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

SOFTWARE ENGINEERING (CO3001)

Capstone Project

Urban waste collection aid - UWC 2.0

Task 1 - 3

Lecturer: Prof. Quan Thanh Tho
Students: Nguyen Thi Que Anh - 2052380
Ton Huynh Long - 2052153
Le Hoang Minh - 2052595
Trieu Phuc Nguyen - 2011715
Le Ngoc Minh Thu - 2053476

HO CHI MINH CITY, NOVEMBER 2022

Contents

List of Figures

1 Requirement Elicitation

1.1 Context of Project

1.1.1 Relevant Stakeholders

- Back officers
- Collectors
- Janitors
- Service Provider Y

1.1.2 The needs of Stakeholders

- Service Provider Y and Back officers
  - Operate the central system smoothly.
  - Information about janitors and collectors, their work calendars.
  - 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.
  - Information should be updated from MCPs every 15 minutes with the availability of at least 95% of their operating time.
  - Map of the area (traffic information, route, travel distance, fuel consumption) to properly locate MVP and send collectors, janitors.
  - Be able to contact collectors and janitors in case of emergency.
- Collectors and Janitors
  - Have their work planned out on the system.
  - Be able to communicate with each other and with back officers to better coordinate.
  - Be able to check in and out of work everyday on the application.

1.1.3 Problems facing Stakeholders

a. Waste Management

- Urban waste collected is not properly separated into organic and inorganic.
- Temporary dumps are usually not very well-managed, this leads to population and poses a health risk for the local population.

b. Organization

- Back officers

  - Lack information about types of vehicles and MCPs, therefore cannot assign tasks for janitors and collectors in an exact way.
- Janitors and collectors:
  - Need multiple platforms to perform their work efficiently (Calendar, Maps, Chat apps, etc.).
  - Are unable to have instant access to the necessary information while on the move.

1.1.4 Benefits of UWC 2.0

- Back officers
  - Can now easily keep track of the work calendars and availability of janitors and collectors, as well as obtaining information about different kinds of vehicle available, along with route and MCPs. This makes it easier to assign work calendars and proper vehicles to collectors and janitors.
  - Can now immediately contact collectors and janitors to keep them informed of emergencies or sudden changes of route because of traffic problems.
- Collectors and Janitors
  - Can easily keep track of their work shift, and can focus on their work without caring too much about external information.
  - Can now get access to their work information directly without the need to memorize all the details, reducing ambiguity during work.
  - Are able to communicate with other collectors or janitors and back officers to inform about emergencies and get immediately notified of sudden changes of plan.

1.2 Requirements

1.2.1 Functional Requirements

- For back officers
  - They should be able to view information about collectors and janitors working in the same organization.
  - They should be able to view information about the organization's vehicles.
  - They should be able to access information about MCPs, including a map of where they are.
  - They could then assign personnel and vehicles to designated MCPs.
  - The best path for collectors and janitors should also be automatically found.
  - They can contact janitors and collectors on duty.
- For Collectors and Janitors

- They should be able to view their work calendars, their daily tasks and weekly tasks.
- They can contact back officers while on duty.
- They could check in and check out of the system.
- They should get a notification when an MCP is full.

1.2.2 Non-functional Requirements

- Performance
  - System is capable of handling data from 1000 MCPs at the initial launch and 10000 MCPs in 5 years.
  - System is able to import data from UWC 1.0 and work in tandem with it.
  - System should have a user-friendly interface and the most important data should all be accessible in one screen.
  - Support Vietnamese, and also English in the future.
  - The system should function even in low-signal conditions for workers who are on the move.
- Security
  - There should be a backup system for the important data in case of a data loss.
  - A logging on/off system should be implemented and is robust enough to withstand simple attacks.

1.2.3 Use-Case Diagram of the System

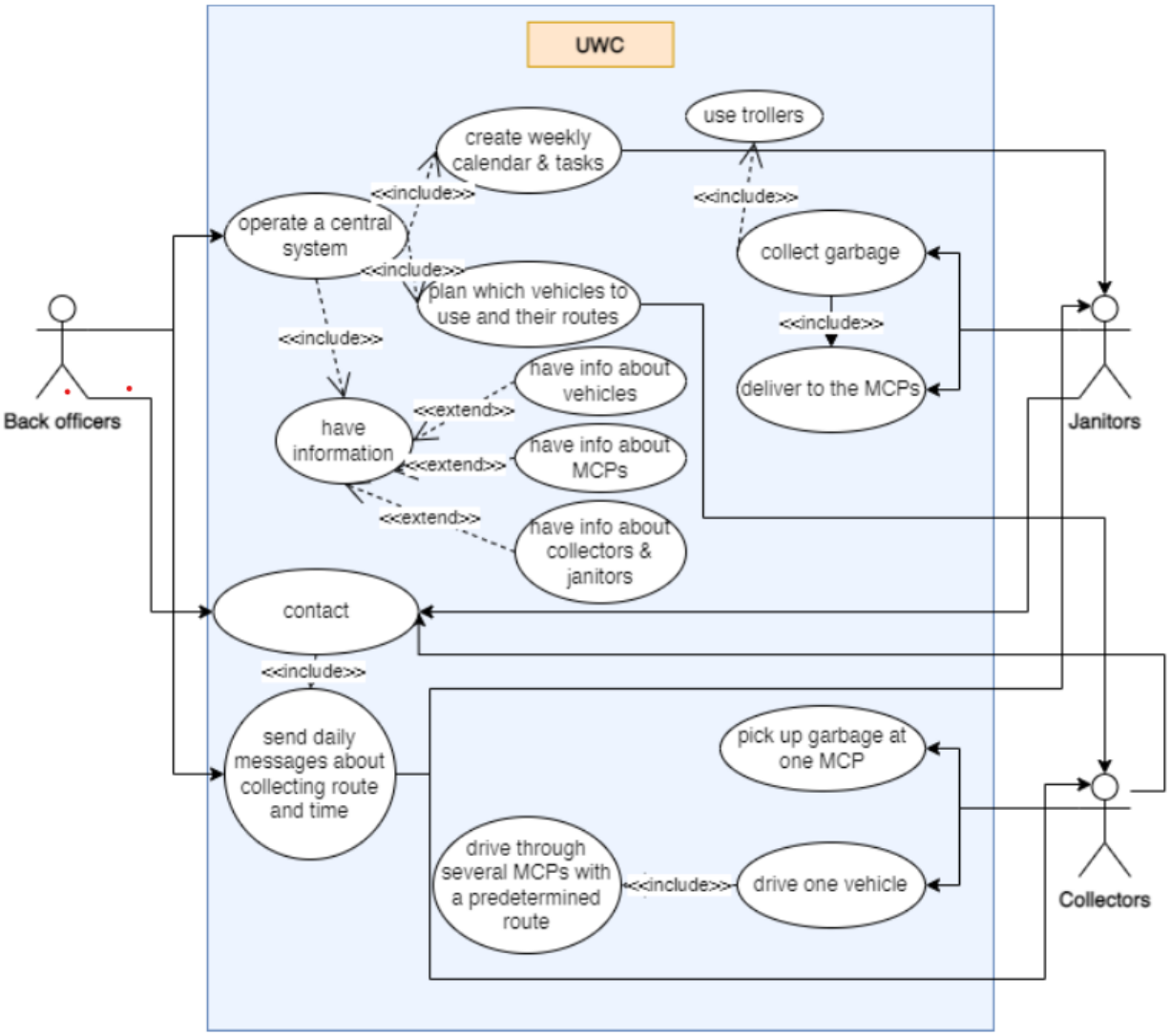

Figure 1: *Use-case Diagram for the system*

1.3 Task Assignment Module

1.3.1 Use-Case Diagram of Task Assignment Module

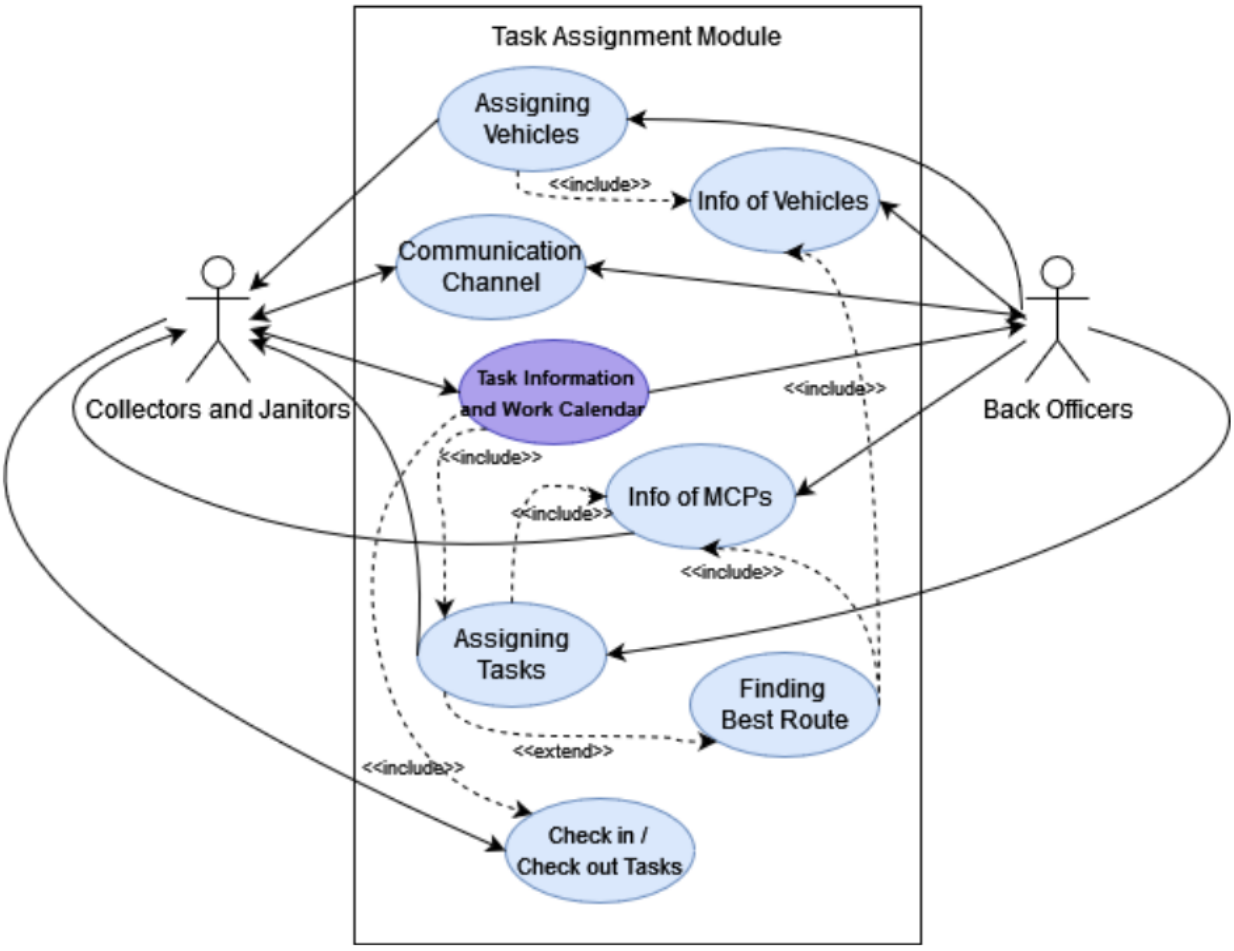

Figure 2: *Use-case Diagram for Task Assignment Module*

1.3.2 Use-Case Description

Use-case name	Assign Task
Actors	Back officers, collectors and janitors
Description	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. This module is provided so that back officers can assign task corresponding to collectors and janitors' work calendar.
Trigger	Click the app to open the system
Pre-condition	Back officers should login to use the system.
Post-condition	Collectors and janitors have the detail of their task.
Normal Flow	1. Back officers check the overview of collectors and janitors and their work calendar. 2. Vehicles and their technical details (weight, capacity, fuel consumptions, etc) are provided for back officers so that they can assign suitable vehicles to janitors and collectors. 3. Details of MCPs can be viewed by back officers, and on-field workers are notified when they're full. 4. Back officers can then assign tasks to janitors and collectors. This makes use of the above MCPs data. 5. In addition, an optimal route is also found for the designated task, making use of both the vehicles data and MCPs data. 6. Collectors and janitors can view details of their given task on a daily and weekly basis; as well as check in and out of tasks every day - this is reflected in their overview information and work calendar. 7. Back officers, collectors and janitors can contact each other through communication channel.
Exception Flow	2a. There are no vehicles available to assign to collectors. 4a. There are no janitors or collectors available to give task. 4b. Task schedule is not feasible.
Non-functional Requirement	1. Information about the capacity of MCPs should be updated from MCPs every 15 minutes with the availability of at least 95 % of their operating time. 2. All important information about tasks should be displayed in one view (without scrolling down). 3. The messages should be communicated in a real-time manner with delay less than 1 second.

Table 1: *Description*

2 System Modeling

2.1 Activity Diagram of Task Assignment Module

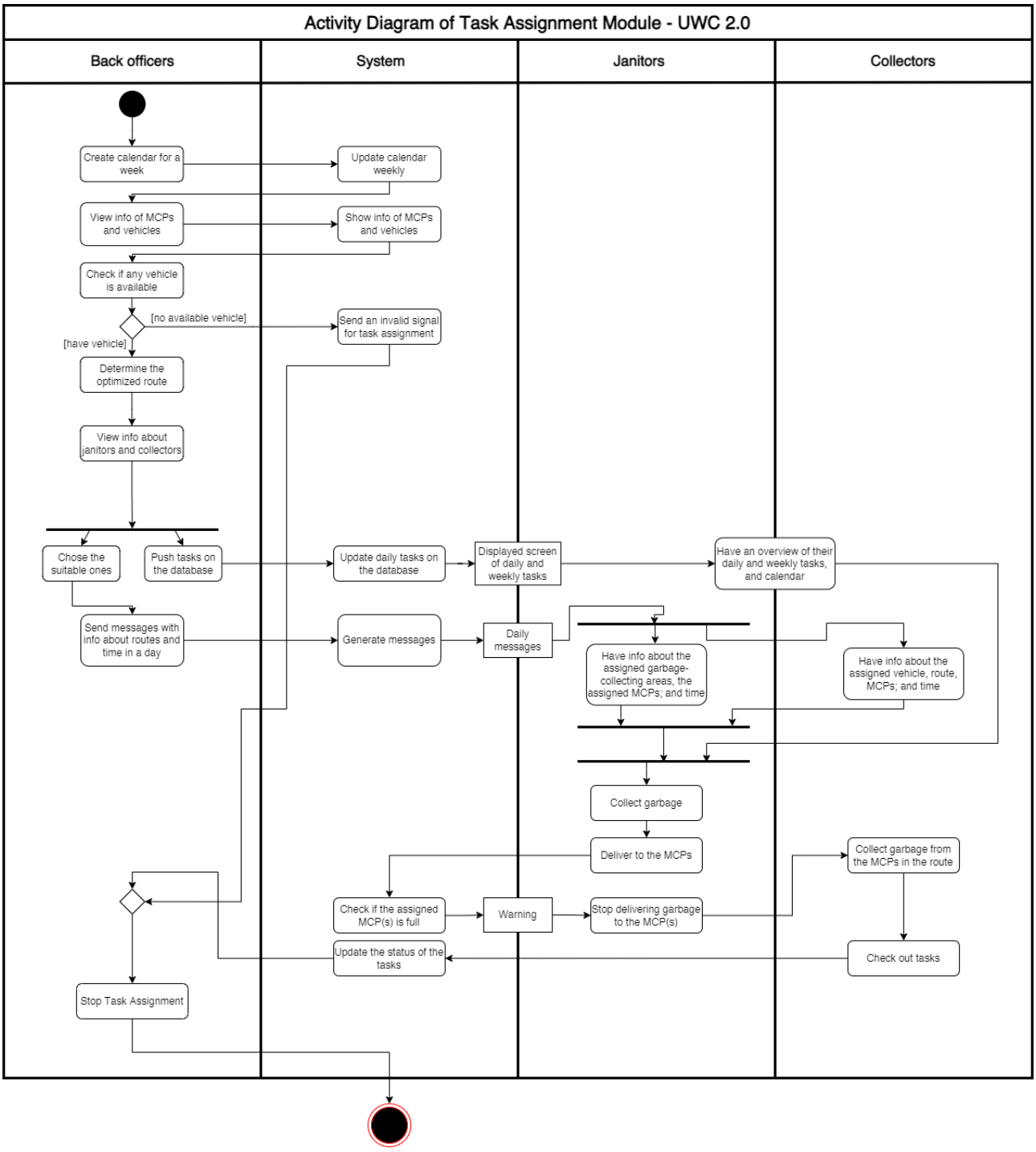

Figure 3: *Activity Diagram*

Description:
First, the back officers are those who create calendar weekly and create plan to assign vehicle, determine route and destine the garbage-collecting sites, as well as assigning the MCPs to the available collectors and janitors. While doing this, the back officers update and request information to/from the system.

For unavailable actions (ex.: when there is no available vehicle), the system generates invalid signal to the task assignment that will stop the whole module job.

After doing all planning and delegating tasks, the back officers will send daily message(s) to the collectors and janitors through the system, that includes information about the tasks and time of a day. The collectors and janitors also have an overview of all weekly and daily tasks, as well as the calendar in one screen from the system. By the way they check their messages, the collectors and janitors confirm to check in their tasks.

The janitors will then collect garbage at their assigned areas and deliver to the MCPs. If any MCP is full, the system will update the status of that MCP so that the janitors stop delivering garbage to that MCP, and then collectors are those who collect garbage from the MCPs in their predetermined route of that day.

When the janitors finish their work, they check out their tasks and update tasks status on the system.

2.2 Sequence Diagram of Route Planning Task

2.2.1 Conceptual Solution

Route planning	Description
1. Back officers will discover information about MCPs and vehicles. Then, back officers determine the route(s) by using the system to create the optimized one(s).	1.Homepage sends requests to the system to get information about MCPs and vehicles. 2. After receiving general information, back officers check if any vehicle is available. 3.Back officers then decide to find the route(s) based on the suggestions from the system.
2. Back officers will eventually arrange the available janitors and collectors according to the suitable vehicles.	1. If any vehicle is available, back officers then view information about janitors and collectors. 2. When choosing the suitable one, back officers will assign task(s) for them and also push the recording task(s) on the database. 3. If there are unavailable janitors or collectors, the system sends an invalid signal for task assignment. 4. If there are no available vehicles, then return the similar message.

2.2.2 Sequence Diagram

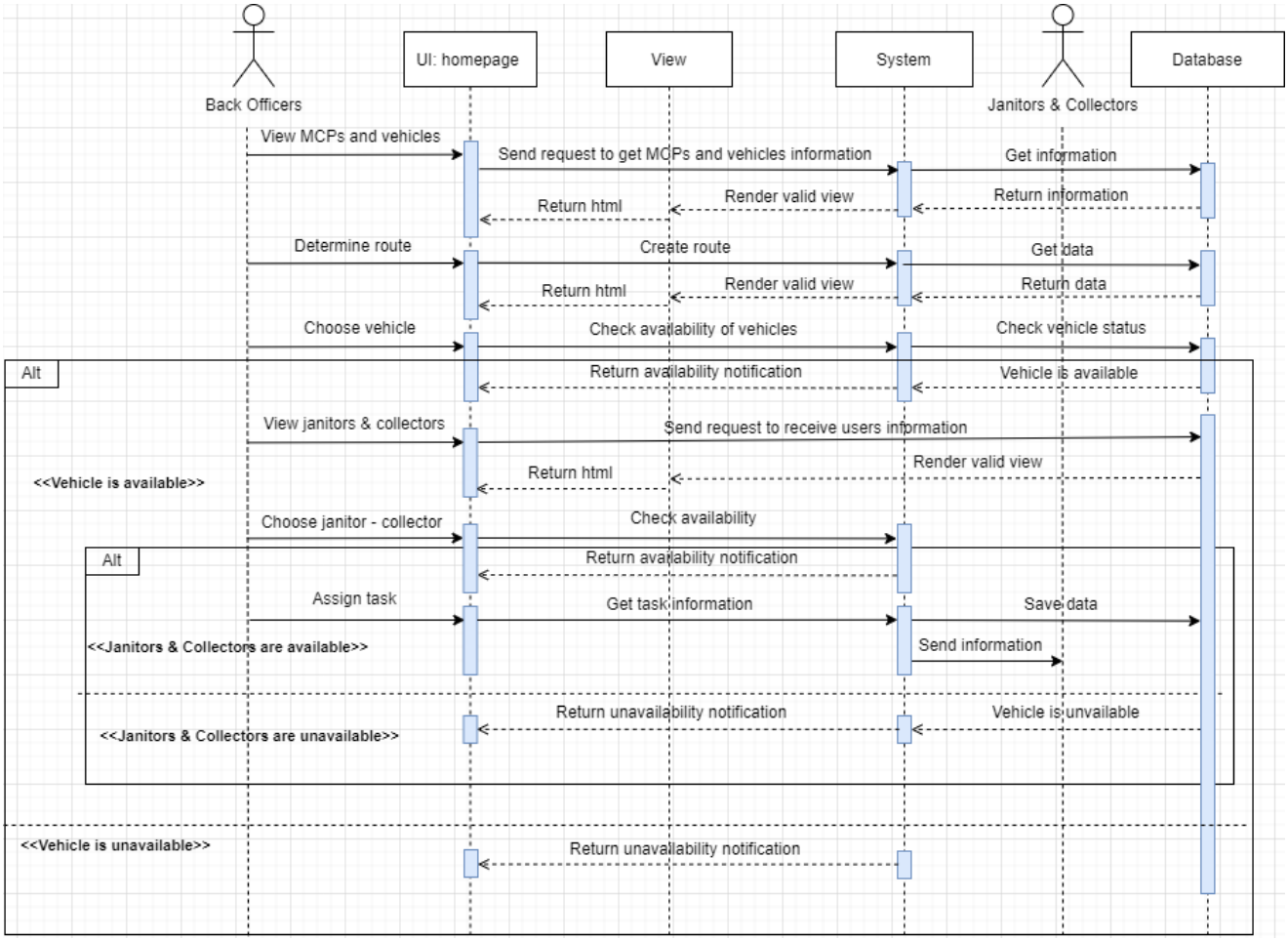

Figure 4: *Sequence Diagram*

2.3 Class Diagram of Task Assignment Module

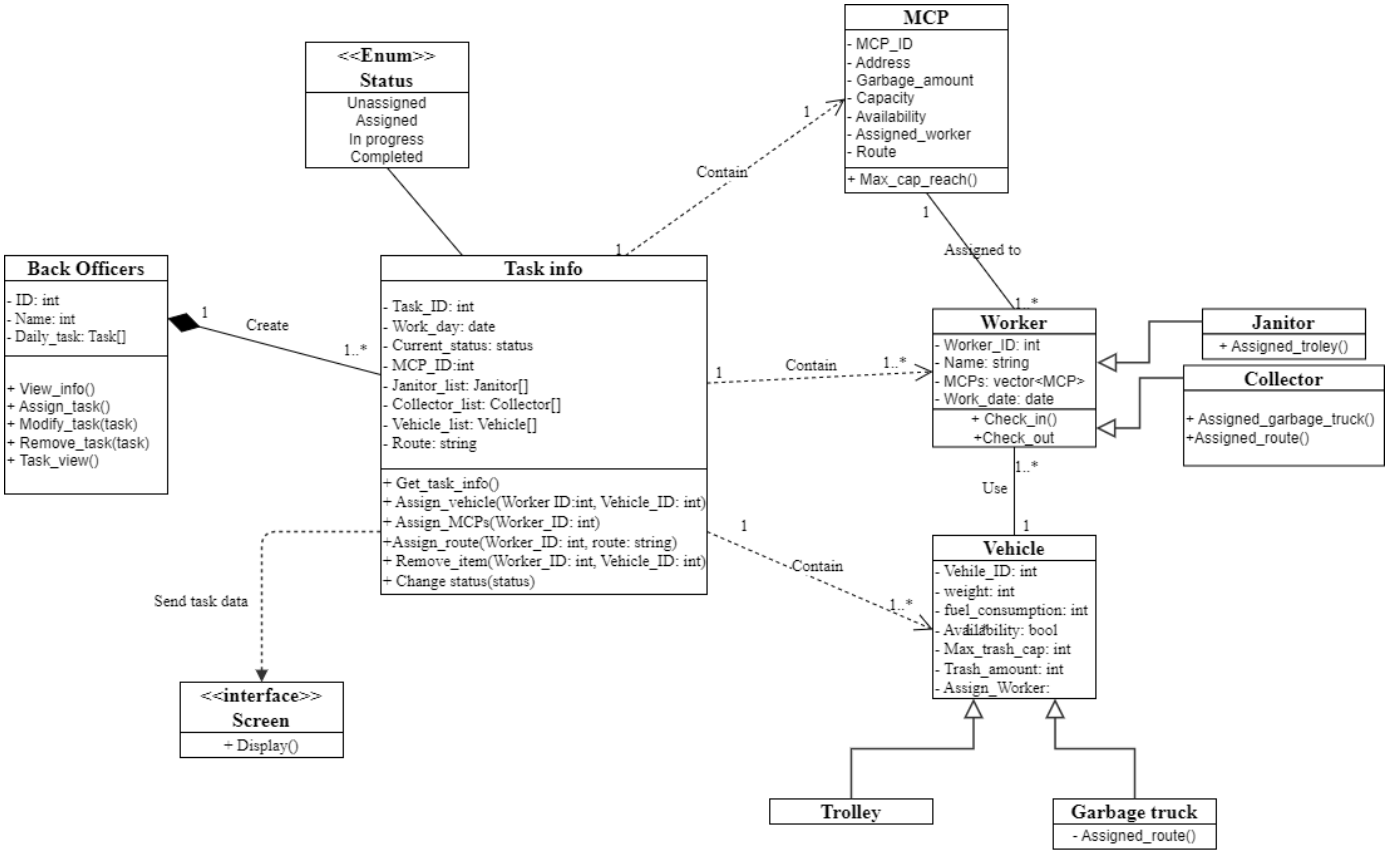

Figure 5: *Class Diagram*

3 Architecture Design

3.1 Architectural approach

THEORY

About MVC design pattern

MVC (Model-View-Controller) is a pattern in software design which specifies that an application consists of data model, presentation information, and control information. It emphasizes a separation between the software's business logic and display. This "separation of concerns" provides for a better division of labor and improved maintenance. Some other design patterns are based on MVC, such as MVVM (Model-View-Viewmodel), MVP (Model-View-Presenter), and MVW (Model-View-Whatever).

I. Controller:

1. Input:
   - User Interactions and Requests (from View).
   - New information (from Model).
2. Output:
   - Information to be added, updated or deleted (to View).
   - User-generated Information (to Model).
3. Function:
   - Send the message that has been typed in by the user.
   - Receive message from other users.
   - Fetch and Deliver information.

II. View:

1. Input:
   - Data to be displayed or updated (from Controller).
   - Notifications (from View).
   - User inputs (from User).
2. Output:
   - User-inputted information and User Events (to Controller).
   - Web page and Interface (to User).
3. Function:
   - Draw Web Page.
   - Adapt to window size dynamically.
   - Display information.

- Update displayed information.
- Take in user inputs.

III. Model:

1. Input: User-generated Information (from Controller).
2. Output:
   - Information to be displayed or updated (to Controller).
   - Necessary notifications (to View).
3. Function:
   - Work with databases - get information or append new information.
   - Personnel Database (Back Officers, Janitors, Collectors, etc).
   - Vehicle Database.
   - MCP Database.
   - Perform necessary algorithms for tasks like finding the best path for vehicles.
   - Handle internal logic.

SOLUTION

We are intended to divide our whole system into 4 big modules discussed in task 1:

1. Account user
   - Input: Username and Password.
   - Function This module will receive user login information (user name, password, ..), then compare with the registered ID in the database system and allow the user access the internal website if they match, otherwise, log in again.
   - Output: User's authentication and ID.
2. Task Assignment
   - Input: User ID, MCPs ID, Vehicles ID, date and time.
   - Function: This module will give data for back officers which include working workers, available vehicles and a calendar to see tasks and when to update MCPs.
   - Output: An updated calendar with proper tasks' assignment.
3. Contact
   - Input: Message, Sender ID, Receiver ID.
   - Function: This module helps back officers and workers to communicate to each other in case of changing course or task update.
   - Output: Text output at the receiver's view.

4. MCP's state updation

    - Input: MCP's id, new state
    - Function: This module checks the status of MCPs in real-time, and updates its status (capacity) to back officers.
    - Output: New state of MCP.

MVC Adaptation

1. Account Module:

    - Model: Database of register's User ID.
    - View: Authentication UI (for user to type in their log in ID).
    - Controller: Receive users' login ID -> Request Model to check login ID from database, if Model cannot find the related ID, update View to show Reject notification, if Model finds the matched ID, Model sends notification to View to display user's data view of that ID.

2. Task Assignment Module:

    - Model: Calendar information and updating tasks.
    - View: Calendar UI with task checklist.
    - Controller: Receive task assignment information from back officers, update Model then generate the new tasks (discussed further in 3.2).

3. Contact module:

    - Model: Chat log of users.
    - View: Chatbox between users.
    - Controller: Receive messages from sender, update it into Model, Model updates that message in both user's data and then send to the corresponding receiver.

4. MCP's state updation:

    - Model: Different ID state.
    - View: No UI is needed.
    - Controller: Receive the status from MCP, update it to Model then also generate new task list.

3.2 Implementation Diagram for Task Assignment Module

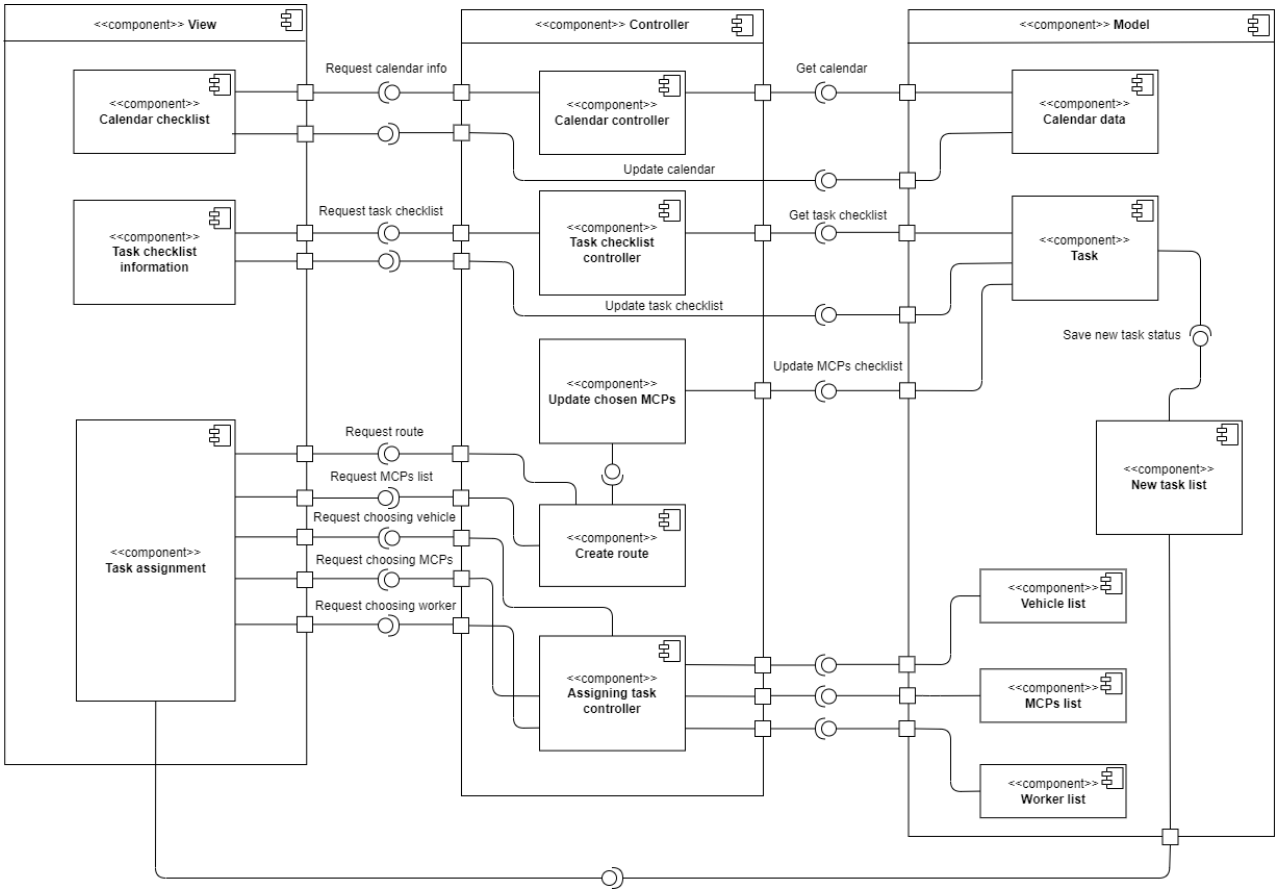

Figure 6: *Implementation Diagram*

4 Task 4

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

Here is the layout of our GitHub repository:

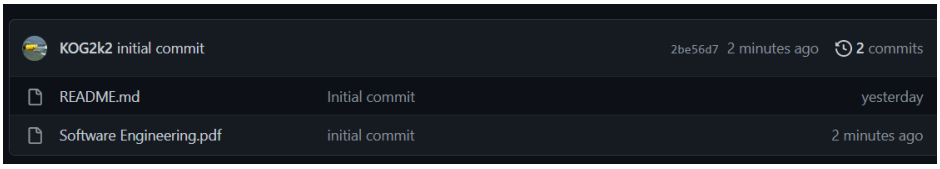

Figure 7: *GitHub Repository layout*

For log version control, we will have git history to check member's contribution in this

sprint.

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

In order to gain the full control of our project, please access the following link: https://github.com/KOG2k2/UWC-2.0

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

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

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

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

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