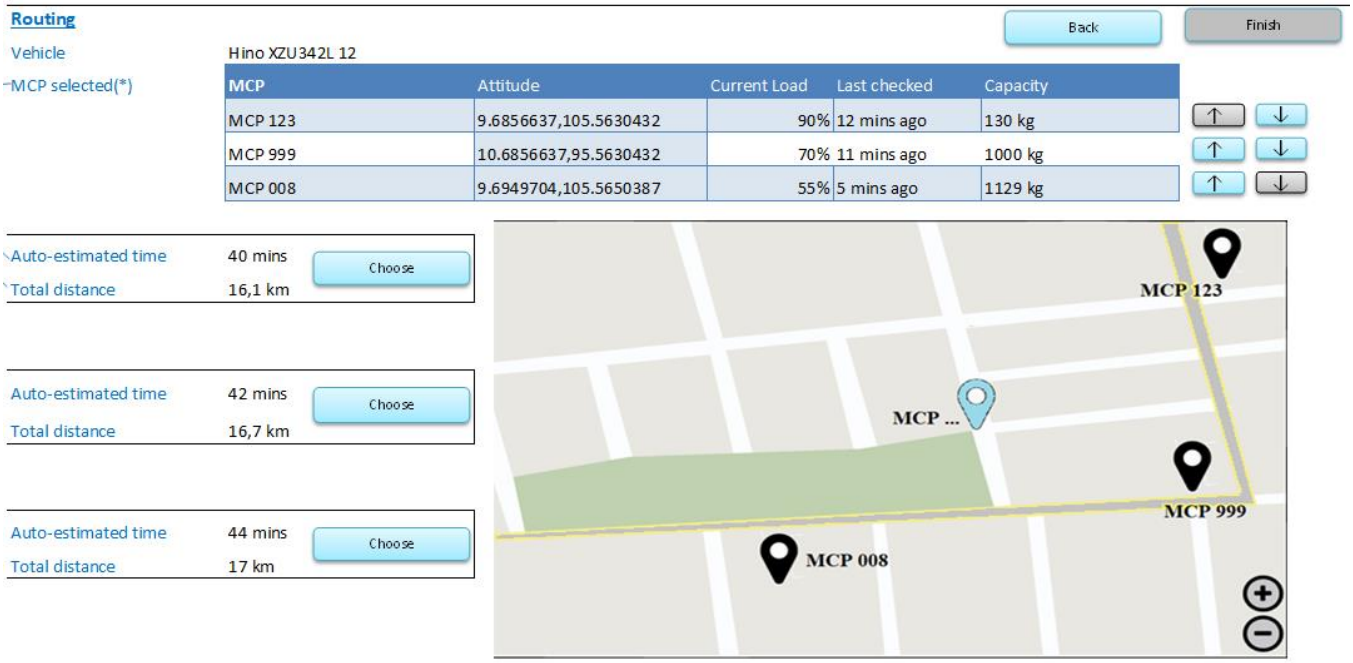
2.2: Route planning task.

Proposal: With the desire to be able to offer a fast route, saving time, effort and money, and at the same time minimizing the impact on the environment when using petrol-powered transportation vehicles to collect waste at MCPs, the team proposed export solution for "Create route" feature.

To create convenience and reduce effort, the team will take advantage of services provided by 3rd parties (maybe Google Maps, HERE WeGo, MAPS.ME, ...) to select destinations that are MCPs for the type of vehicle desired by the user. From there, the system runs algorithms to calculate the optimal route and display it for the user to choose.

Sequence diagram:


Sequence diagram
MINH ĐỖ THÀNH | October 2, 2022
Back officers
:Interface
:Model find optimization routes
:Database
:System
Loop
[Condition]
edit order of MCPs
changeOderOfMCPs(arrMCPsID)
returns the changed order of MCPs
return the edited order of MCPs
clickOnCreateRoute()
findOptimizationRoutes(arrMCPsID)
getLocationOfMCPs(arrMCPsID)
return location of MCPs
return optimization routes
show optimization routes
clickOnChoose()
saveSelectedRoute(routeID)
return saved route
show selected route on map
clickOnBack()
return assign vehicle interface
clickOnFinish()
completed create new task



Description:

- Back officers through the "Routing" interface to select the optimal path for Collectors through pre-selected MCPs.
- The "Routing" interface displays the MPCs selected in the "Create Task" interface with the following attributes: MCP ID, Attitude, Current Load, Last checked, Capacity
- The back officer can change the order of MCPs that Collectors must follow by:
  + Click “↓” to swap the order of the same row and bottom MCP.
  + Click “↑”to swap the order of the MCP of same row and row above.
- The changed order of MCPs is sent to the system, which processes and returns the order interface of the changed MCPs.
- Back officers click “Create route” to create routes through all selected MCPs. At this point, the software switches to a third-party service, retrieves the location of the MCPs from the database to calculate, returning the routes through all the MCPs.
- Back officers click “Choose” to choose the most optimal route among the proposed routes, the system proceeds to save the selected route, the interface displays the route on the map.
- Changing the order of MCPs, calculating and choosing the optimal path for the Collectors can be repeated many times.
- Back officers can click “Back” to switch to the “Assign Vehicle” interface.
- Back officers click “Finish” to switch back to complete the task of creating a new task.

2.3. Class diagram

2.3.1. Log in

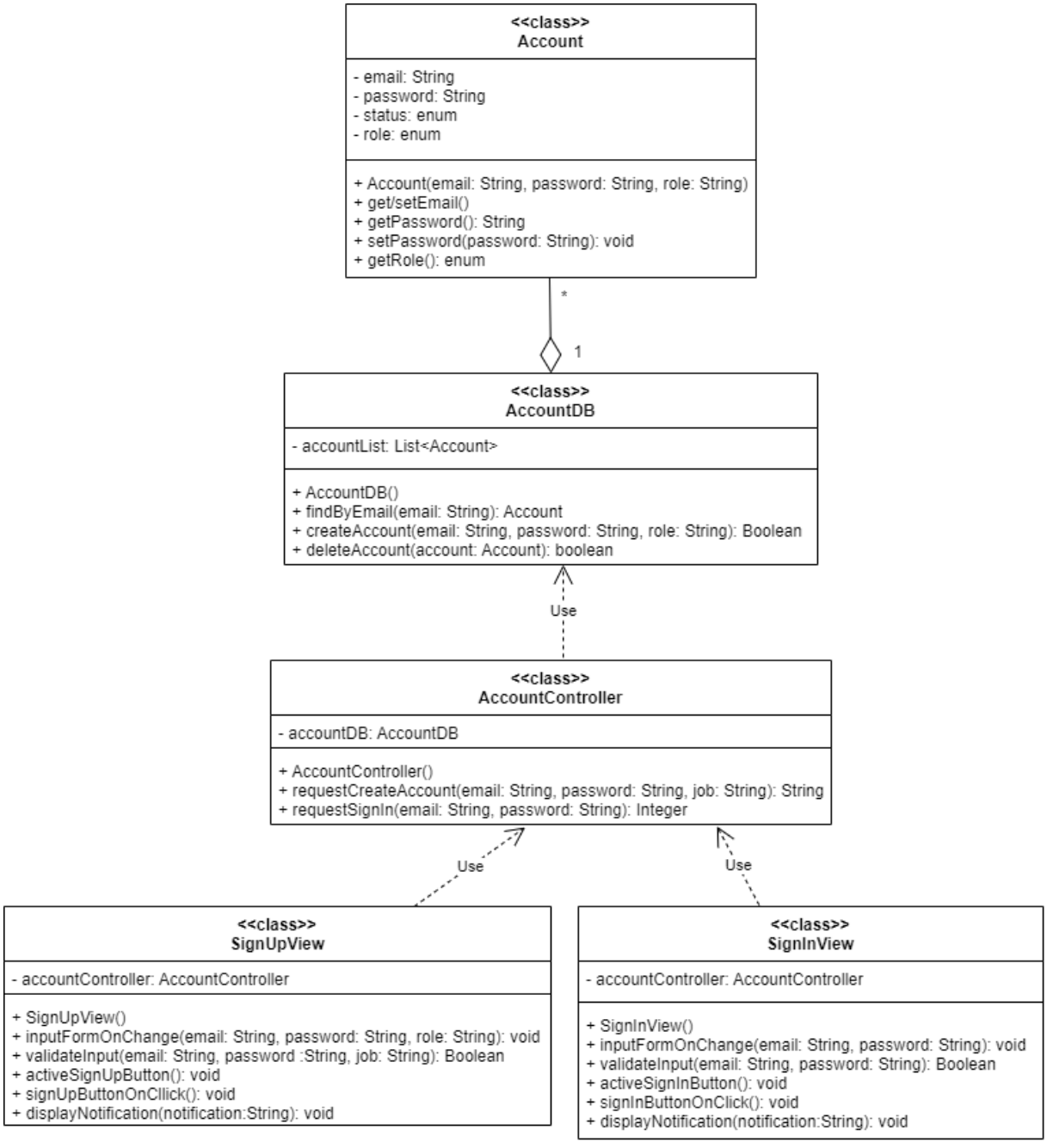


Diagram 1. User Sign in and Sign up Activity

2.3.2. Employee management

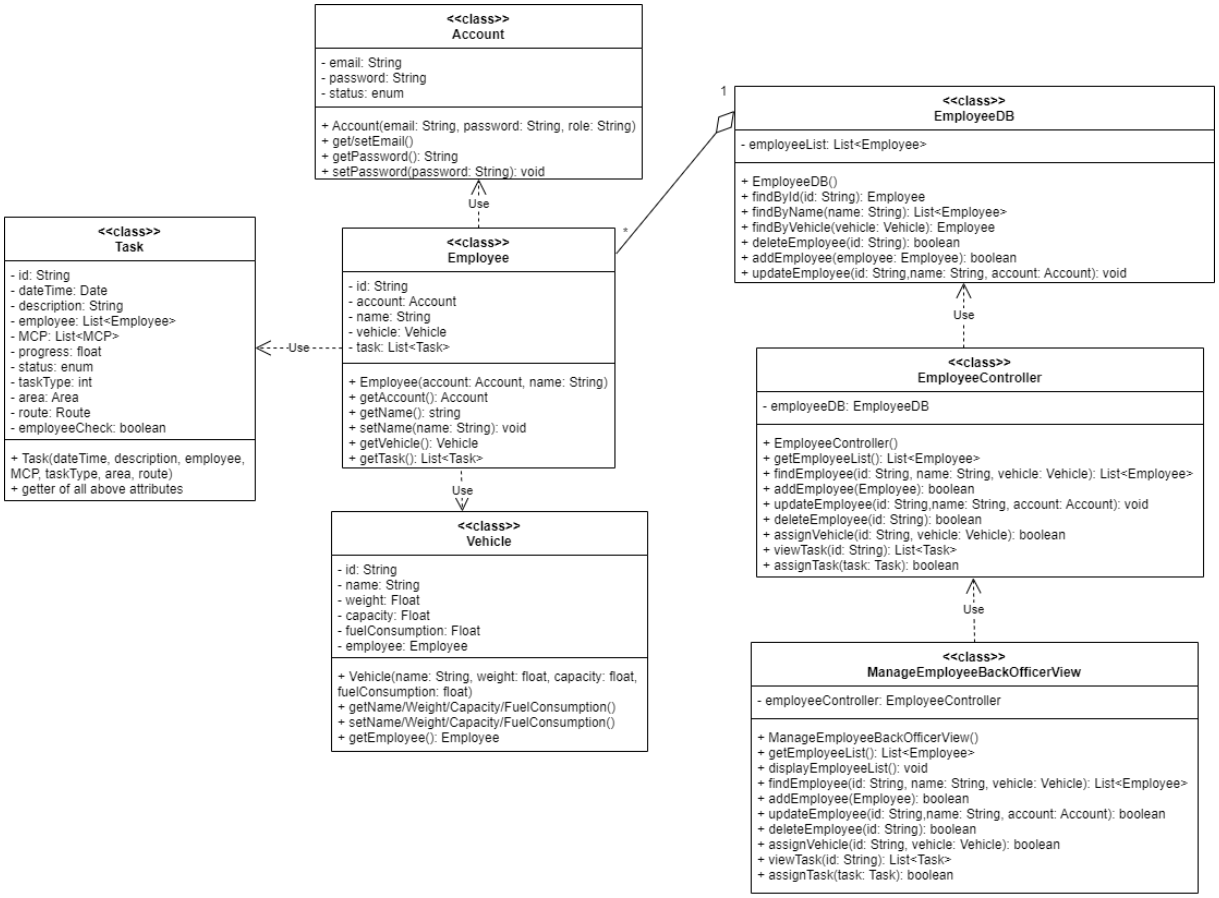


Diagram 2. Back officer Manage employee Activity

2.3.3. Vehicle management

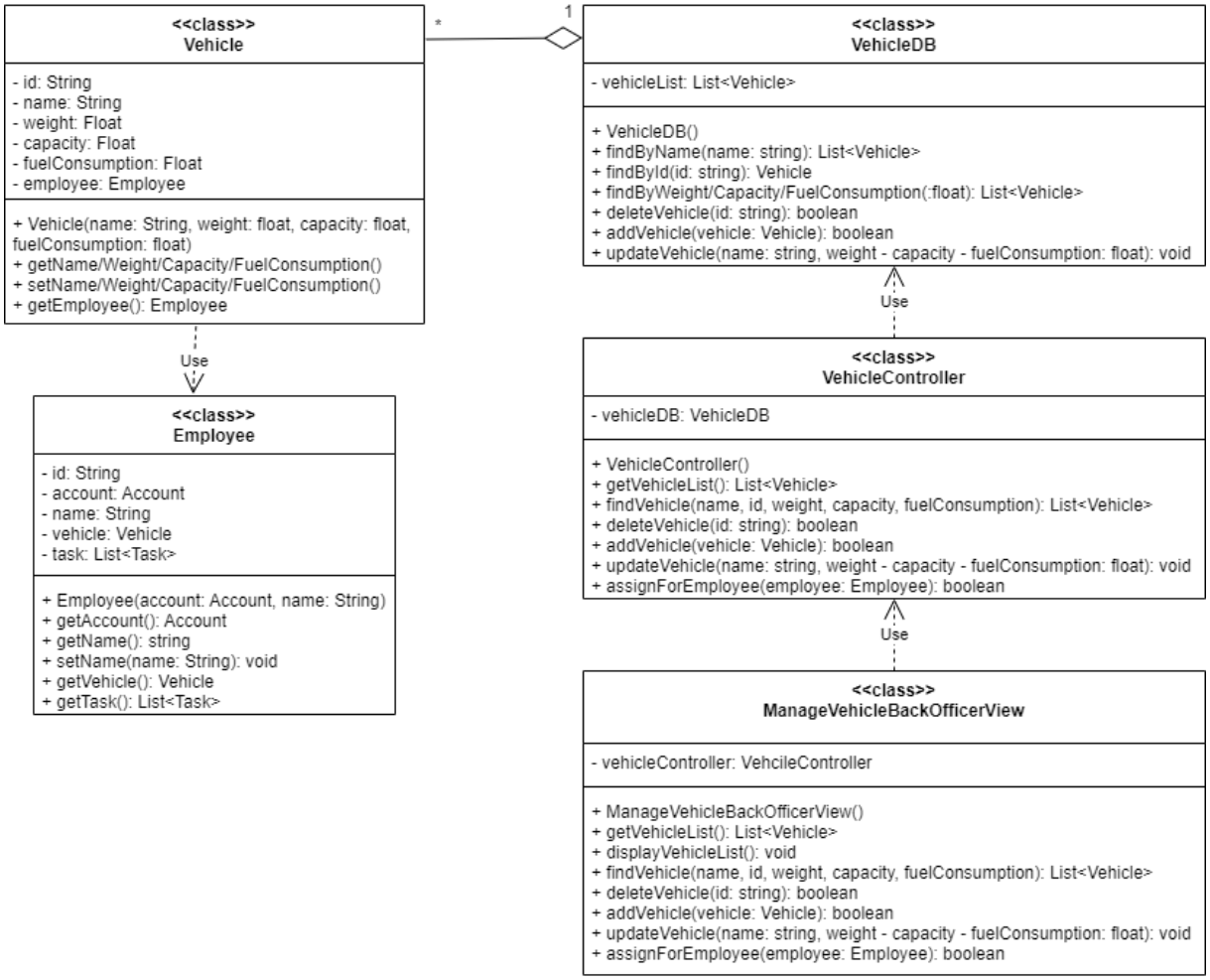


Diagram 3. Back officer Manage vehicle Activity

2.3.4. Task Management

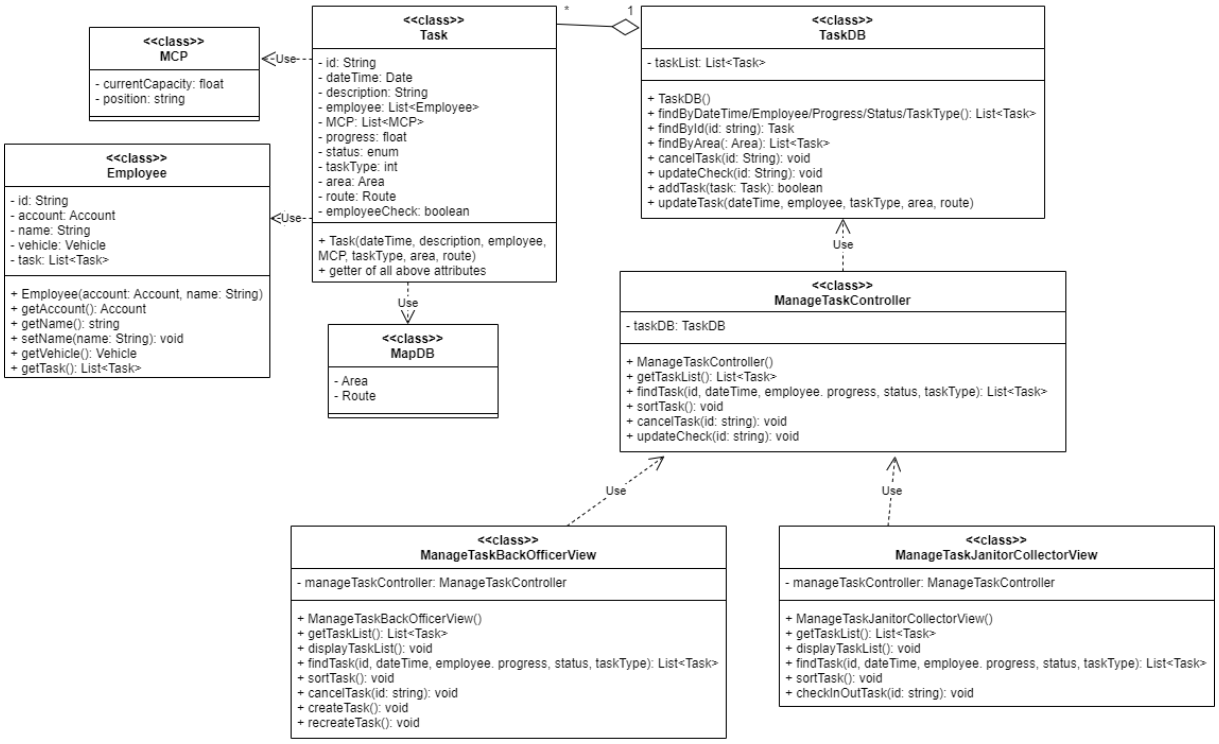


Diagram 4. Back officer / Janitor – Collector Manage task Activity

2.3.5. Create task

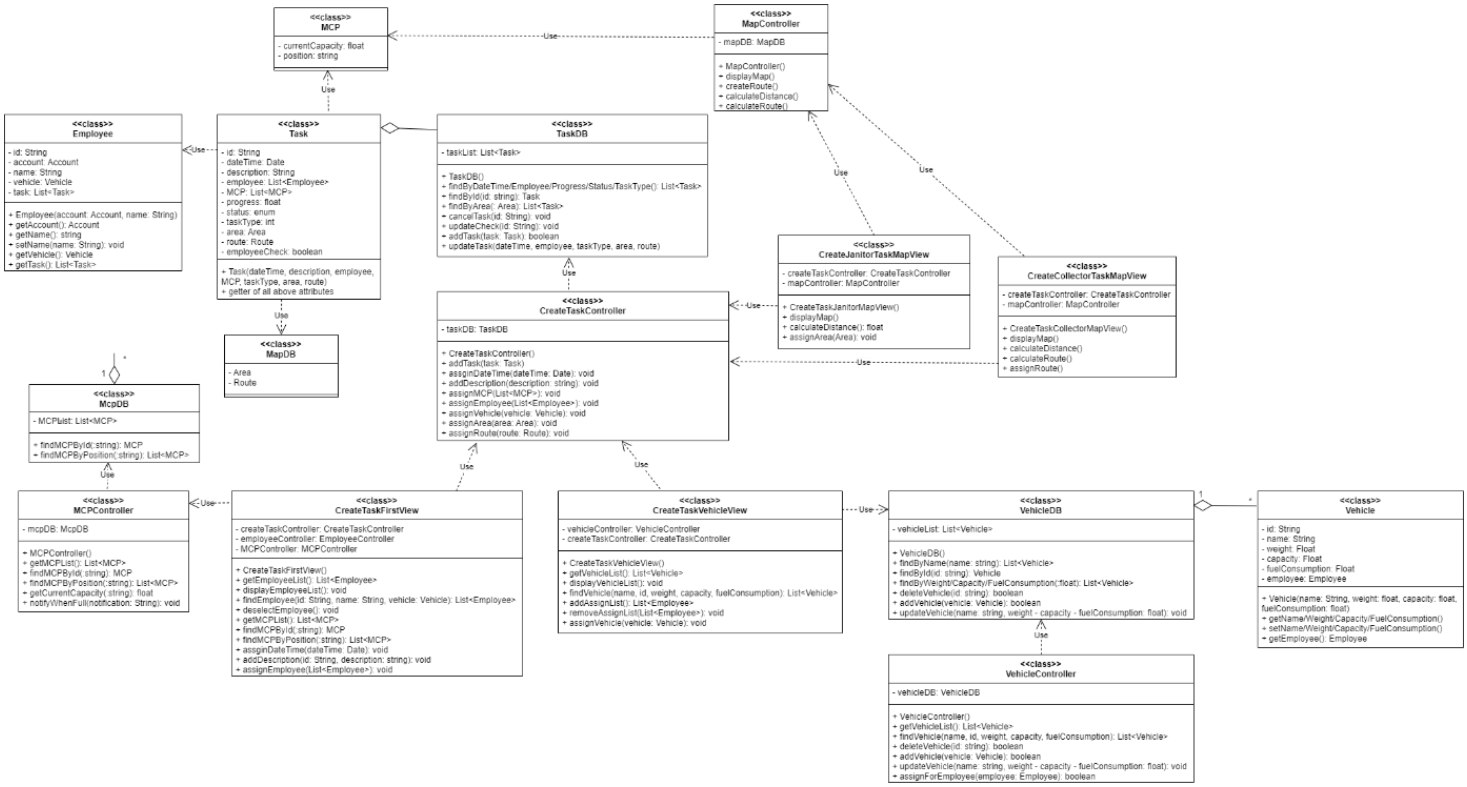


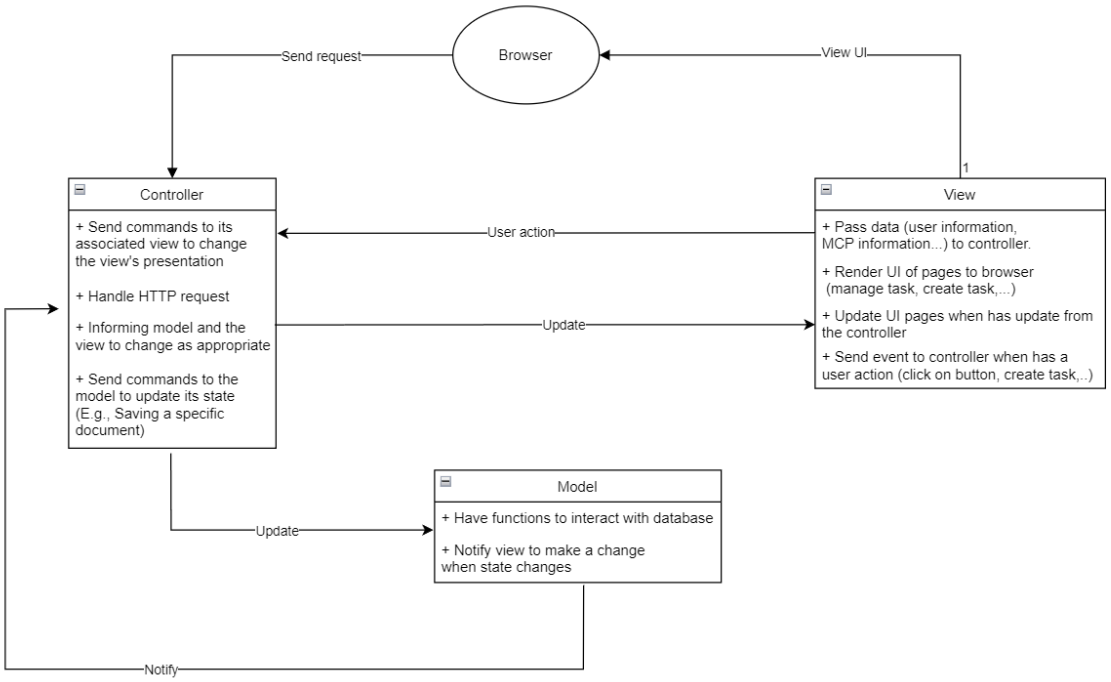
Diagram 5. Back officer Create task Activity

Google Drive:
https://drive.google.com/drive/folders/1nUJXA_6PSIWp5ORsfGcAvlXZAEo-IWeD

IV. Task 3

3.1. Activity diagram:

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



<u>Mô tả:</u>

- View:

+ Send the entered data into the interface to the Controller for processing:
  - User login information such as account, password, role; account registration information; change the account information
  - Search for employees by attributes (id, name,...); add, delete or update an employee's information; assign tasks and vehicles to employees
  - Search vehicle by attributes (id, name, weight, capacity,...); add, delete, update information of a vehicle.
  - Find task information through properties (DateTime, Employee, Process, ID, Area,...); create task; cancel task; update check; add tasks; update task; sort task; recreate task.
  - Search MCP through attributes (ID, Position); create route, calculate distance, calculate route.

+ Render UI for page:
  - Log in page: render login forms, account registration, buttons such as: confirmation, login, forget password,...
  - Employee management page: render an employee management interface including a list of employees, a search bar, a sort tool by attributes, a button to switch to an employee's information interface, task, vehicle.

- Vehicle management page: render the vehicle management interface including a list of vehicles, a search bar, a button to switch to a vehicle's information interface, and an employee interface.
- Task management page: render task management interface including task list, search bar, sort tool; The “create task” button switches to the create task interface, the “recreate” button switches to the recreate task interface, the “cancle” button to cancel the task, the “clear filters” button, the “apply filters” button is used for task filtering.
- Create task page: render create task interface, including a task information input form; button “clear filters”, “apply filters” for filtering workers, “select” button, “deselect” for canceling and selecting workers, “next” button to switch to “Assign vehicle” interface.
- Assign vehicle page: render interface including information table “person selected”, “assign vehicle”, “checkout vehicle”, buttons “assign”, “detail”, “assign vehicle” interact with vehicle, button “done” to complete, button “back” to return to “create task” interface, “next” button to switch to “set map” interface.
- Set map page: render the interface includes the "MCP selected" table containing information about the selected MCPs; map shows the MCPs; buttons  to change the order of MCP, button “choose” to select route, “back” button to return to “assign vehicle” interface, “finish” button to complete task creation.

+ Update UI pages when has update from the controller: perform update is the interface when receiving changes from the controller, retrieve data from the model to do it accordingly

+ Send event to controller when has a user action:

- Send events performed by the user on the interface such as click on button, dropdown list, ... to the controller for retrieval and processing.

- Controller:

+ Returns a view depending on the requested page.

+ Calling functions from the model to perform operations on data:

- Use the model's methods to do things like:
  - Update task, vehicle, employee information, route, area of the MCP.
  - Delete task, vehicle, employee information, route, area of the MCP.
  - Edit details of task, vehicle, employee information, route, area of the MCP.
- Use methods from the model to perform registration, login, and user account management operations.

+ Calling functions from models to use APIs from external systems:

- View details of each employee's individual calendar.
- View detailed information about the system's MCPs.
- View detailed information of a vehicle in the list.

+ Returns the response to the user with the corresponding request.

- Model:

+Provides methods to handle the assignment of collectors and janitors to optimize collection quickly and efficiently.

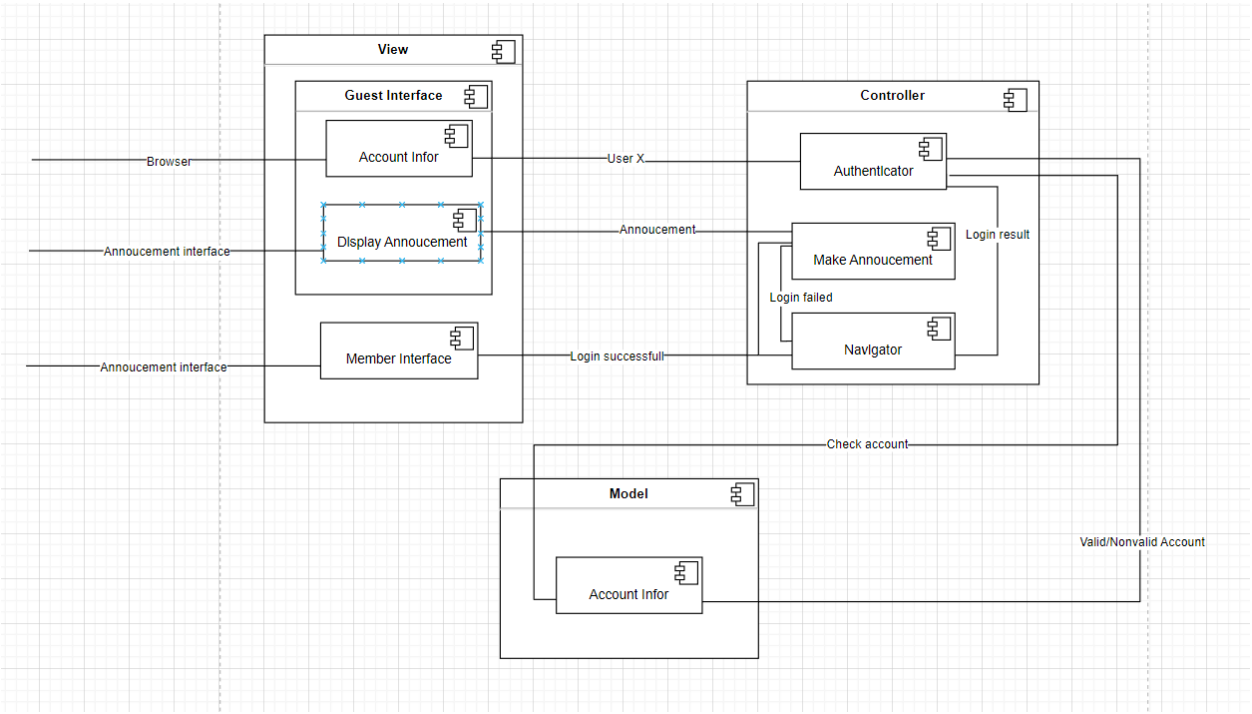
+ Provides methods to interact with the database to perform login, user account management Back officers, collectors and janitors.

+ Provide methods to perform database operations (add, delete, edit, assign...) with the features of route management, vehicle management, and MCP management.

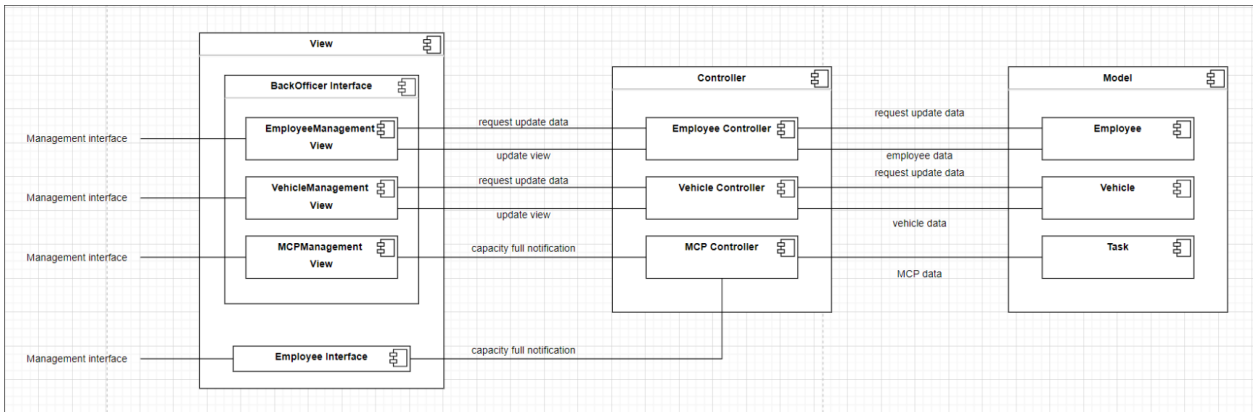
+ When the model's state (about the amount of garbage contained at the MCP, the nearest route, the means of collection...) changes => notify the view to refresh and update the view again.

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

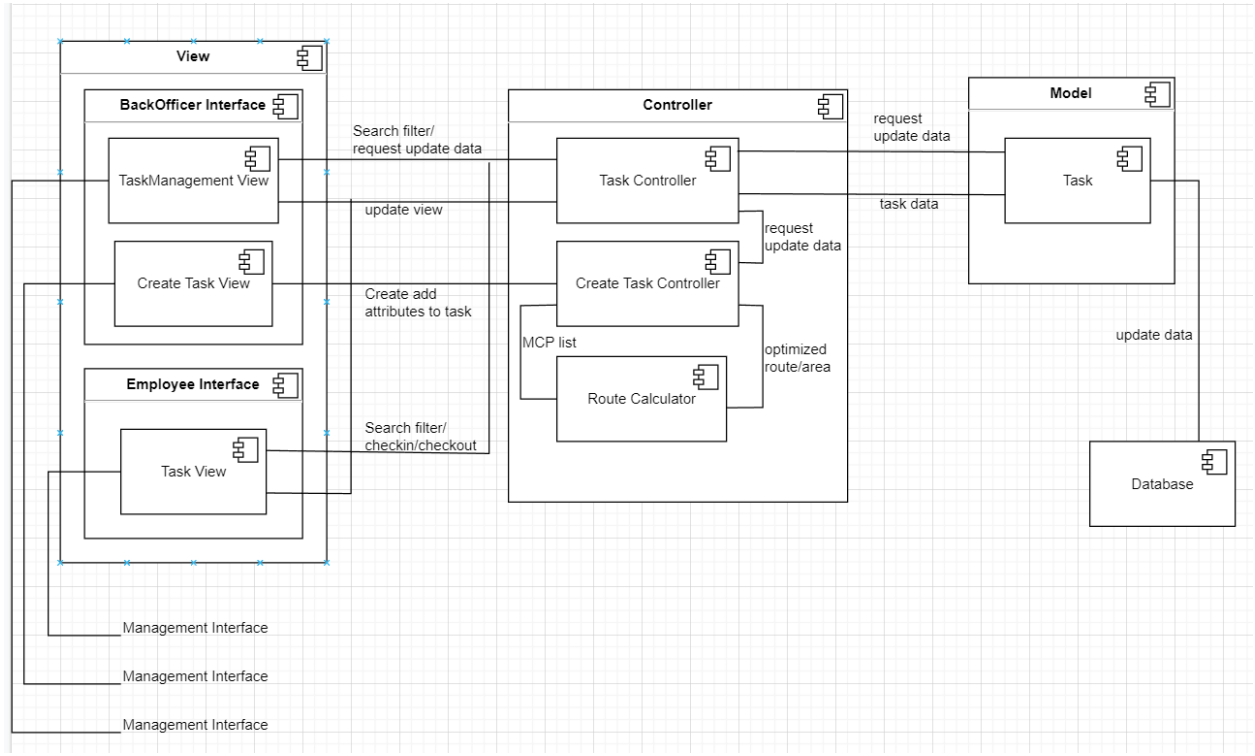
- Component diagram – Log in.



- Component diagram – Vehicle, Employee and MCP Management.



- Component diagram – Task Management.

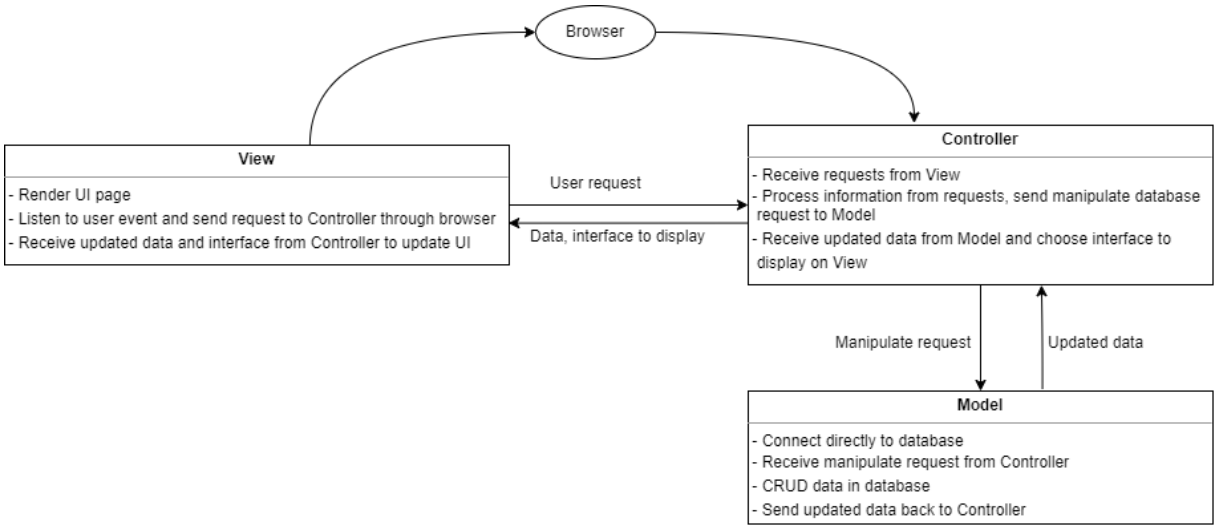


3.2. Draw implementation diagram

For higher quality of images and diagram file, please take a look in our GG Drive:

https://drive.google.com/drive/folders/1ZEsLQrs6oZ98L3Wqgmdqe9oZ0S_uGryQ?usp=sharing

***Approach*:** Use MVC (Model – View – Controller) model, description



Description:

- *Controller* and *Controller connect to each other through browser.*

- *Controller:*

+ Render UI element, data receive from Controller display to the user

+ Listen to event from user like sort, search, create, update, delete and send requests to Controller to handle.

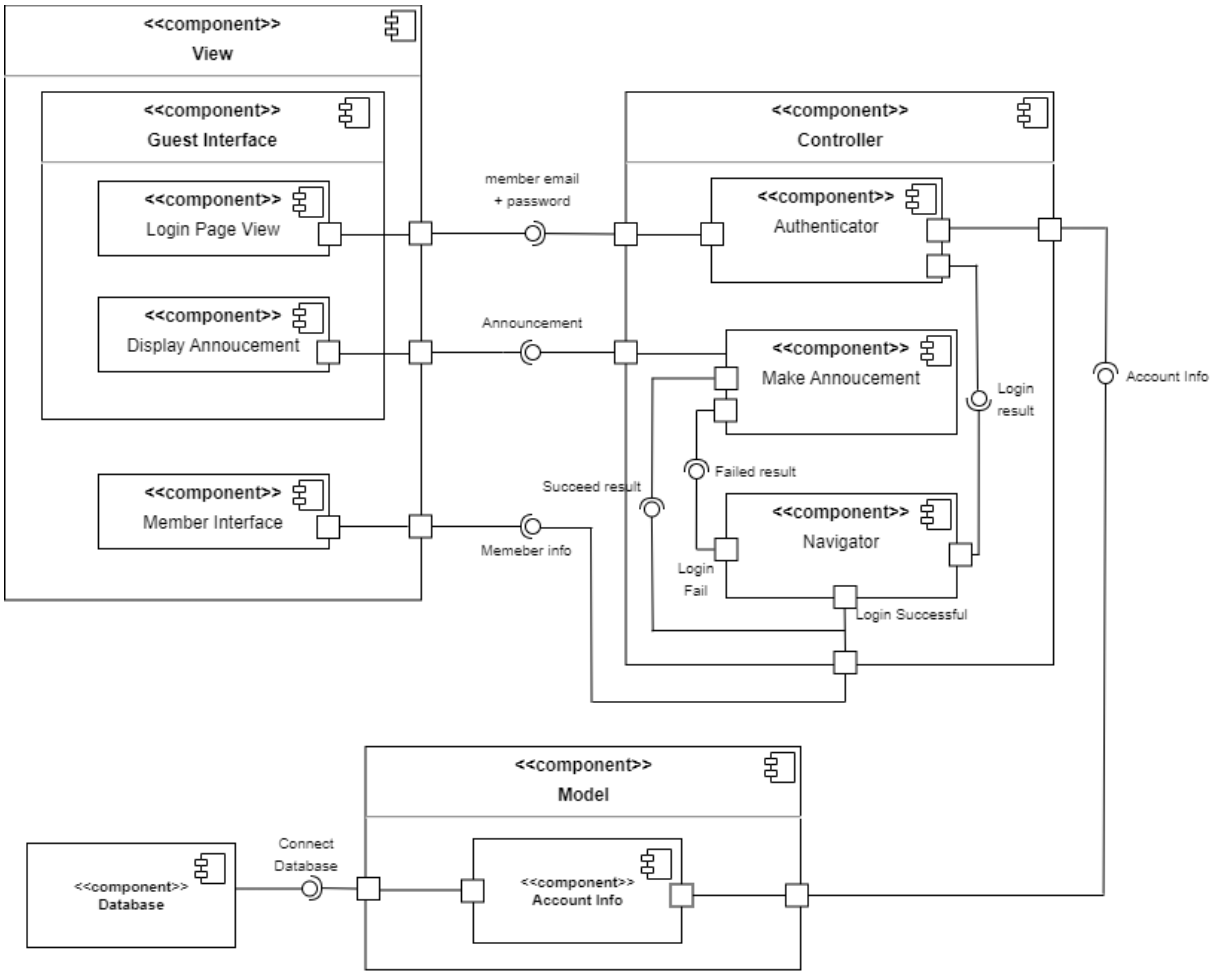
- *Controller:* Handle request and input data from View and request manipulate request to Model, receive data back from user and send to update *View*

- *Model:* connect and can change data in database, send data to *Controller*

Advantages: Separate into three parts, easier to maintain and can be used for more than one component, like *Controller* can be use for both Employee, Vehicle and MCP management.

Difficulty: Make sure the logic between the components work well

3.2.1. Component diagram – Log in



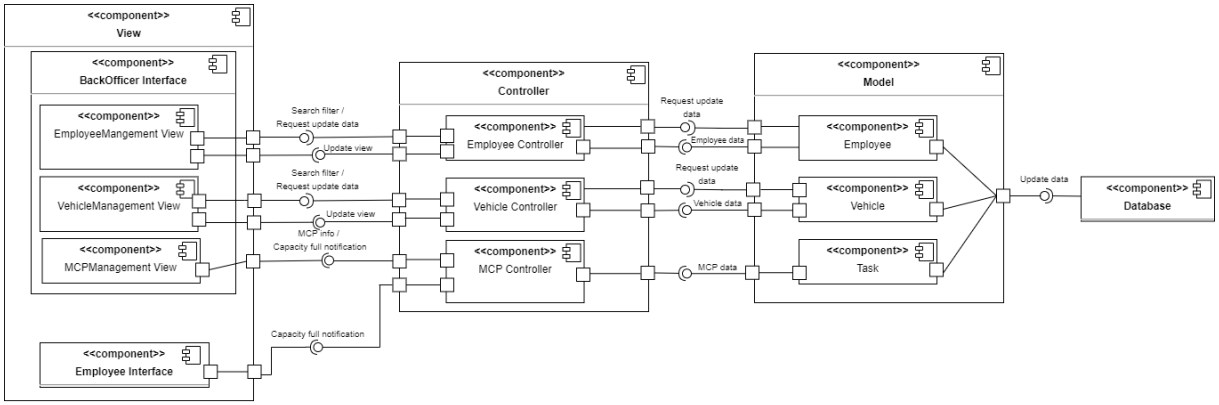
Description:

- At the login page in <*Guest Interface*>, the employees (officer and janitor / collector) input *member email* and *password*. This information are received by <*Authenticator*> component, and compared with data from <*Account Info*> in <*Model*>. After that, the result is sent to <*Navigator*> component to navigate, and notifies to the employee through <*Display Annoucement*> component.

- In case find corresponding account, employee will be navigated to the appropriate <*Member Interface*> with their role and information.

- In case cannot find corresponding account, employee will be notified and requested to login again.

3.2.2. Component diagram – Vehicle, Employee and MCP Management



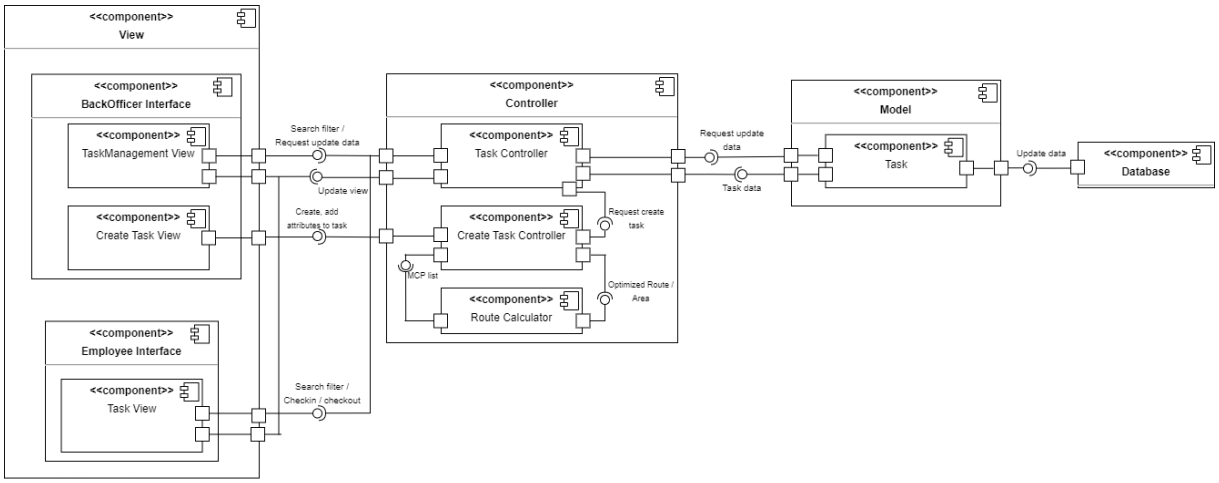
Description:

- According to *MVC* model, from <*View*>, at <BackOfficer Interface>, the back officer has three functions to manages *Employee*, *Vehicle* and *MCP*, which can request to *sort, filter to search* an update data in database.

- Requests from <*View*> are sent to and handled by <*Controller*>, finally sent to <*Model*> to manipulate data in <*Database*>. And again the change of database (include notification when a MCP is full) is sent back to <*Controller*> to update <*View*>.

- In <*Employee Interface*> include janitor and collector, they can't send request to manipulate data, but also receive notification when MCP is full.

3.2.3. Component diagram – Task Management



Description:

With *Task Management* module, the back officer has full fuctions CRUD to the task database (create, read, update, delete (cancel)).

About read, update and cancel tasks, the work flow is same as Employee/ Vehicle/ MCP Management module was presented in previous section, following the MVC model.

The <*Task Assignment*> / <*Create new task*> component has own interface, where the back officer can select information for a new task manually, includes time, employee, vehicle, list of MCPs,… This information is sent to <*Create Task Controller*> component.

- The list of assigned MCP is sent to <*Route Calculator*> component, which is implemented to calculate the optimized area (for janitor) or route (for collector). The output is sent back to <*Create Task Controller*>, now has full information of a task, will send request to create a new task to <*Task Controller*>, the Controller finally work with Model to manipulate database.

About <*Employee Interface*> of janitor/ collector, they can search, sort and change the state of check in / check out their own tasks.