MINISTRY OF EDUCATION AND TRAINING

FPT UNIVERSITY

Capstone Project Document

Scheduling working time and Timekeeping by Face Recognize for Staff

Group - GSU21SE39	
Group Members	Pham Minh Dao - SE130164 Ly Van Cuong - SE130136 Do Quoc Trung - SE130447 Vu Thi Ngoc Mai - SE130264
Supervisor	Mr. Lam Huu Khanh Phuong
Ext Supervisor	Mr. Doan Nguyen Thanh Hoa
Capstone Project code	SU21SE41

- Ho Chi Minh, 08/2021 -

Table of Contents

Table of Contents

List of figures

List of tables

Acknowledgement

First of all, we would like to express our deep appreciation for our mentors, Mr. Lam Huu Khanh Phuong and Mr. Doan Nguyen Thanh Hoa for guiding us on this capstone project. Without your help and support we could not finish this project.

Secondly, we would like to thank our lecturers for the time we spent studying in FPT University. Without you, we could not prepare enough knowledge to do the capstone project.

And lastly, we would like to thank our friends and family, who supported us during the time we did our project. And of course all the team members, who do their best and hard-working for this capstone.

Definition and Acronyms

Acronym	Definition
STS	Scheduling working time and Timekeeping by face recognize for Staff
AWS	Amazon Web Services
MD	Module
BR	Business Rule
ERD	Entity Relationship Diagram
GUI	Graphical User Interface
PM	Project Manager
SDD	Software Design Description
SPMP	Software Project Management Plan
SRS	Software Requirement Specification
UAT	User Acceptance Test
UC	Use Case
API	Application Program Interface

I. Project Introduction

1. Overview

1.1 Project Information

- Project name: Scheduling working time and Timekeeping by Face recognize for Staff
- Project code: SU21SE41
- Group name: GSU21SE39
- Software type: Mobile Application, Web Application

1.2 Project Team

a. Supervisor

Full Name	Email	Phone Number	Title
Lâm Hữu Khánh Phương	phuonglhk@fe.edu.vn	0915353001	Lecturer
Đoàn Nguyễn Thành Hòa	hoadnt@fe.edu.vn		Lecturer

b. Team Members

Full Name	Email	Mobile	Role
Phạm Minh Đạo	daopmse130164@fpt.edu.vn	0368888100	Leader
Đỗ Quốc Trung	Trungdqse130447@fpt.edu.vn	0917920689	Member
Lý Văn Cường	Cuonglvse130136@fpt.edu.vn	0979568357	Member
Vũ Thị Ngọc Mai	maivtnse130264@fpt.edu.vn	0332756462	Member

2. Product Background

The number of staff in chain stores is increasing rapidly these days. As a result, managers who have to schedule staff's working time and attendance become more difficult and time consuming. Furthermore, managers who manage store chains have difficulty when the number of stores grow, if they manage it in traditional ways, it is hard to keep track of their business.

The problem is that manually scheduling staff's working time and timekeeping becomes complicated and costly. Especially when the number of staff in each store is large. So it would be nice if there is some software which helps them to do the work.

3. Existing Systems

3.1 Deputy

Deputy is a cloud-based human resource management (HRM) solution that caters to businesses of all sizes across various industry verticals

and provides them employee management and scheduling functionalities. Furthermore, Deputy provides an Auto-Scheduling feature that uses AI to build shift structure based on as many metrics (demand signals).
Preference: https://www.deputy.com

3.2 7Shifts

7shifts is an online employee scheduling software system designed for shift workers operating in the restaurant industry. The solution allows employees to check their upcoming shifts, scheduled availability and time-off requests from the 7shifts home screen using a mobile app. However, scheduling is done manually using templates
Preference: https://www.7shifts.com

4. Business Opportunity

This provides the opportunity for brands that need to digitize their business. Which will help them manage their chain of stores and staff. When optimizing the schedule arrangement, it will save business salary costs for staff, save time and effort for managers,…
Compared with traditionally, when managers arrange working time for staff, they may use excel or manual methods, it would be much more difficult if the number of staff grows.

5. Software Product Vision

We want to help businesses to optimize working schedule, provide staff with a reasonable working time sheet. So it will improve both the satisfaction of managers and employees.
In the future, it is necessary to automate working schedule arrangements for staff and time attendance to save management time and effort.

6. Project Scope & Limitations

6.1 Major Features

FE-01: Allow store managers to track the working schedule.

FE-02: Allow store managers to track the staff’s attendance.

FE-03: Allow store managers to run schedule algorithms.

FE-04: Allow store managers to publish the working schedule.

FE-05: Allow store managers to generate reports.

FE-06: Allow brand managers to manage all stores of their brand.

FE-07: Allow brand managers to manage accounts of their brand.

FE-08: Allow staff to register his/her working time for next week.

FE-09: Allow staff to track his/her working schedule.

FE-10: Allow staff to track his/her attendance.

FE-11: Allow staff to register his/her face identity at store face recognized device.

FE-12: Allow staff to take attendance by store face recognized device.

6.2 Limitations & Exclusions

LI-1: Some scenarios when the staff available time is not enough for the demands required, the schedule can not be computed.

EX-1: STS does not support the calculation and management salary of employees. STS only schedules working time and checks attendance.

II. Project Management Plan

1. Overview

1.1 WBS & Estimation

#	WBS Item	Complexity	Est. Effort
1	***Initiating***		***8***
1.1	Create Project	Simple	1
1.2	Kick-off Meeting	Simple	2
1.3	Get Requirements	Medium	5
2	***Planning***		***11***
2.1	Create Scope Management Plan	Simple	1
2.2	Create Schedule Plan	Simple	2
2.3	Create Project Development Plan	Simple	3
2.4	Create Testing Plan	Simple	2
2.5	Team meeting For Project Management plans discussion	Simple	2
2.6	Deliver Project Management Plan	Simple	1
3	***Analysis***		***30***
3.1	System Requirement Specification	Medium	5
3.2	Software Design Document	Medium	5
3.3	System Technology		
3.3.1	Google OR-Tools scheduling	Complex	10
3.3.2	AI face detect	Complex	10
4	***Design***		***45***
4.1	System Architecture	Medium	6
4.2	Database Architecture	Complex	14
4.3	Api Design	Medium	5
4.4	Admin Web App Interface	Simple	3
4.5	Brand & Store Manager Web App Interface	Medium	10

4.6	Mobile App Interface	Medium	7
5	***Implementation***		***182***
5.1	Deploy database on cloud	Simple	1
5.2	Setup development environment	Simple	2
5.3	Implement Web API server		
5.3.1	Initialize project	Simple	1
5.3.2	Authentication API (Login/Register)	Simple	3
5.3.3	CRUD Entities API	Medium	16
5.3.4	Reports API	Medium	5
5.3.5	Scheduling request API	Medium	10
5.3.6	Send notifications to clients	Medium	5
5.4	Implement Scheduling Algorithm API server		
5.4.1	Initialize project	Simple	1
5.4.2	Compute schedule	Complex	29
5.5	Implement Web admin		
5.5.1	Initialize project	Simple	1
5.5.2	Login/Logout	Simple	1
5.5.3	Manage Brands	Simple	2
5.5.4	Manage Users	Simple	3
5.6	Implement Web Brand Manager		
5.6.1	Initialize project	Simple	1
5.6.2	Login/Logout	Simple	3
5.6.3	Manage brand’s stores	Medium	5
5.6.4	Manage brand’s users	Medium	5
5.7	Implement Web Store Manager		
5.7.1	Initialize project	Simple	1
5.7.3	Manage store schedules	Complex	10
5.7.4	Track staff attendances	Medium	7
5.7.5	Manage staff requests	Medium	5

5.7.6	Reports	Medium	5
5.8	Implement Staff Mobile Application		
5.8.1	Initialize project	Simple	1
5.8.2	Register schedule	Medium	8
5.8.3	Track working schedule	Medium	8
5.8.4	Track working attendance	Medium	8
5.8.5	Request day off / change shift	Medium	5
5.9	Implement Face recognition		
5.9.1	Train model	Medium	15
5.9.2	Setup Devices	Medium	15
6	***Testing***		***17***
6.1	Perform Unit Test	Medium	5
6.2	Perform Integration Test	Medium	5
6.3	Perform System Test	Medium	5
6.4	Perform User Acceptance Test	Simple	2
7	***Closing***		***2***
7.1	Deliver documents	Simple	1
7.2	Project Transfer	Simple	1
8	***Presentation***		***7***
8.1	Prepare presentation slides	Medium	3
8.2	Presentation practices	Medium	4
		Total Estimated Effort (man-days)	***302***

Documentation WBS

#	WBS Item	Complexity	Est. Effort
1	***Project Introduction***		***3***
1.1	Market research	Simple	1
1.2	Writing docs	Simple	2
2	***Project Management Plan***		***6***

2.1	WBS	Simple	3
2.2	Writing Plans	Simple	3
3	***System Requirement Specification***		***28***
3.1	Business rules	Simple	2
3.2	Use Case Diagram	Medium	8
3.3	Entity Relationship Diagram	Medium	10
3.4	Feature List	Simple	3
3.5	Functional/Non-functional requirements	Medium	5
4	***Software Design Document***		***35***
4.1	Package Diagram	Medium	5
4.2	Class Diagram	Medium	5
4.3	Sequence Diagram	Medium	10
4.4	Database Design	Medium	15
5	***Software Test documentation***		***12***
5.1	Test Plan	Simple	2
5.2	Perform Testing	Medium	8
5.3	Report	Simple	2
6	***Software User Guides***		***4***
6.1	Installation	Simple	2
6.2	Application usage	Simple	2
7	***Final Project Report***		***2***
7.1	Gather Report	Simple	2
		Total Estimated Effort (man-days)	***90***

1.2 Project Objectives

#	Quality Stage	No. of Defects	% of Defect	Notes
1	Reviewing	10	27%	
2	Unit Test	15	40.54%	
3	Integration Test	5	13.51%	
4	System Test	4	10.81%	

5	User Acceptance Test	3	8.14%	
	Total	***37***	***100%***	

1.3 Project Risks

#	Risk Description	Impact	Possibility	Response Plans
1	Business is not clear or understand wrong	Critical	Medium	Review with the product owner/mentor frequently to ensure the team is working in the right direction.
2	Can not finish project on time	Critical	Medium	Priority core features.
3	Can not find best solution, technical approach	High	Medium	Clarify business point of view, ask for help from mentors.
4	Conflicts between team members, mentor	Medium	High	Arrange a meeting to talk clearly.

2. Management Approach

2.1 Project Process

SCRUM Framework

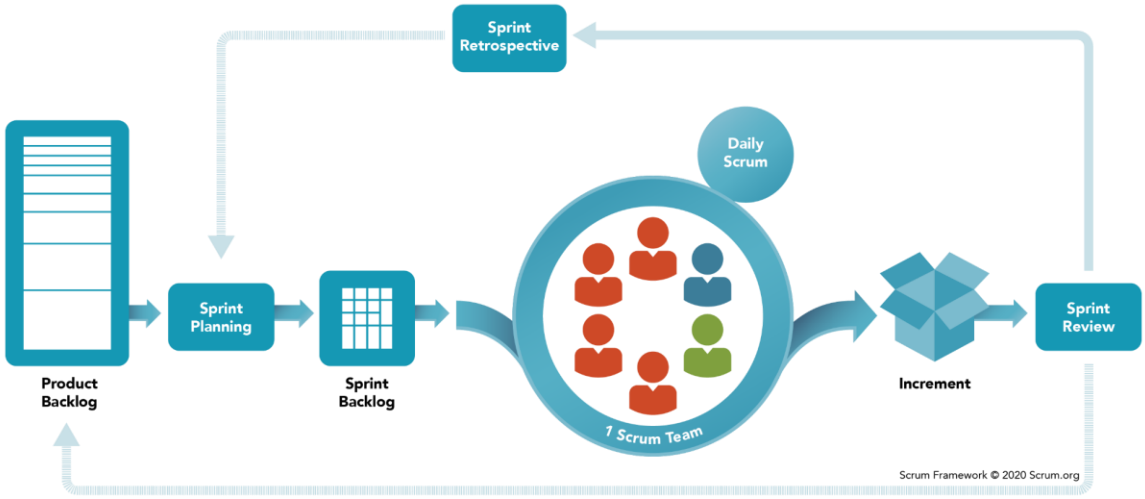


Figure 1 <Reference> SCRUM Framework

Source: https://www.scrum.org/resources/what-is-scrum

Our system is developed using the Scrum Framework - an Agile mindset for developing, delivering software products. We chose this framework because of the following reasons.

- **Easy to provide tasks and track working processes**: Each member is assigned to do individual features in the system, which require fixed time to deliver deliverable components.
- **Fast learning and development**: This project has many technologies that we need to research and apply. Including learning, analysis, design, coding and testing. With the Scrum Framework, we can do it in parallel.
- **Flexible in changing requirements and scope**: Product owners can change their requirements due to the needs of realistic business.

2.2 Quality Management

To ensure project quality, we apply these following techniques:

- Team policy: Work hours, responsibility,...
- Review daily: Tasks, cross review.

- Coding convention: apply clean code, follow pre-defined coding conventions for front-end, back-end and mobile application. Commit code clearly.
- Testing: Write and do unit test cases, integration, system, user acceptance test.

2.3 Training Plan

Training Area	Participants	When, Duration	Waiver Criteria
ASP.Net Core	DaoPM, CuongLV	10/05/2021, 2 weeks	Mandatory
ReactJS	CuongLV, MaiVTN	10/05/2021, 2 weeks	Mandatory
Flutter	TrungDQ	10/05/2021, 2 weeks	Mandatory
Flask	TrungDQ	10/05/2021, 2 weeks	Mandatory
Git, Github	Everyone	10/05/2021, 2 days	Mandatory

3. Master Schedule

#	Deliverable	Effort (days)	Due Date	Deliverable Scope
1	Project Introduction	3	12/05/2021	Product vision, scope and major features list
2	Project Management Plan	5	17/05/2021	WBS, Team structure, risks and plans
3	SRS	10	27/05/2021	Project overview, business rules, use cases, functional/non-functional requirements

4	Software Design Document	30	20/06/2021	Architecture Design, Detailed Design, Database, diagrams
5	Code Package - Web API	40	30/06/2021	Code & Unit test, System test cases
6	Code Package - Web Admin	10	10/06/2021	Code & Unit test, System test cases
7	Code Package - Web Brand & Store Manager	40	23/07/2021	Code & Unit test, System test cases
8	Code Package - Web Schedule Algorithm Server	30	30/06/2021	Code & Unit test, System test cases
9	Code Package - Staff Mobile App	30	30/06/2021	Code & Unit test, System test cases
10	Code Package - Timekeeping by Face Detect	30	10/07/2021	Code & Unit test, System test cases
11	UAT Package	7	30/07/2021	Test reports
12	Final Package	5	15/08/2021	Final Codes & documents, User manual

4. Project Organization

4.1 Team & Structures

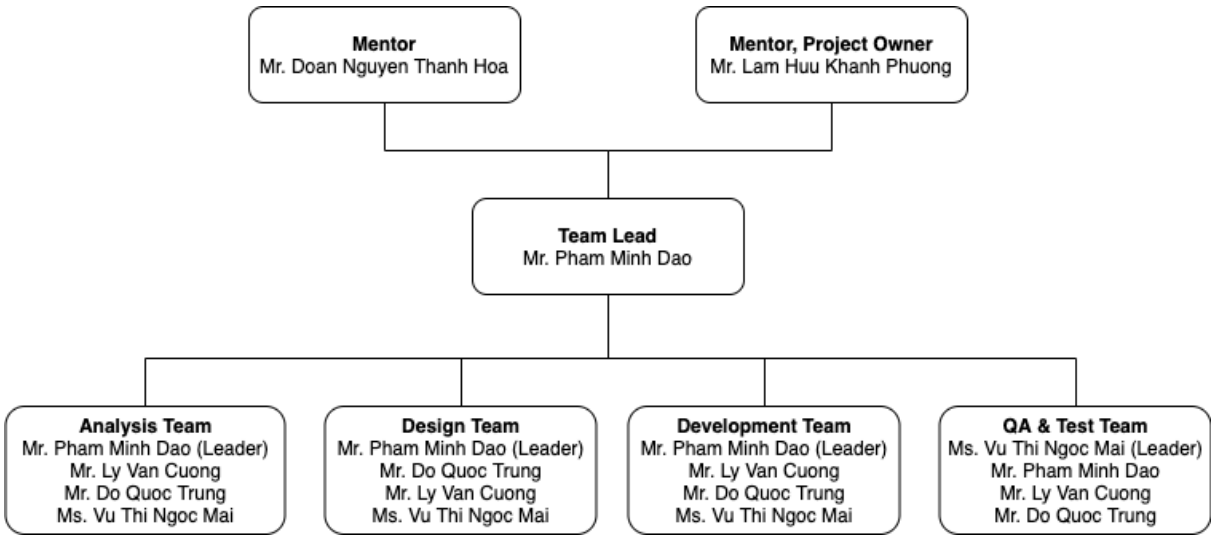


Figure 2 Team Structures

4.2 Roles & Responsibilities

Role	Responsibility
Product Owner	Review business rules Provide & Review project features
Mentor	Review project process weekly Guide technical approach Support when team struggle with problems
Team Lead	Set up environment for team work & development Arrange meeting between team members and mentor Provide tasks and estimate risks
Analysis Leader	Clarify business points to team members and ensure they understand. Review team members process to make sure if they do right
Analysis Member	Write business rules Write user stories Write functional requirements
Design Leader	Design & Review diagrams Review UI designs
Design Member	Design UI for web & mobile application
Development Leader	Define coding conventions and working rules Define technology stack

	Ensure team members doing right to the product features
Test Leader	Define general overview for testing the product Review test report
Test Member	Write test cases Test & Report

5. Project Communication

5.1 Communication Plan

Communication Item	Who/ Target	Purpose	When, Frequency	Type, Tool, Method(s)
Meeting Mentor	Lâm Hữu Khánh Phương, Đoàn Nguyễn Thành Hoà	Review project business, project process and guidelines.	Every Tuesday at 17h45 and Friday at 7h45	Messenger, google meet
Meeting team	Team members	Review tasks, problems discussion.	Daily	Google meet, messenger, slack

5.2 External Interface

a. FU Contacts

Function	Contact Person (name, position)	Contact address (email, telephone)	Responsibility
Supervisor	Doan Nguyen Thanh Hoa	hoadnt@fe.edu.vn	- Provide document template - Guide scheduling algorithm

			- Guide AI face detection implementation.
Supervisor	Lam Huu Khanh Phuong	phuonglhk@fe.edu.vn	- Receive project reports - Answer questions about the project business - Give instruction to project team - Supervise project status

6. Configuration Management

6.1 Tools & Infrastructures

Programming languages	C#, Python, JavaScript, TypeScript, dart
Framework	ASP.Net Core, ReactJs, flutter, Flask
API	Restful API
DBMS	SQL Server
IDEs/Editors	Visual Studio Code, Visual Studio, Android Studio
UML tools	draw.io, StarUML
Version Control	Github
Deployment server	Microsoft Azure, heroku, AWS, Windows 10
Project management tool	Trello, Google Drive

6.2 Document Management

Documents management using Google Drive.

Reference: https://www.google.com/drive

6.3 Source Code Management

Source code management using Github.

Reference: https://github.com

III. Software Requirement Specification

1. Overall Description

1.1 Product Overview

The Scheduling working time and Timekeeping by face recognize for Staff is a system that helps Managers manage staff, schedules and attendances. The system is expected to handle the traditional process of scheduling and timekeeping manually.

1.2 Business Rules

BR1: Business rule for Brand.

BR2: Business rule for Store.

BR3: Business rule for Brand Manager.

BR4: Business rule for Store Manager.

BR5: Business rule for Staff.

BR6: Business rule for Timekeeping.

BR7: Business rule for Schedule.

BR8: Business rule for Store face recognition device.

BR9: Business rule for System.

ID	Rule Definition
BR1 - Business rule for Brand	
BR1-01	One brand can have many store managers.
BR1-02	One brand can have many stores.
BR2 - Business for Store	
BR2-01	Store’s shifts are flexible.
BR2-02	One store can have more than one manager.
BR3 - Business rule for Brand Manager	
BR3-01	Brand managers can create more stores if the brand has multiple stores (when they first open an account) or if the brand just opens more stores (additional).
BR3-02	Staff’s skill applied to all stores under the brand.
BR3-03	One Brand can have more than one manager.
BR4 - Business rule for Store Manager	
BR4-01	Store managers can create accounts for staff.
BR4-02	The number of staff, skill in one shift is set by store managers.
BR4-03	Before the store manager approves the employee's request for leave, a replacement must be arranged first.
BR4-05	The store manager can request additional shifts for staff if staff do not register to take time off at the time of need.
BR4-06	Store manager can’t update staff’s username
BR4-07	Store managers must require employees to register enough and appropriate available time to prepare for next week's schedule. If the employee's available time is insufficient and appropriate, the store manager asks the employee to re-register the available time.
BR5 - Business rule for Staff	
BR5-01	Staff can sign up for periods during which they cannot work during the week. The calendar will not schedule staff to work at this time. If there is no schedule before.
BR5-02	Staff can not view shifts of other Staff
BR5-03	Staff are only allowed to leave if the store manager accepts the leave request.

BR5-04	The working schedule of each staff member includes the working day, the shift from which time frame to which time frame, the store address and the skill they will do.
BR5-05	Requests to add shifts, change shifts must be made before the shift starts (6 hours) and after the time period from the published calendar date.
BR5-06	Staff can only register for his/her shifts for the next week.
BR5-07	Staff can have more than one skill
BR5-08	Requests to change shifts must be accepted by another staff member and store manager
BR5-09	Staff must register enough available time for schedule to run probably
BR6 - Business rule for Timekeeping	
BR6-01	If a staff leaves a shift without permission or the manager does not approve it, it is counted as an unexcused leave.
BR6-02	Staff who do not attend during their shift are marked as absent.
BR6-03	Staff who leave before the shift ends will be marked as leaving early.
BR6-04	Staff who take attendance after the shift starts will be marked as late.
BR6-05	Attendances only recorded within 15 minutes when the shift starts or ends.
BR6-06	If staff forgot to check in or check out, the result will be recorded to report when the store manager wants to view.
BR7 - Business rule for Schedule	
BR7-01	In a shift, all necessary professional positions must be ensured.
BR7-02	The staff only works one shift at one store at a time.
BR7-03	Work schedules in the past cannot be changed.
BR7-04	The published calendar, the manager can still update based on possible surprises (employees are busy having to leave or change shifts).
BR7-05	Next week's schedule should be published latest on Sunday of this week.
BR7-06	Each shift at each skill needs a number of employees within the specified range.

BR7-07	Number of shifts / working time of staff in a week must be within the specified range.
BR7-08	Shifts assigned to staff must be within one day.
BR7-09	Working day starts on monday.
BR7-10	Schedule demand depends on store’s operating hours.
BR7-11	Shifts must be within one day.
BR7-12	Can not assign staff to work in the past.
BR7-13	Schedule is constructed for one week.
BR7-14	When a weekly schedule is created , its default status is “Unpublished”.
BR7-15	Time in one day is divided in to 48 time frame, each time frame is 30 minutes
BR7-16	One week schedule can have more than one plan.
BR7-17	One week schedule only has at most one “published” plan.
BR7-18	One week schedule only has at most one plan with status “register”.
BR8 - Business rule for Store Face Recognize Device	
BR8-01	When staff take attendance they must not wear a hat or face mask.
BR8-02	When taking attendance, the staff's face is in a direct position with the camera.
BR8-03	Staff can take his/her photo only when the face is detected.
BR8-04	Staff register his/her face identity by face video record.
BR9 - Business rule for System	
BR9-01	The system requires an internet connection.
BR9-02	When creating a new staff or store manager account, the system will send login information to the registered email.
BR9-03	Salary-related features are out of system scope.
BR9-04	User password must be at least 6 characters.
BR9-05	Passwords are hashed with the SHA-512 algorithm.
BR9-06	User email can be duplicated.
BR9-07	New password request will be sent to email.

BR9-08	Confirmation pop-up will show when the user deletes something.
BR9-09	Only get entities when its status is not “deleted”.
BR9-10	Store Manager account created by Brand Manager.
BR9-11	Staff account created by StoreManager or Brand Manager.
BR9-12	User login successful will be redirected to the home page.
BR9-13	Staff Mobile Application theme is dark or light based on the default setting in his/her phone.
BR9-14	Get attendance list default return all attendances from beginning of the week to current day.
BR9-15	Get shift list default return from beginning of the week to current day.
BR9-16	Get staff's report default return result of the first staff from the beginning of the week to the current day.
BR9-17	Get store's report default return from the beginning of the week to the current day.
BR9-18	Staff Mobile Application will get notified when he/she receives new assignments or attendance records.
BR9-19	The system currently only uses English.

2. User Requirements

2.1 Overview

a. Use Case Diagram

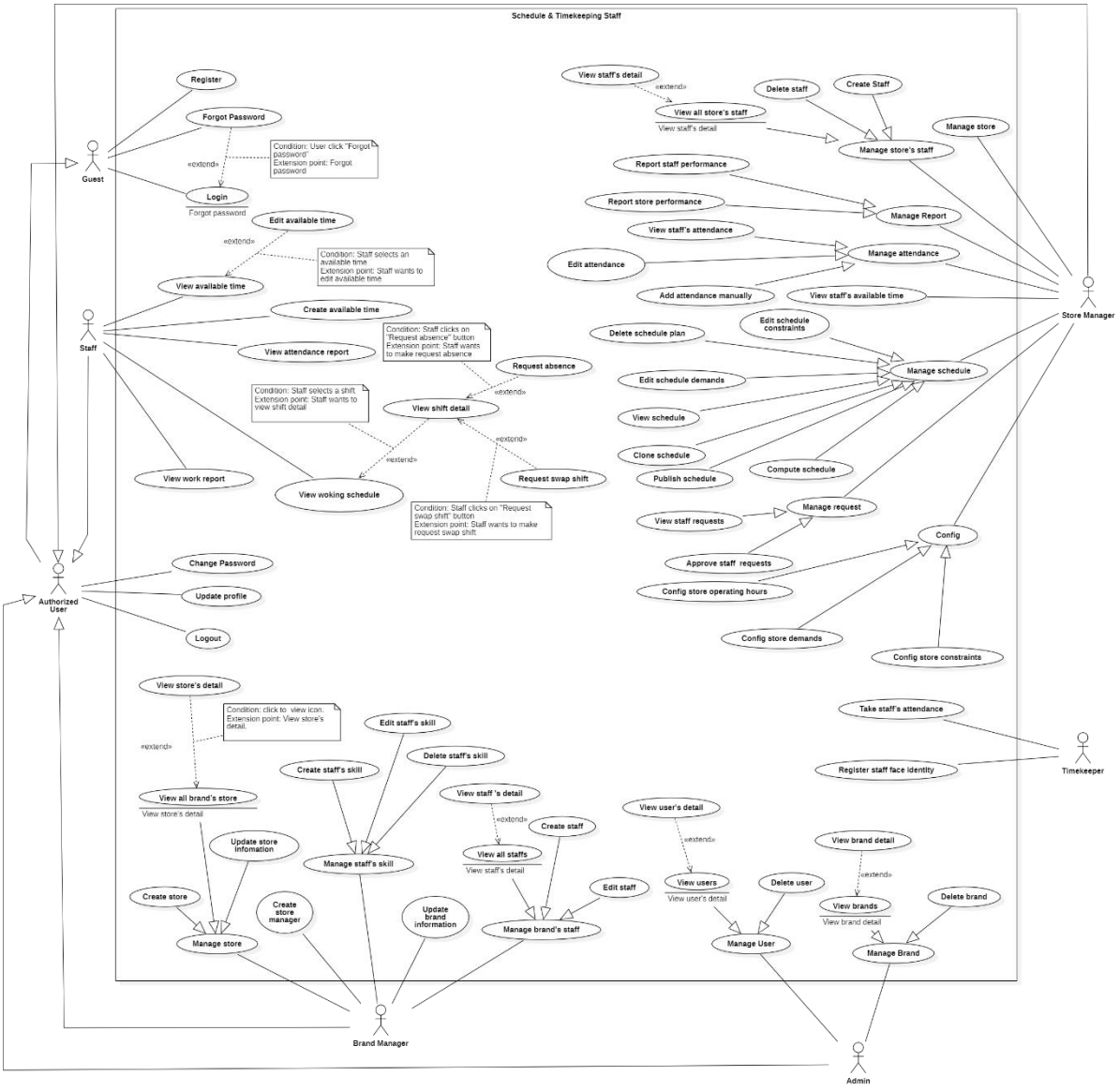


Figure 3 <Use Case Overview>STS Use Case Diagram

b. System Actors

#	Actor	Description
1	Web Admin	Web admin manages users and brands in the system.
2	Brand Manager	Brand manager that had an account in the system. They use the system to manage their store and staff, set up information for the brand, and track staff attendance.
3	Store Manager	Store manager that had an account in the system. Their account was created by the Brand Manager. They use the system to manage staff in their store, create schedules automatically or manually through the system, track or edit staff attendance.
4	Staff	Staff had an account in the system. Their account was created by the Store Manager or Brand Manager. They use system to take attendance, register time to work, view schedule, view
5	Guest	Guests that don't have an account in the system.
6	Authenticated User	Authenticated users have authorization to use the system and have accounts in the system.
7	Timekeeper	Timekeeper is used to take staff's attendance.

c. Use Cases List

ID	Use Case	Primary Actors	Secondary Actors
MD1. Guest Module			
MD1.01	Register	Guest	
MD1.02	Login	Guest	
MD1.03	Forgot password	Guest	
MD2. Authentication Module			
MD2.01	Update profile	Authenticated User	
MD2.02	Change Password	Authenticated User	
MD2.03	Logout	Authenticated User	

MD3. Brand Manager Module			
MD3.01	Create store manager	Brand Manager	
MD3.02	Create store	Brand Manager	
MD3.03	Create staff	Brand Manager	
MD3.04	Update brand information	Brand Manager	
MD3.05	Update store information	Brand Manager	
MD3.06	Create staff's skill	Brand Manager	
MD3.07	Edit staff's skill	Brand Manager	
MD3.08	Delete staff's skill	Brand Manager	
MD3.09	View attendance statistics.	Brand Manager	
MD3.10	Delete staff	Brand Manager	
MD3.11	View brand's store	Brand Manager	
MD3.12	View brand's staff	Brand Manager	
MD3.13	Delete store	Brand Manager	
MD4. Store Manager Module			
MD4.01	Create staff	Store Manager	
MD4.02	View staff available time	Store Manager	
MD4.03	Update store	Store Manager	
MD4.04	View store's staff	Store Manager	
MD4.05	View store information	Store Manager	
MD4.06	Update staff	Store Manager	
MD4.07	Edit schedule constraints	Store Manager	
MD4.08	Edit schedule demands	Store Manager	
MD4.09	View schedule	Store Manager	
MD4.10	Clone schedule	Store Manager	
MD4.11	Publish schedule	Store Manager	
MD4.12	Delete schedule plan	Store Manager	

MD4.13	View staff's attendance	Store Manager	
MD4.14	Add attendance manually	Store Manager	
MD4.15	View staff requests	Store Manager	
MD4.16	Approve staff absence request	Store Manager	
MD4.17	Approve staff change shift request	Store Manager	
MD4.18	Report staff performance	Store Manager	
MD4.19	Report store performance	Store Manager	
MD4.20	Config store operating hours	Store Manager	
MD4.21	Config store constraints	Store Manager	
MD4.22	Config store demands	Store Manager	
MD4.23	Compute schedule	Store manager	
MD5. Staff Module			
MD5.01	View working schedule	Staff	
MD5.02	View shift detail	Staff	
MD5.03	Request absence	Staff	
MD5.04	Request swap shift	Staff	
MD5.05	View available time	Staff	
MD5.06	Create available time	Staff	
MD5.07	Edit available time	Staff	
MD5.08	View attendance report	Staff	
MD5.09	View work report	Staff	
MD6. Timekeeper			
MD6.01	Take staff's attendance	Timekeeper	
MD6.02	Register staff face identity	Timekeeper	
MD7. Admin			
MD7.01	View users	Admin	
MD7.02	Add User	Admin	
MD7.03	View User Detail	Admin	
MD7.04	Delete user	Admin	
MD7.05	View Brands	Admin	
MD7.06	View Brand Detail	Admin	
MD7.07	Delete Brand	Admin	

2.2 MD1- Guest Module

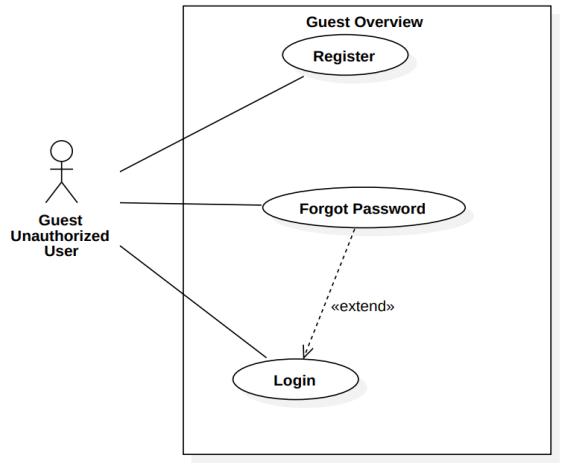


Figure 4 <Use Case Overview> Guest Overview

2.2.1. Register.

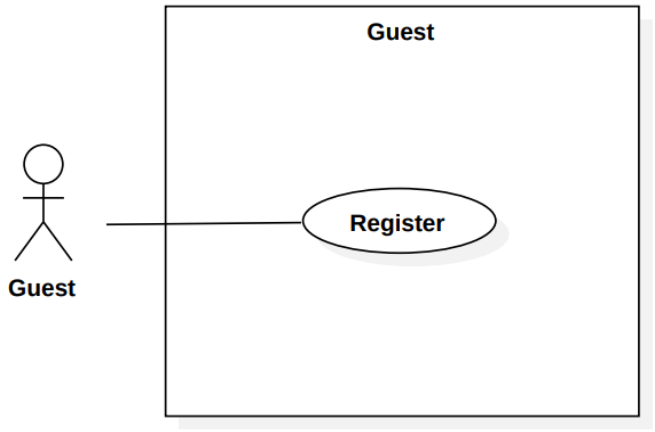


Figure 5 <Use Case> Register

<table>
<tr><td>UC ID and Name:</td><td colspan="3">MD1.1 Register</td></tr>
<tr><td>Created By:</td><td>MaiVTN</td><td>Date Created:</td><td>20/05/2021</td></tr>
<tr><td>Primary Actor:</td><td>Guest</td><td>Secondary Actors:</td><td>N/A</td></tr>
<tr><td>Trigger:</td><td colspan="3">Guest sends a request to register.</td></tr>
<tr><td>Description:</td><td colspan="3">When guests want to use the system, they must create a new account with the “brand manager” role and brand information.</td></tr>
<tr><td>Preconditions:</td><td colspan="3">PRE-1: The actor click “Register now” in Login Page.</td></tr>
<tr><td>Post-conditions:</td><td colspan="3">POST-1: success: create a new brand manager account.
fail: Show error message</td></tr>
<tr><td>Normal Flow:</td><td colspan="3">
<table>
<tr><th>Step</th><th>Actor Action</th><th>System Response</th></tr>
<tr><td>1</td><td>Fill all information in Register Page include “Email”, “User Name”, “FirstName”, “Last Name”, “Date of birth”, “Gender”, “Phone”, “Address”; brand information: “Brand Name”, “Hotline”, “Brand Address”</td><td></td></tr>
<tr><td>2</td><td>Press “Register”</td><td>- Check to see if the actor's input information is valid.
- Create a new brand manager account and redirect to the Login Page.</td></tr>
</table>
</td></tr>
<tr><td>Alternative Flows:</td><td colspan="3">N/A</td></tr>
<tr><td>Exceptions:</td><td colspan="3"></td></tr>
</table>

<table>
<tr><td></td><td>
<table>
<tr><th>No</th><th>Cause</th><th>System Response</th></tr>
<tr><td>1</td><td>Fields are required.</td><td>System returns error message label that fields is required</td></tr>
<tr><td>2</td><td>Username must be unique</td><td>System returns an error message “Username already exists”.</td></tr>
<tr><td>3</td><td>Brandname must be unique</td><td>System returns error message “Brandname already exists”</td></tr>
<tr><td>4</td><td>Day of birth must be in the past.</td><td>System returns error message “Invalid Day of birth”</td></tr>
<tr><td>5</td><td>Username must be more than 4 characters.</td><td>System returns an error message “Username must be more than 4 characters”.</td></tr>
<tr><td>6</td><td>Password must be more than 6 characters.</td><td>System returns an error message “Password must be more than 6 characters”.</td></tr>
</table>
</td></tr>
<tr><td>Priority:</td><td>High</td></tr>
<tr><td>Frequency of Use:</td><td>Medium</td></tr>
<tr><td>Business Rules:</td><td>BR1-03, BR1-04, BR3-08
Register contains:
- User Name: Text input, more than 4 characters, required.
- First Name: Text input.
- Last Name: Text input.
- Date of birth: must be before the current date.
- Gender: Dropdown button: Male or Female.
- Email: text input, must contain ‘@’, required
- Phone: Number input, 10 or 11 characters, required.
- Address: Text input, required</td></tr>
</table>

	- **Brand Name**: text input, required. - **Hotline**: number input, required. - **Brand Address**: string, required.
Other Information:	N/A
Assumptions:	N/A

Table 1 <Use Case> Register

2.2.1. Login.

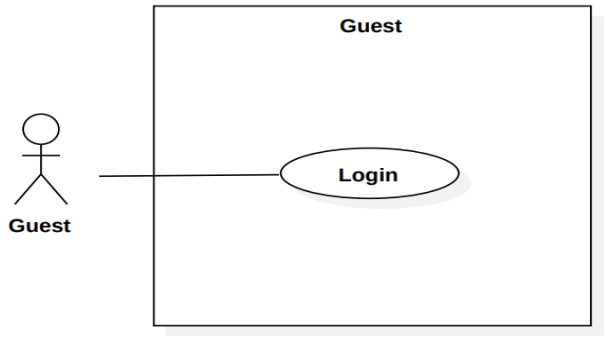


Figure 6 Login

UC ID and Name:	**MD1.2 Login**		
Created By:	MaiVTN	Date Created:	20/05/2021
Primary Actor:	Guest	Secondary Actors:	N/A
Trigger:	User sends a request to Login.		
Description:	The user who has an authentication account can login to the system.		
Preconditions:	PRE-1: For **Brand Manager** or **Store Manager** who is authorized need to use Web application. PRE-2: For **Staff** who are authorized need to use Mobile Application.		
Post-conditions:	POST-1: success: the actor is authenticated, logged and redirected to Home Page. fail: the actor is not authenticated, show error message.		
Normal Flow:			

Step	Actor Action	System Response
1	Fill “User Name”, “Password”	Check to see if the actor's input information is valid or not.
2	Press “Login”	- Check authentication,

<table>
<tr><td></td><td><table>
<tr><td></td><td></td><td>redirect to Home Page (web app) or Home Screen (mobile app).</td></tr>
</table></td></tr>
<tr><td>Alternative Flows:</td><td>N/A</td></tr>
<tr><td>Exceptions:</td><td><table>
<tr><th>No</th><th>Cause</th><th>System Response</th></tr>
<tr><td>1</td><td>Fields are required.</td><td>System returns error message label that fields is required</td></tr>
<tr><td>2</td><td>Username must exist.</td><td>System returns an error message “Invalid username”.</td></tr>
<tr><td>3</td><td>Password must be correct.</td><td>System returns error message “Incorrect password”</td></tr>
</table></td></tr>
<tr><td>Priority:</td><td>High</td></tr>
<tr><td>Frequency of Use:</td><td>High</td></tr>
<tr><td>Business Rules:</td><td>BR3-01, BR3-02, BR3-09, BR4-01
Login Page contains:
- User Name: Text input, required.
- Password: password input, required.</td></tr>
<tr><td>Other Information:</td><td>N/A</td></tr>
<tr><td>Assumptions:</td><td>N/A</td></tr>
</table>

Table 2 <Use Case> Login

2.3 MD2- Authentication Module

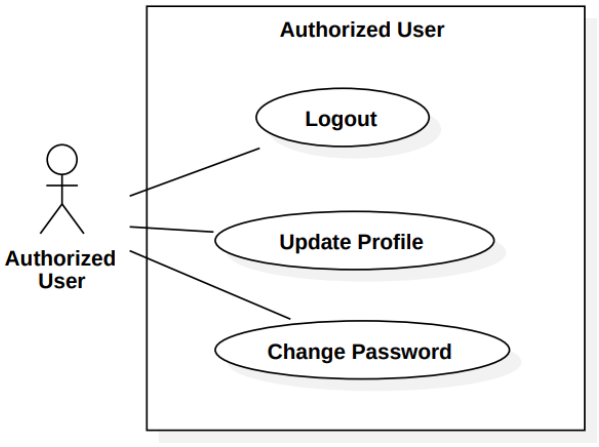


Figure 7 <Use Case Overview> Authentication Use Case

2.3.1. Update profile.

<table>
<tr><td>UC ID and Name:</td><td colspan="3">MD2.1 Update Profile</td></tr>
<tr><td>Created By:</td><td>MaiVTN</td><td>Date Created:</td><td>20/05/2021</td></tr>
<tr><td>Primary Actor:</td><td>Authenticated User</td><td>Secondary Actors:</td><td>N/A</td></tr>
<tr><td>Trigger:</td><td colspan="3">Authenticated Users requested to edit their information.</td></tr>
<tr><td>Description:</td><td colspan="3">The user having an authentication account has the ability to update their profile in the system.</td></tr>
<tr><td>Preconditions:</td><td colspan="3">PRE-1: the actors who are authorized.
PRE-2: “Update profile” screen.</td></tr>
<tr><td>Post-conditions:</td><td colspan="3">POST-1: success: the actor’s information is edited.
fail: the actor’s information is unchanging.</td></tr>
<tr><td>Normal Flow:</td><td colspan="3"><table><tr><th>Step</th><th>Actor Action</th><th>System Response</th></tr><tr><td>1</td><td>The actor request to Update Profile</td><td>Check to see if the actor's input information is valid or not.</td></tr><tr><td>2</td><td>The actor edits new information that want to change.</td><td>New profile.</td></tr></table></td></tr>
<tr><td>Alternative Flows:</td><td colspan="3">N/A</td></tr>
<tr><td>Exceptions:</td><td colspan="3"><table><tr><th>No</th><th>Cause</th><th>System Response</th></tr><tr><td>1</td><td>Save change with don’t</td><td>System return old profile.</td></tr></table></td></tr>
</table>

		edit anything		
Priority:	Medium			
Frequency of Use:	Low			
Business Rules:	- **User Name:** Text input, more than 4 characters, required. - **First Name:** Text input. - **Last Name**: Text input. - **Phone:** Number input, 10 or 11 characters, required. - **Address:** Text input, required			
Other Information:	N/A			
Assumptions:	N/A			

Table 3. <Use Case> Update profile

2.3.2. Logout.

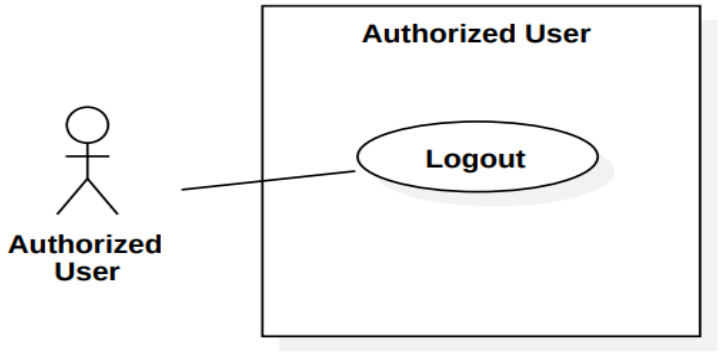


Figure 8 <Use Case> Logout

UC ID and Name:	**MD2.2 Logout**		
Created By:	MaiVTN	Date Created:	20/05/2021

<table>
<tr><td>Primary Actor:</td><td>Authenticated User</td><td>Secondary Actors:</td><td>N/A</td></tr>
<tr><td>Trigger:</td><td colspan="3">Authenticated User request logout</td></tr>
<tr><td>Description:</td><td colspan="3">The user who has an authentication account wants to log out of the system.</td></tr>
<tr><td>Preconditions:</td><td colspan="3">PRE-1: the actor who is authorized.
PRE-2: the actor press Profile section.</td></tr>
<tr><td>Post-conditions:</td><td colspan="3">POST-1: success: the actor’s logged out.
Fail: current page.</td></tr>
<tr><td>Normal Flow:</td><td colspan="3"><table><tr><th>Step</th><th>Actor Action</th><th>System Response</th></tr><tr><td>1</td><td>Click “Logout”.</td><td>Unauthenticated and return login page. Users can’t access any page else.</td></tr></table></td></tr>
<tr><td>Alternative Flows:</td><td colspan="3">N/A</td></tr>
<tr><td>Exceptions:</td><td colspan="3"><table><tr><th>No</th><th>Cause</th><th>System Response</th></tr><tr><td>1</td><td>Return to the previous page after logout.</td><td>System return Login Page.</td></tr></table></td></tr>
<tr><td>Priority:</td><td colspan="3">High</td></tr>
<tr><td>Frequency of Use:</td><td colspan="3">Medium</td></tr>
</table>

Business Rules:	N/A
Other Information:	N/A
Assumptions:	N/A

Table 4. <Use Case> Logout

2.3.3. Change Password.

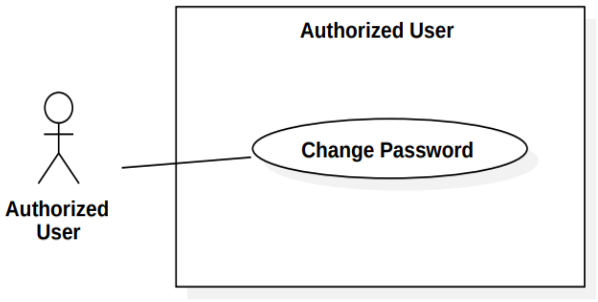


Figure 9 <Use Case> Change Password

UC ID and Name:	**MD2.3 Change Password**		
Created By:	MaiVTN	Date Created:	20/05/2021
Primary Actor:	Authenticated User	Secondary Actors:	N/A
Trigger:	Authenticated User request change password.		
Description:	The user who has an authentication account wants to change password to login in the system.		
Preconditions:	PRE-1: the actor who are authorized PRE-2: staying at the “Change Password” page.		
Post-conditions:	POST-1: success: the actor’s password has changed. fail: show error message		
Normal Flow:			

Step	Actor Action	System Response
1	Fill “Present password”, “New Password”, “Confirm New Password”	Check to see if the actor's input

<table>
<tr><td></td><td>
<table>
<tr><td></td><td></td><td>information is valid or not.</td></tr>
<tr><td>2</td><td>The actor sends a command to save the changes.</td><td>Save a new password</td></tr>
</table>
</td></tr>
<tr><td>Alternative Flows:</td><td>N/A</td></tr>
<tr><td>Exceptions:</td><td>
<table>
<tr><th>No</th><th>Cause</th><th>System Response</th></tr>
<tr><td>1</td><td>Confirm new password not match with New Password.</td><td>System return message error “Incorrect confirm password”.</td></tr>
<tr><td>2</td><td>Present password is incorrect.</td><td>System return message error “Incorrect current password”.</td></tr>
<tr><td>3</td><td>Any field is empty.</td><td>System return message error “Password/confirm password is required”.</td></tr>
</table>
</td></tr>
<tr><td>Priority:</td><td>Low</td></tr>
<tr><td>Frequency of Use:</td><td>Low</td></tr>
<tr><td>Business Rules:</td><td>- Present password: string, required.
- New Password: string, required
- Confirm New Password: string, required.</td></tr>
<tr><td>Other Information:</td><td>N/A</td></tr>
<tr><td>Assumptions:</td><td>N/A</td></tr>
</table>

Table 5. <Use Case> Change Password.

2.4 MD3- Brand Manager Module

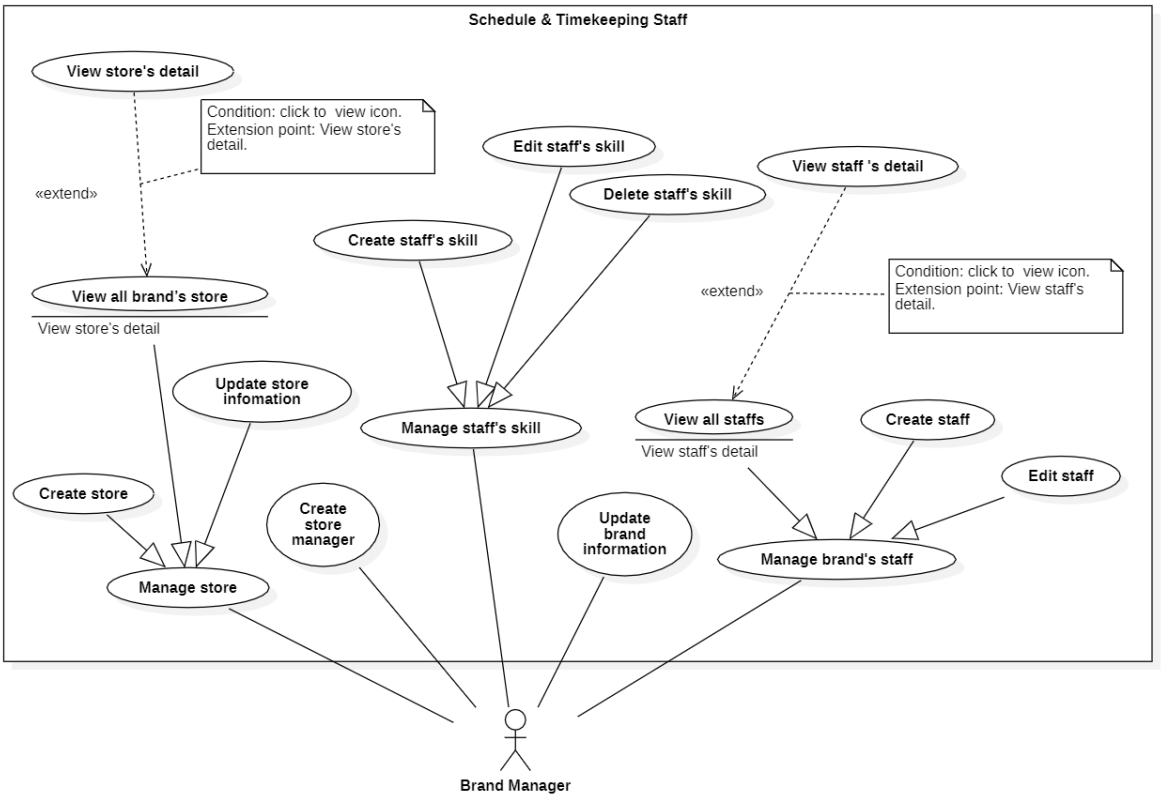


Figure 10 <Use Case Overview> Brand Manager Use Case

2.4.2. Create a store.

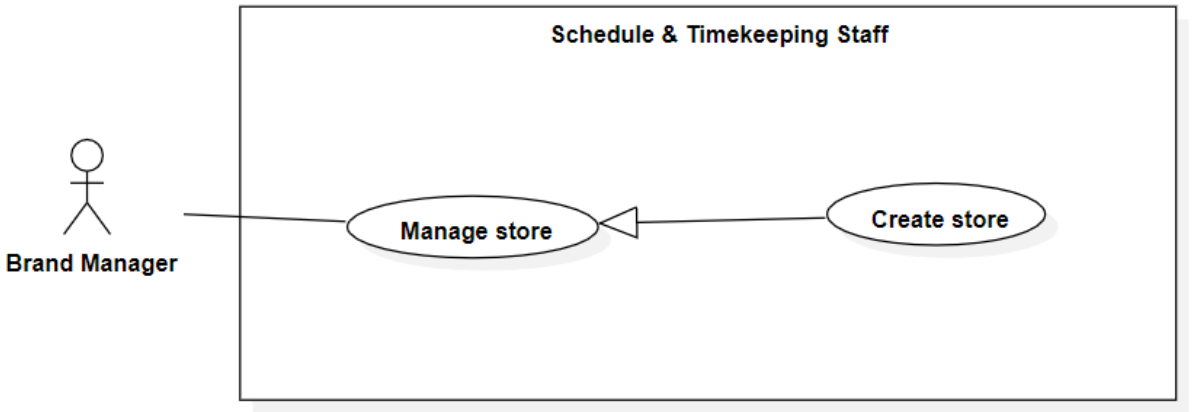


Figure 11 <Use Case> Create Store

UC ID and Name:	**MD3.2 Create store**

<table>
<tr><td>Created By:</td><td>MaiVTN</td><td>Date Created:</td><td>21/05/2021</td></tr>
<tr><td>Primary Actor:</td><td>Brand Manager</td><td>Secondary Actors:</td><td>N/A</td></tr>
<tr><td>Trigger:</td><td colspan="3">The Brand Manager requested to create a new store.</td></tr>
<tr><td>Description:</td><td colspan="3">After creating a brand, the actor can create a chain store.</td></tr>
<tr><td>Preconditions:</td><td colspan="3">PRE-1: Users has logged in as brand manager role, brand is existed</td></tr>
<tr><td>Post-conditions:</td><td colspan="3">POST-1: success: new store.
Fail: cancel process and return “Store Page”</td></tr>
<tr><td>Normal Flow:</td><td colspan="3"><table>
<tr><th>Step</th><th>Actor Action</th><th>System Response</th></tr>
<tr><td>1</td><td>The actor presses “Add Store” on screen.</td><td>Open page “Add Store”</td></tr>
<tr><td>2</td><td>Fill all information about store include: “Store Name”, “Address”, “Phone”</td><td>Check to see if the actor's input information is valid or not.</td></tr>
<tr><td>3</td><td>Press Create Store</td><td>Create new store</td></tr>
</table></td></tr>
<tr><td>Alternative Flows:</td><td colspan="3">N/A</td></tr>
<tr><td>Exceptions:</td><td colspan="3"><table>
<tr><th>No</th><th>Cause</th><th>System Response</th></tr>
<tr><td>1</td><td>Store name is empty.</td><td>System return message error “Store Name is required”.</td></tr>
<tr><td>2</td><td>Address is empty.</td><td>System return message error “Address is required”.</td></tr>
<tr><td>3</td><td>Phone is empty.</td><td>System return message error “Phone is required”.</td></tr>
</table></td></tr>
<tr><td>Priority:</td><td colspan="3">Low</td></tr>
<tr><td>Frequency of Use:</td><td colspan="3">Low</td></tr>
<tr><td>Business Rules:</td><td colspan="3">- BR3-01
- Store Name: text input, required.
- Address: text input, required</td></tr>
</table>

	- **Phone**: text input, required
Other Information:	N/A
Assumptions:	N/A

Table 6 <Use Case> Create store

2.4.3. Create staff.

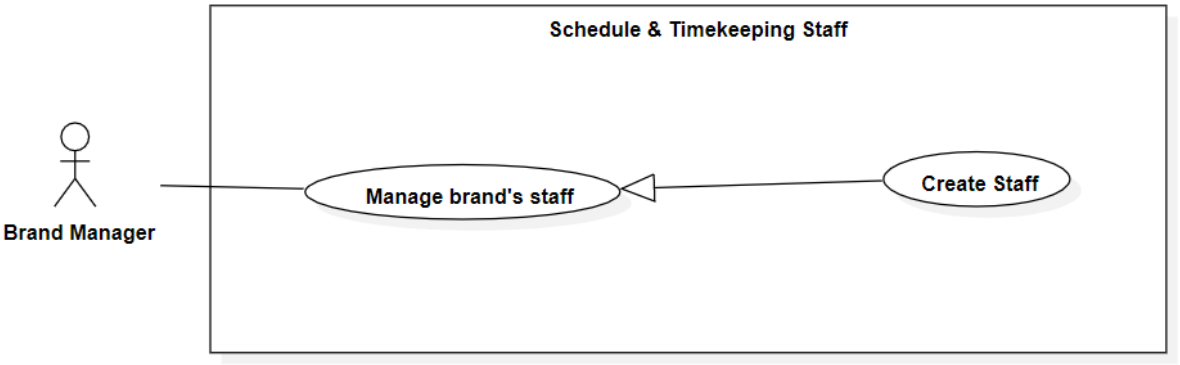


Figure 12 <Use Case> Create Staff

UC ID and Name:	**MD3.3 Create staff**		
Created By:	MaiVTN	Date Created:	21/05/2021
Primary Actor:	Brand Manager	Secondary Actors:	N/A
Trigger:	The Brand Manager requested to create new staff for the store.		
Description:	The Brand Manager creates new staff and assigns them to the store.		
Preconditions:	PRE-1: Users have logged in as brand manager or store manager role, store existed.		
Post-conditions:	POST-1: success: new staff.		
Normal Flow:			

Step	Actor Action	System Response
1	The actor press “Add Staff”	Open Add Staff Page
2	Fill information about staff include: “First name”, “Last Name”, “User Name”,	Check to see if the actor's input information is valid or not.

<table>
<tr><td></td><td>
<table>
<tr><td></td><td>“Address”, “Day of birth”, “Gender”, “Email”, “Phone”, “Wort at”, “Hired On”, “Type”, “Skill”.</td><td></td></tr>
<tr><td>3</td><td>Press “Add staff”</td><td>create new staff</td></tr>
</table>
</td></tr>
<tr><td>Alternative Flows:</td><td>N/A</td></tr>
<tr><td>Exceptions:</td><td>
<table>
<tr><th>No</th><th>Cause</th><th>System Response</th></tr>
<tr><td>1</td><td>Fields are required.</td><td>System returns error message label that fields is required</td></tr>
<tr><td>2</td><td>User name must be unique</td><td>System returns an error message that the user name is not unique.</td></tr>
<tr><td>3</td><td>Invalid inputs</td><td>System returns error message label that fields are invalid</td></tr>
</table>
</td></tr>
<tr><td>Priority:</td><td>Low</td></tr>
<tr><td>Frequency of Use:</td><td>N/A</td></tr>
<tr><td>Business Rules:</td><td>- BR3-01
- First Name: text input, required.
- Last Name: text input, required.
- User Name: text input, required.
- Address: text input, required.
- Date of birth: date time input.
- Gender: select box.
- Email: text input, required.
- Phone: text input, required.
- Wort at: select box.
- Hired On: date time input.
- Type: Select box
- Skill: select box</td></tr>
<tr><td>Other Information:</td><td>N/A</td></tr>
</table>

Assumptions:	N/A

Table 7 <Use Case> Create staff

2.4.4. Update brand information.

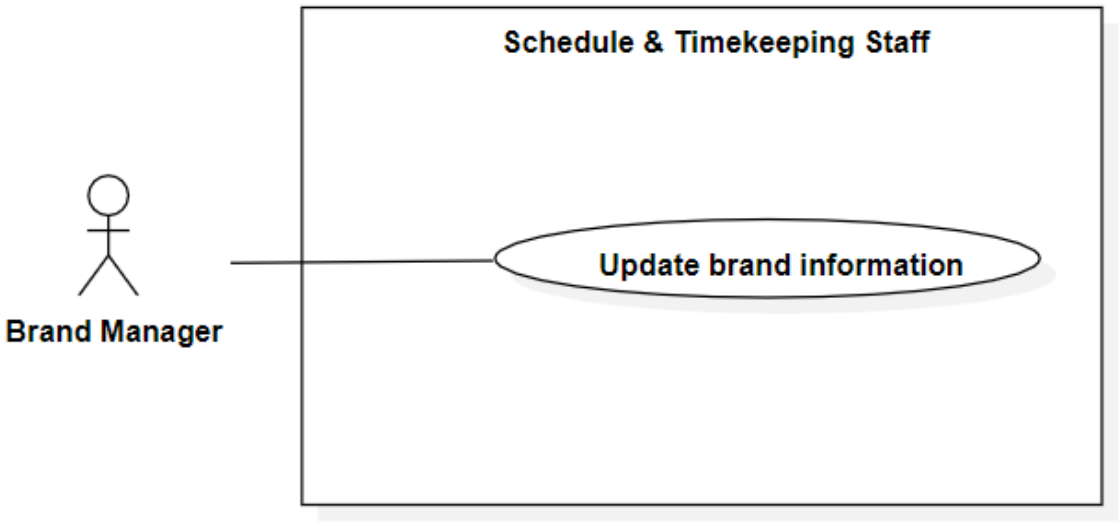


Figure 13 <Use Case> Update Brand Information

<table>
<tr><td>UC ID and Name:</td><td colspan="3">MD3.4 Update brand information</td></tr>
<tr><td>Created By:</td><td>MaiVTN</td><td>Date Created:</td><td>21/05/2021</td></tr>
<tr><td>Primary Actor:</td><td>Brand Manager</td><td>Secondary Actors:</td><td>N/A</td></tr>
<tr><td>Trigger:</td><td colspan="3">The Brand Manager requested to update brand information</td></tr>
<tr><td>Description:</td><td colspan="3">The Brand Manager can edit their brand information in case their information has changed.</td></tr>
<tr><td>Preconditions:</td><td colspan="3">PRE-1: Users have logged in as role brand manager.</td></tr>
<tr><td>Post-conditions:</td><td colspan="3">POST-1: success: new brand information.</td></tr>
<tr><td>Normal Flow:</td><td colspan="3">
<table>
<tr><th>Step</th><th>Actor Action</th><th>System Response</th></tr>
<tr><td>1</td><td>The actor inputs the information to the fields which are allowed to be edited: “Brand Name”, “Address”, “Hotline”.</td><td>The system checks to see if the user input is valid or not.</td></tr>
</table>
</td></tr>
</table>

<table>
<tr><td></td><td>
<table>
<tr><td>2</td><td>The actor sends a command to save the changes.</td><td></td></tr>
</table>
</td></tr>
<tr><td>Alternative Flows:</td><td>N/A</td></tr>
<tr><td>Exceptions:</td><td>
<table>
<tr><th>No</th><th>Cause</th><th>System Response</th></tr>
<tr><td>1</td><td>Fields are required.</td><td>System returns error message label that fields is required</td></tr>
<tr><td>2</td><td>Invalid inputs</td><td>System returns error message label that fields are invalid</td></tr>
</table>
</td></tr>
<tr><td>Priority:</td><td>Low</td></tr>
<tr><td>Frequency of Use:</td><td>Low</td></tr>
<tr><td>Business Rules:</td><td>- Brand Name: string, not null.
- Address: string.
- Hotline: string</td></tr>
<tr><td>Other Information:</td><td>N/A</td></tr>
<tr><td>Assumptions:</td><td>N/A</td></tr>
</table>

Table 8 <Use Case> Update brand information

2.4.5. Update store's information.

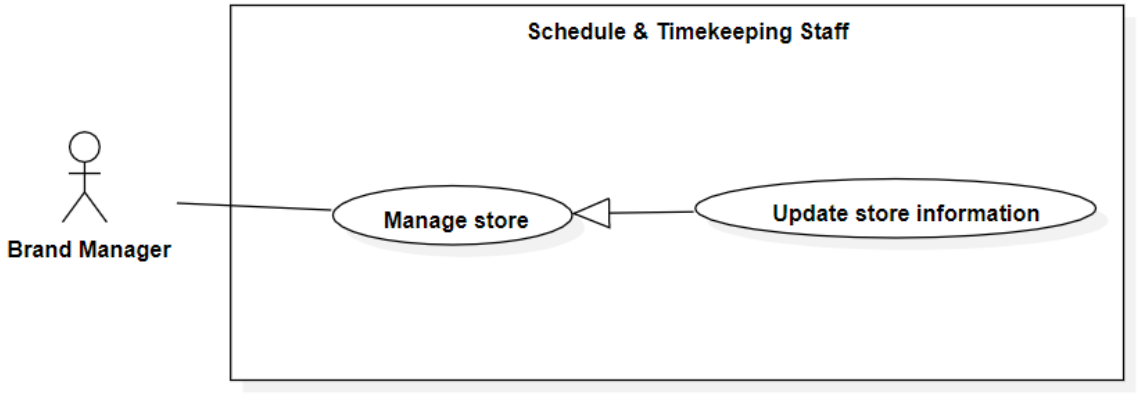


Figure 14 <Use Case> Update Store information

<table>
<tr><td>UC ID and Name:</td><td colspan="3">MD3.5 Update store’s information</td></tr>
<tr><td>Created By:</td><td>MaiVTN</td><td>Date Created:</td><td>21/05/2021</td></tr>
<tr><td>Primary Actor:</td><td>Brand Manager</td><td>Secondary Actors:</td><td>N/A</td></tr>
<tr><td>Trigger:</td><td colspan="3">The Brand Manager requested to update the store’s information.</td></tr>
<tr><td>Description:</td><td colspan="3">The Brand Manager can edit all their store information in case they have changed.</td></tr>
<tr><td>Preconditions:</td><td colspan="3">PRE-1: Users have logged in as role “brand manager”.
PRE-2:</td></tr>
<tr><td>Post-conditions:</td><td colspan="3">POST-1: success: updated store’s information.</td></tr>
<tr><td>Normal Flow:</td><td colspan="3">
<table>
<tr><th>Step</th><th>Actor Action</th><th>System Response</th></tr>
<tr><td>1</td><td>The actor inputs the information to the fields which are allowed to be edited: “Name”, “Address”, “Phone”.</td><td></td></tr>
<tr><td>2</td><td>The actor clicks “Confirm” to save the changes.</td><td>- The system checks to see if the user input is valid.
- Update store information.</td></tr>
</table>
</td></tr>
<tr><td>Alternative Flows:</td><td colspan="3">N/A</td></tr>
<tr><td>Exceptions:</td><td colspan="3">
<table>
<tr><th>No</th><th>Cause</th><th>System Response</th></tr>
<tr><td>1</td><td>Fields are required.</td><td>System returns error message label that fields is required</td></tr>
<tr><td>2</td><td>Invalid inputs</td><td>System returns error message label that fields are invalid</td></tr>
</table>
</td></tr>
</table>

Priority:	Medium
Frequency of Use:	Low
Business Rules:	Store Name can not null.
Other Information:	N/A
Assumptions:	N/A

Table 9 <Use Case> Update store's information

2.4.6. Create staff's skill.

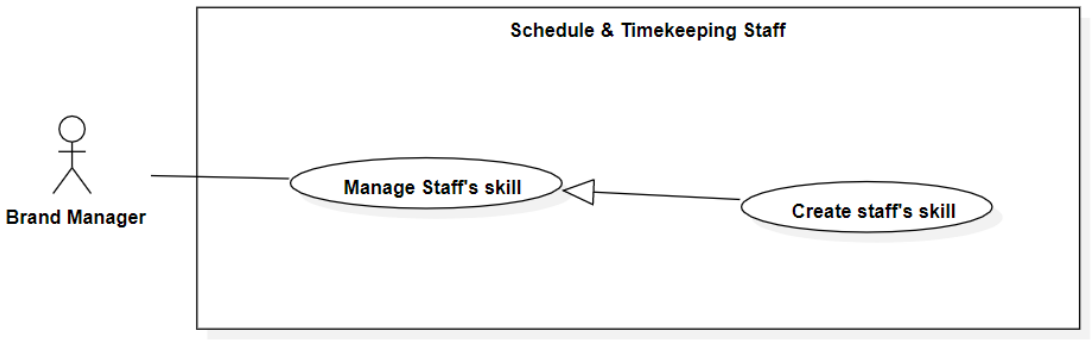


Figure 15 <Use Case> Create staff's skill

UC ID and Name:	**MD3.6 Create staff's skill**		
Created By:	MaiVTN	Date Created:	22/05/2021
Primary Actor:	Brand Manager	Secondary Actors:	N/A
Trigger:	The Brand Manager requested to create a staff's skill		
Description:	Staff's skill is general rules for all stores.		
Preconditions:	PRE-1: Users have logged in as role brand manager.		
Post-conditions:	POST-1: success: new staff's skill		
Normal Flow:			

Step	Actor Action	System Response
1	Fill "Skill Name" and "Description"	
2	Press "Add"	Create new staff's skill

<table>
<tr><td></td><td></td></tr>
<tr><td>Alternative Flows:</td><td>N/A</td></tr>
<tr><td>Exceptions:</td><td>
<table>
<tr><th>No</th><th>Cause</th><th>System Response</th></tr>
<tr><td>1</td><td>Fields are required.</td><td>System returns error message label that fields is required</td></tr>
<tr><td>2</td><td>Invalid inputs</td><td>System returns error message label that fields are invalid</td></tr>
</table>
</td></tr>
<tr><td>Priority:</td><td>High</td></tr>
<tr><td>Frequency of Use:</td><td>Medium</td></tr>
<tr><td>Business Rules:</td><td>Staff’s skill include:
- Skill Name: text input, not null.
- Description: text input.</td></tr>
<tr><td>Other Information:</td><td>N/A</td></tr>
<tr><td>Assumptions:</td><td>N/A</td></tr>
</table>

Table 10 <Use Case> Create staff's skill

2.4.7. Edit staff skill.

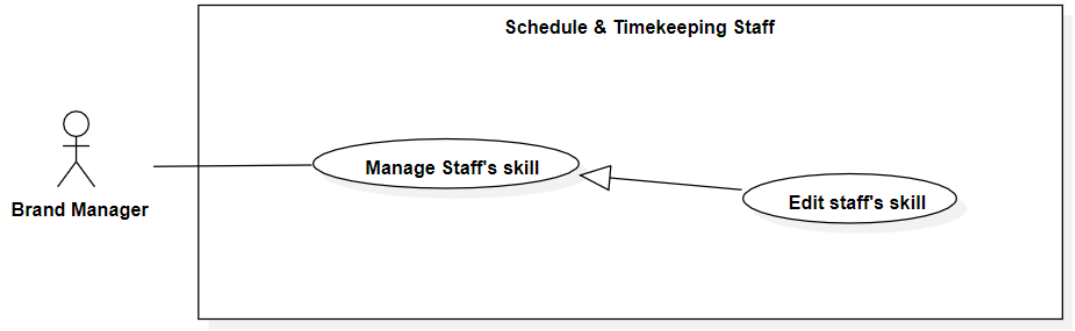


Figure 16 <Use Case> Edit staff's skill

UC ID and Name:	**MD3.7 Edit staff’s skill**		
Created By:	MaiVTN	Date Created:	22/05/2021
Primary Actor:	Brand Manager	Secondary Actors:	N/A
Trigger:	The Brand Manager requested to update the staff's skill.		
Description:	Brand Manager can edit staff’s skill.		

<table>
<tr><td>Preconditions:</td><td>PRE-1: Users have logged in as role brand manager, staff’s skill exists.</td></tr>
<tr><td>Post-conditions:</td><td>POST-1: success: new store.</td></tr>
<tr><td>Normal Flow:</td><td>
<table>
<tr><th>Step</th><th>Actor Action</th><th>System Response</th></tr>
<tr><td>1</td><td>Fill “Skill Name” and “Description"</td><td></td></tr>
<tr><td>2</td><td>Click “Update” button</td><td>Staff’s skill updated.</td></tr>
</table>
</td></tr>
<tr><td>Alternative Flows:</td><td>N/A</td></tr>
<tr><td>Exceptions:</td><td>
<table>
<tr><th>No</th><th>Cause</th><th>System Response</th></tr>
<tr><td>1</td><td>Fields are required.</td><td>System returns error message label that fields is required</td></tr>
<tr><td>2</td><td>Invalid inputs</td><td>System returns error message label that fields are invalid</td></tr>
</table>
</td></tr>
<tr><td>Priority:</td><td>Low</td></tr>
<tr><td>Frequency of Use:</td><td>N/A</td></tr>
<tr><td>Business Rules:</td><td>Update staff’s skill include:
- Skill Name: text input, required.
- Description: text input.</td></tr>
<tr><td>Other Information:</td><td>N/A</td></tr>
<tr><td>Assumptions:</td><td>N/A</td></tr>
</table>

Table 11 <Use Case> Edit staff's skill

2.4.8. Delete staff's skill.

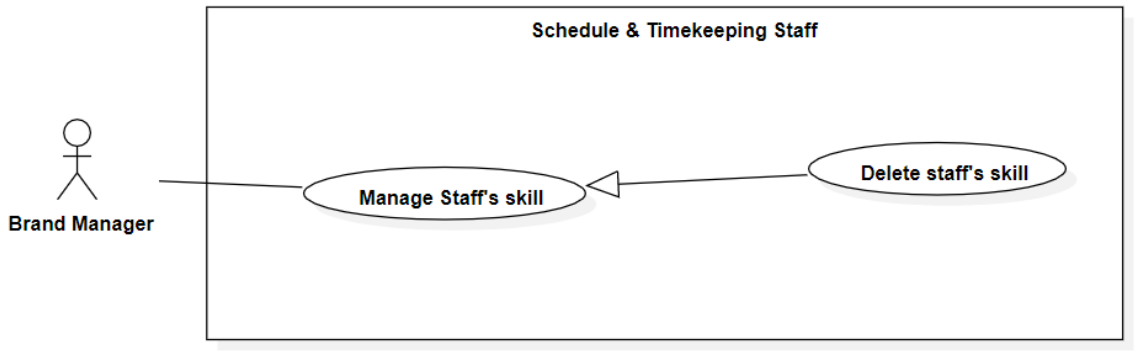


Figure 17 <Use Case> Delete staff's skill

UC ID and Name:	**MD3.8 Delete staff's skill**		
Created By:	MaiVTN	Date Created:	23/05/2021
Primary Actor:	Brand Manager	Secondary Actors:	N/A
Trigger:	The Brand Manager requested to delete the staff's skill.		
Description:	The Brand Manager deleted the staff's skill.		
Preconditions:	PRE-1: User has logged in as brand manager role, PRE-2: User at setting brand skill page PRE-3: Staff's skill is existed		
Post-conditions:	POST-1: success: remove staff's skill		
Normal Flow:			

Step	Actor Action	System Response
1	The actor presses "Delete" on the skill to delete.	- Delete this skill.

Alternative Flows:	N/A
Exceptions:	N/A
Priority:	Medium
Frequency of Use:	Low
Business Rules:	N/A

Other Information:	N/A
Assumptions:	N/A

Table 12 <Use Case> Delete staff's skill

2.4.9 View attendance statistics.

UC ID and Name:	**MD3.9 View attendance statistics**		
Created By:	MaiVTN	Date Created:	24/05/2021
Primary Actor:	Brand Manager	Secondary Actors:	N/A
Trigger:	The Brand Manager requested to view the Home Page		
Description:	The Brand Manager can view attendance statistics, information of stores that start earliest, end earliest, start latest, end latest.		
Preconditions:	PRE-1: Users have logged in as role brand manager.		
Post-conditions:	POST-1: success: Screen with attendance statistics		
Normal Flow:			

Step	Actor Action	System Response
1	Login in the system	Show Home Page which attendance statistics

Alternative Flows:	N/A
Exceptions:	N/A
Priority:	Medium
Frequency of Use:	High

Business Rules:	N/A
Other Information:	N/A
Assumptions:	N/A

Table 13 <Use Case> View attendance statistics

2.4.10 Delete staff

UC ID and Name:	**MD3.10 Delete staff**		
Created By:	MaiVTN	Date Created:	24/05/2021
Primary Actor:	Brand Manager	Secondary Actors:	N/A
Trigger:	The Brand Manager requested to delete staff		
Description:	The Brand Manager deletes staff when they stop working.		
Preconditions:	PRE-1: Users has logged in as brand manager role PRE-2: stay at Staff screen		
Post-conditions:	POST-1: success: inactive staff		
Normal Flow:			

Step	Actor Action	System Response
1	Click delete icon	Show pop-up “Are you sure to delete staff?”
2	Click “Confirm”	Change the staff's status to “deleted”

Alternative Flows:	N/A
Exceptions:	N/A
Priority:	High
Frequency of Use:	Medium
Business Rules:	N/A

Other Information:	N/A
Assumptions:	N/A

Table 14 <Use Case> Delete staff

2.4.11 View brand's store

UC ID and Name:	**MD3.11 View brand's store**		
Created By:	MaiVTN	Date Created:	03/06/2021
Primary Actor:	Brand Manager	Secondary Actors:	N/A
Trigger:	The Brand Manager requested to view their store.		
Description:	View all stores and manage them.		
Preconditions:	PRE-1: Users has logged in as brand manager role PRE-2: screen Store.		
Post-conditions:	POST-1: success: table with store information.		
Normal Flow:			

Step	Actor Action	System Response
1	Click "Store" in the dashboard.	Redirect to page "Store"

Alternative Flows:	N/A
Exceptions:	N/A
Priority:	Medium
Frequency of Use:	High
Business Rules:	N/A
Other Information:	N/A
Assumptions:	N/A

Table 15 <Use Case> View brand's store

2.3.12 View brand's staff

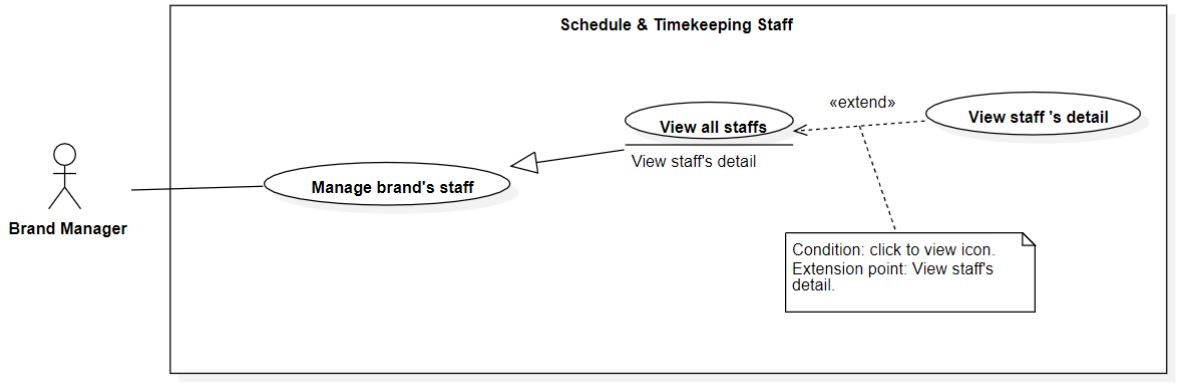


Figure 18 <Use Case> View brand's staff

UC ID and Name:	**MD3.12 View brand's staff**		
Created By:	MaiVTN	Date Created:	03/06/2021
Primary Actor:	Brand Manager	Secondary Actors:	N/A
Trigger:	The Brand Manager requested to view all staff in their brand.		
Description:	The Brand Manager views all staff in their brand.		
Preconditions:	PRE-1: Users has logged in as brand manager role PRE-2: Staff Page		
Post-conditions:	POST-1: success: table with staff information.		
Normal Flow:			

Step	Actor Action	System Response
1	Click "Staff" in the dashboard.	Show all store

Alternative Flows:	N/A
Exceptions:	N/A
Priority:	Low
Frequency of Use:	N/A
Business Rules:	N/A

Other Information:	N/A
Assumptions:	N/A

Table 16 <Use Case> View brand's staff

2.3.13 Delete store

UC ID and Name:	**MD3.13 Delete store**		
Created By:	MaiVTN	Date Created:	03/06/2021
Primary Actor:	Brand Manager	Secondary Actors:	N/A
Trigger:	The Brand Manager wants to delete a store.		
Description:	The Brand Manager wants to delete a store.		
Preconditions:	PRE-1: Users has logged in as brand manager role		
Post-conditions:	POST-1: success: store deleted.		
Normal Flow:			

Step	Actor Action	System Response
1	Click the "delete" icon on the store you want to delete.	Delete store is chosen.

Alternative Flows:	N/A
Exceptions:	N/A
Priority:	Medium
Frequency of Use:	Low
Business Rules:	N/A
Other Information:	N/A
Assumptions:	N/A

Table 17 <Use Case> Delete store

2.5 MD4. Store Manager Module

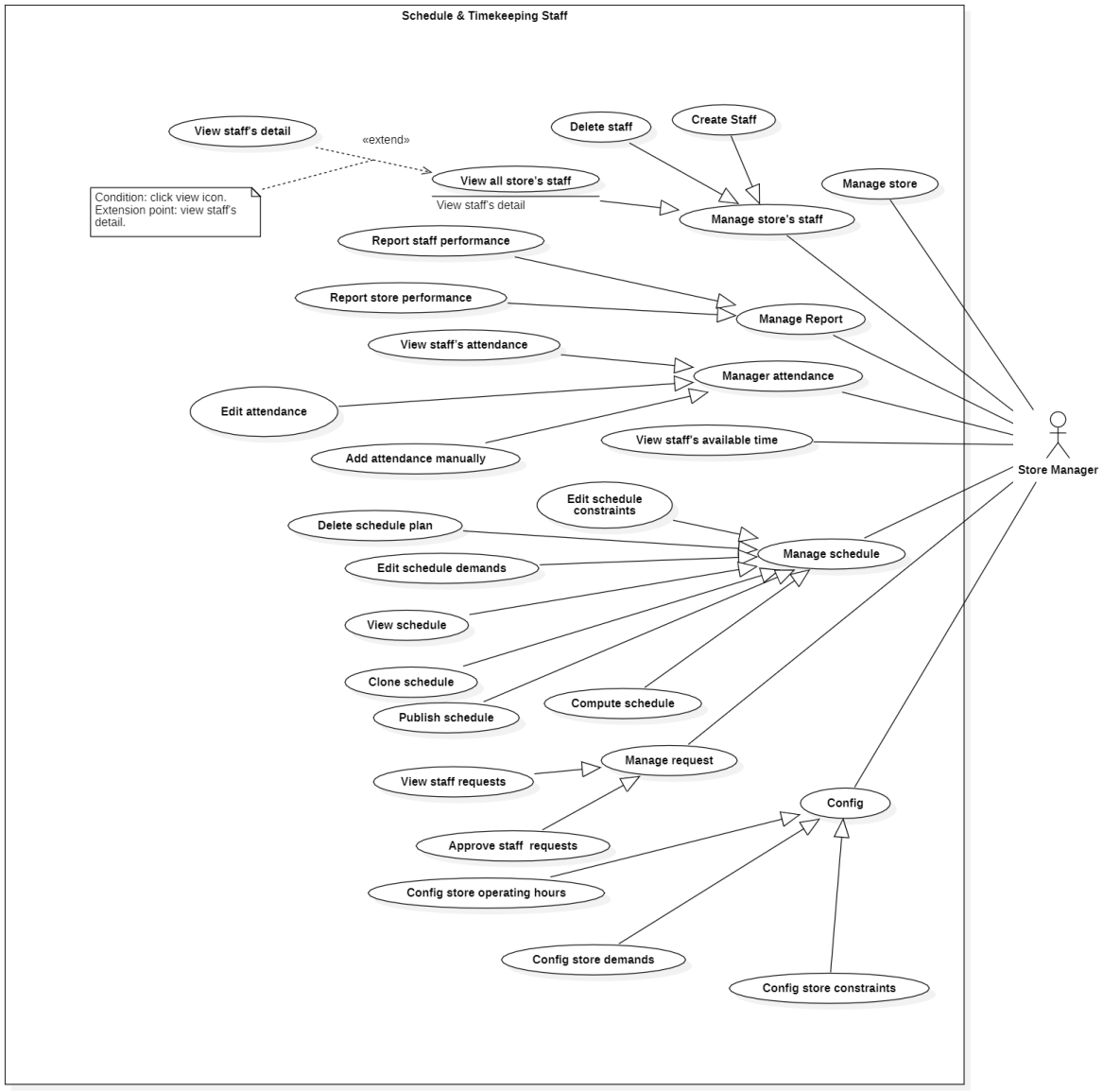


Figure 19 <Use Case Diagram> Store Manager Use Case Diagram

2.5.1 Create staff

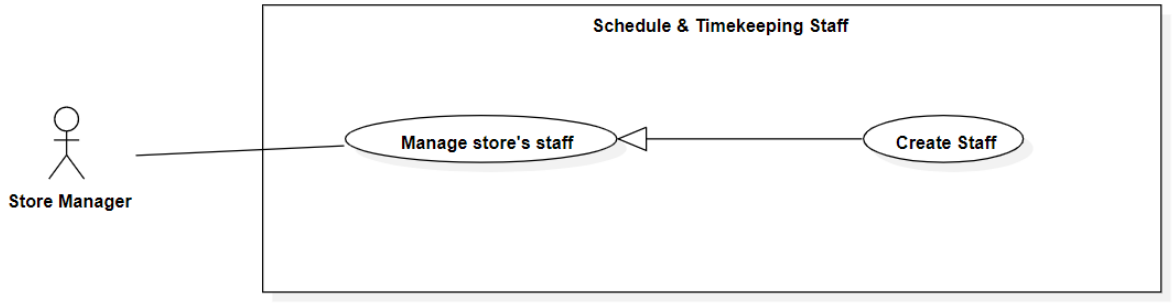


Figure 20 <Use Case> Create staff

<table>
<tr><td>UC ID and Name:</td><td colspan="3">MD4.1 Create staff</td></tr>
<tr><td>Created By:</td><td>CuongLV</td><td>Date Created:</td><td>21/06/2021</td></tr>
<tr><td>Primary Actor:</td><td>Store Manager</td><td>Secondary Actors:</td><td>N/A</td></tr>
<tr><td>Trigger:</td><td colspan="3">Store Manager click on “ADD STAFF” button</td></tr>
<tr><td>Description:</td><td colspan="3">Store Manager create a new account (staff role)</td></tr>
<tr><td>Preconditions:</td><td colspan="3">PRE-1: Users has logged in as store manager role</td></tr>
<tr><td>Post-conditions:</td><td colspan="3">POST-1: new staff account is added into the system.</td></tr>
<tr><td>Normal Flow:</td><td colspan="3">
<table>
<tr><th>Step</th><th>Actor Action</th><th>System Response</th></tr>
<tr><td>1</td><td>Store Manager click on “ADD STAFF” button</td><td>Open Add Staff Page</td></tr>
<tr><td>2</td><td>Fill all information inputs and
clicks “Save” button to send create staff request
[Exception 1] [Exception 2] [Exception 3]</td><td>System response that staff was created success</td></tr>
</table>
</td></tr>
<tr><td>Alternative Flows:</td><td colspan="3">N/A</td></tr>
<tr><td>Exceptions:</td><td colspan="3">
<table>
<tr><th>No</th><th>Cause</th><th>System Response</th></tr>
</table>
</td></tr>
</table>

	1	Fields are required.	System returns error message label that fields is required
	2	User name must be unique	System returns an error message that the user name is not unique.
	3	Invalid inputs	System returns error message label that fields are invalid
Priority:	High		
Frequency of Use:	Medium		
Business Rules:	N/A		
Other Information:	N/A		
Assumptions:	N/A		

Table 18 <Use Case> Create staff

2.5.2 View staff available time

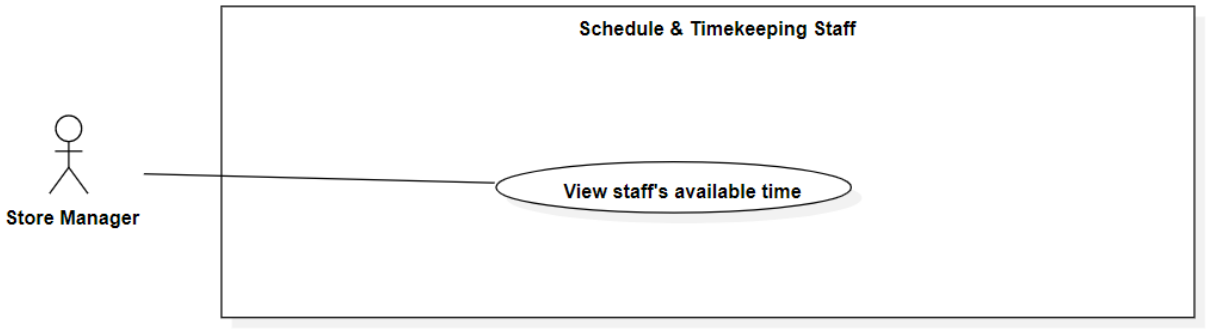


Figure 21 <Use Case> View staff available time

UC ID and Name:	**MD4.2 View staff available time**		
Created By:	CuongLV	Date Created:	21/06/2021
Primary Actor:	Store Manager	Secondary Actors:	N/A
Trigger:	Store Manager clicks on “Register” menu item in “Schedule” menu on sidebar		

<table>
<tr><td>Description:</td><td>Store Manager view all staff’s available time in current store</td></tr>
<tr><td>Preconditions:</td><td>PRE-1: Users has logged in as brand manager role</td></tr>
<tr><td>Post-conditions:</td><td>POST-1: Success: System returns list of staffs’s available time in table</td></tr>
<tr><td>Normal Flow:</td><td>
<table>
<tr><th>Step</th><th>Actor Action</th><th>System Response</th></tr>
<tr><td>1</td><td>Store Manager choose week sends request to get list of staffs’s available time</td><td>System response list of staffs’s available time</td></tr>
</table>
</td></tr>
<tr><td>Alternative Flows:</td><td>N/A</td></tr>
<tr><td>Exceptions:</td><td>N/A</td></tr>
<tr><td>Priority:</td><td>High</td></tr>
<tr><td>Frequency of Use:</td><td>High</td></tr>
<tr><td>Business Rules:</td><td>N/A</td></tr>
<tr><td>Other Information:</td><td>N/A</td></tr>
<tr><td>Assumptions:</td><td>N/A</td></tr>
</table>

Table 19 <Use Case> View staff available time

2.5.3 Update store’s information.

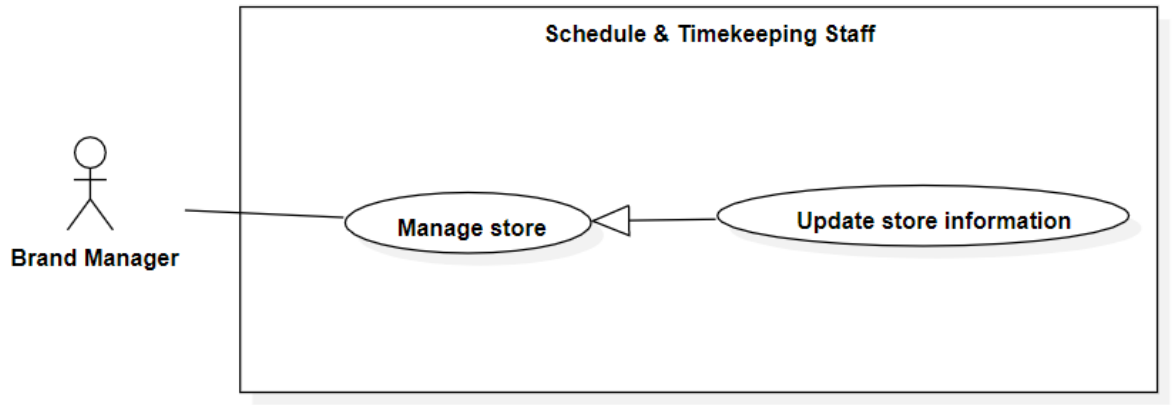


Figure 22 <Use Case> Update store information

<table>
<tr><td>UC ID and Name:</td><td colspan="3">MD4.3 Update store’s information</td></tr>
<tr><td>Created By:</td><td>CuongLV</td><td>Date Created:</td><td>21/06/2021</td></tr>
<tr><td>Primary Actor:</td><td>Store Manager</td><td>Secondary Actors:</td><td>N/A</td></tr>
<tr><td>Trigger:</td><td colspan="3">Store Manager requests to update store information.</td></tr>
<tr><td>Description:</td><td colspan="3">The Store Manager wants to update the store's information.</td></tr>
<tr><td>Preconditions:</td><td colspan="3">PRE-1: Users has logged in as store manager role</td></tr>
<tr><td>Post-conditions:</td><td colspan="3">POST-1: store’s information is updated.</td></tr>
<tr><td>Normal Flow:</td><td colspan="3"><table>
<tr><th>Step</th><th>Actor Action</th><th>System Response</th></tr>
<tr><td>1</td><td>Store manager fills all the information he/she wants to update.</td><td></td></tr>
<tr><td>2</td><td>Clicks on “Save” button</td><td>System validate input and response if that store’s information was updated success</td></tr>
</table></td></tr>
<tr><td>Alternative Flows:</td><td colspan="3">N/A</td></tr>
<tr><td>Exceptions:</td><td colspan="3"><table>
<tr><th>No</th><th>Cause</th><th>System Response</th></tr>
<tr><td>1</td><td>Fields are required.</td><td>System returns error message label that fields is required</td></tr>
</table></td></tr>
<tr><td>Priority:</td><td colspan="3">Medium</td></tr>
<tr><td>Frequency of Use:</td><td colspan="3">Low</td></tr>
<tr><td>Business Rules:</td><td colspan="3">Store Name can not null.</td></tr>
<tr><td>Other Information:</td><td colspan="3">N/A</td></tr>
<tr><td>Assumptions:</td><td colspan="3">N/A</td></tr>
</table>

Table 20 <Use Case> Update store’s information

2.5.4 View store's staff

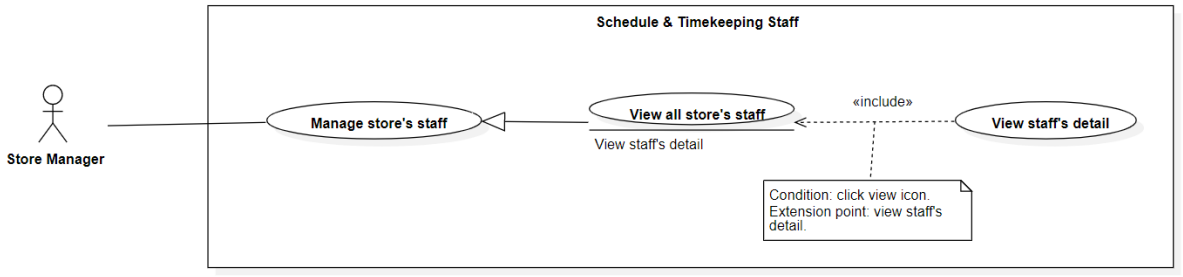


Figure 23 <Use Case> View store's staff

UC ID and Name:	**MD4.4 View store's staff**		
Created By:	CuongLV	Date Created:	21/06/2021
Primary Actor:	Store manager	Secondary Actors:	N/A
Trigger:	Store Manager click "Staff" menu item on sidebar		
Description:	Store Manager views all staff in the current store.		
Preconditions:	PRE-1: Users has logged in as store manager role		
Post-conditions:	POST-1: All staff of store will be displayed		
Normal Flow:			

Step	Actor Action	System Response
1	Store manager sends request to system to get list staffs	System response list of staffs

Alternative Flows:	N/A
Exceptions:	N/A
Priority:	Medium
Frequency of Use:	Medium
Business Rules:	N/A
Other Information:	N/A
Assumptions:	N/A

Table 21 <Use Case> View store's staff

2.5.5 View store information

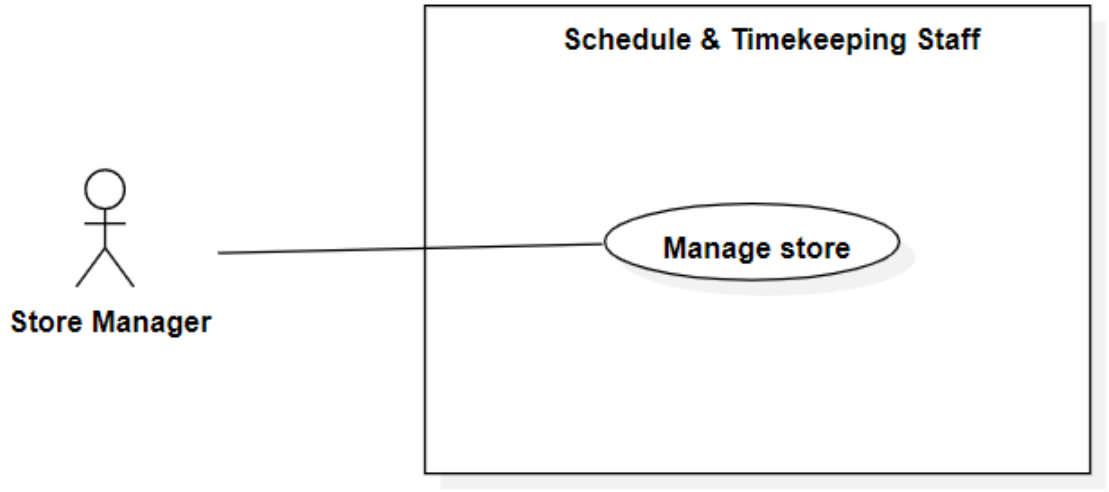


Figure 24 <Use Case> View store information

UC ID and Name:	**MD4.5 View store information**		
Created By:	MaiVTN	Date Created:	21/06/2021
Primary Actor:	Store manager	Secondary Actors:	N/A
Trigger:	The Store Manager click “Home” menu item on sidebar		
Description:	The Store Manager requested to view store information.		
Preconditions:	PRE-1: Users has logged in as store manager role		
Post-conditions:	POST-1: store information will be displayed		
Normal Flow:	Step / Actor Action / System Response 1 / Click “Home” / Store information		
Alternative Flows:	N/A		
Exceptions:	N/A		
Priority:	Medium		
Frequency of Use:	Medium		
Business Rules:	N/A		

Other Information:	N/A
Assumptions:	N/A

Table 22 <Use Case> View store information

2.5.6 Update staff

UC ID and Name:	**MD4.6 Update staff**		
Created By:	CuongLV	Date Created:	21/06/2021
Primary Actor:	Store Manager	Secondary Actors:	N/A
Trigger:	Store Manager requests to update staff.		
Description:	Store Manage update staff's information		
Preconditions:	PRE-1: Users has logged in as store manager role		
Post-conditions:	POST-1: Success: Staff's information is updated. Fail: show error message.		

Normal Flow:

Step	Actor Action	System Response
1	Store manager fills information inputs that he/she wants to update. to send update staff request	
2	Click the "Save" button. [Exception 1] [Exception 2]	System response if that staff was updated success

Alternative Flows:	N/A

Exceptions:

No	Cause	System Response
1	Fields are required.	System returns error message label that fields is required

	2	Invalid inputs	System returns error message label that fields are invalid	
Priority:	Medium			
Frequency of Use:	Low			
Business Rules:	N/A			
Other Information:	N/A			
Assumptions:	N/A			

Table 23 <Use Case> Update staff

2.5.7 Edit schedule constraints

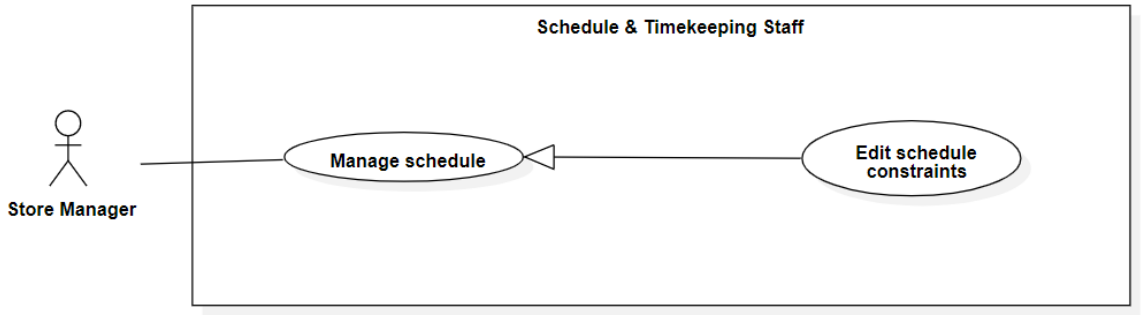


Figure 25 <Use Case> Edit schedule constraints

UC ID and Name:	**MD4.7 Edit schedule constraints**		
Created By:	MaiVTN	Date Created:	21/06/2021
Primary Actor:	Store Manager	Secondary Actors:	N/A
Trigger:	The actor requested to edit schedule constraints.		
Description:	Store Managers edit constraints of pastime or full time staff to run schedule algorithms.		
Preconditions:	PRE-1: Users has logged in as store manager role PRE-2: stay in “Schedule constraints”		
Post-conditions:	POST-1: Success: new constraints are installed. Fail: null		
Normal Flow:			

Step	Actor Action	System Response

	1	Setting all parameter in “Schedule constraints” include: “Min Day Off”, “Max Day Off”, “Min Working Time of the week”, “Max Working Time of the week”, “Max Working Time of the day”, “Min shift duration (30 minutes)”, “Max shift duration (30 minutes)”, “Max number of shift of the day”	Check to see actor’s input information is valid or not: time must be more than 0
	2	Press “Save Change”	Save config.
Alternative Flows:	N/A		
Exceptions:	N/A		
Priority:	High		
Frequency of Use:	Medium		
Business Rules:	BR7-9, BR7-13, BR7-15		
Other Information:	N/A		
Assumptions:	N/A		

Table 24 <Use Case> Edit schedule constraints

2.5.8 Edit schedule demands

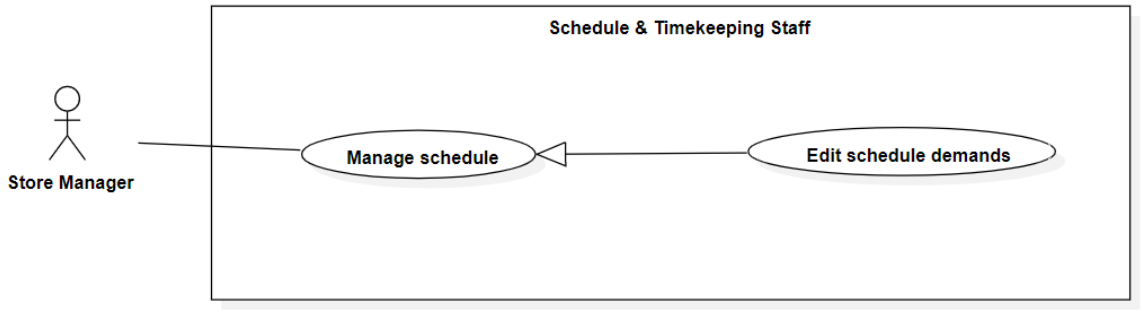


Figure 26 <Use Case> Edit schedule demands

UC ID and Name:	**MD4.8 Edit schedule demands**

<table>
<tr><td>Created By:</td><td>MaiVTN</td><td>Date Created:</td><td>21/06/2021</td></tr>
<tr><td>Primary Actor:</td><td>Store Manager</td><td>Secondary Actors:</td><td>N/A</td></tr>
<tr><td>Trigger:</td><td colspan="3">The actor requested to edit demands.</td></tr>
<tr><td>Description:</td><td colspan="3">Store manager establishes staffing needs according to the actual situation of the store.</td></tr>
<tr><td>Preconditions:</td><td colspan="3">PRE-1: Users have logged in as brand manager roles.
PRE-2: display in “Schedule demand”</td></tr>
<tr><td>Post-conditions:</td><td colspan="3">POST-1: new schedule demands.</td></tr>
<tr><td>Normal Flow:</td><td colspan="3"><table>
<tr><th>Step</th><th>Actor Action</th><th>System Response</th></tr>
<tr><td>1</td><td>Click on an empty cell on the calendar panel.</td><td>Show “New shift” popup</td></tr>
<tr><td>2</td><td>Setting all parameter in “Schedule constraints” include: “Level”, “Quantity”, “From”, “To” and click “Add”</td><td>Set a new demand and show it in the calendar panel.</td></tr>
</table></td></tr>
<tr><td>Alternative Flows:</td><td colspan="3">N/A</td></tr>
<tr><td>Exceptions:</td><td colspan="3">N/A</td></tr>
<tr><td>Priority:</td><td colspan="3">High</td></tr>
<tr><td>Frequency of Use:</td><td colspan="3">Medium</td></tr>
<tr><td>Business Rules:</td><td colspan="3">BR7-9, BR7-10, BR7-11, BR7-13, BR7-15</td></tr>
<tr><td>Other Information:</td><td colspan="3">N/A</td></tr>
<tr><td>Assumptions:</td><td colspan="3">N/A</td></tr>
</table>

Table 25 <Use Case> Edit schedule demands

2.5.9 View Schedule

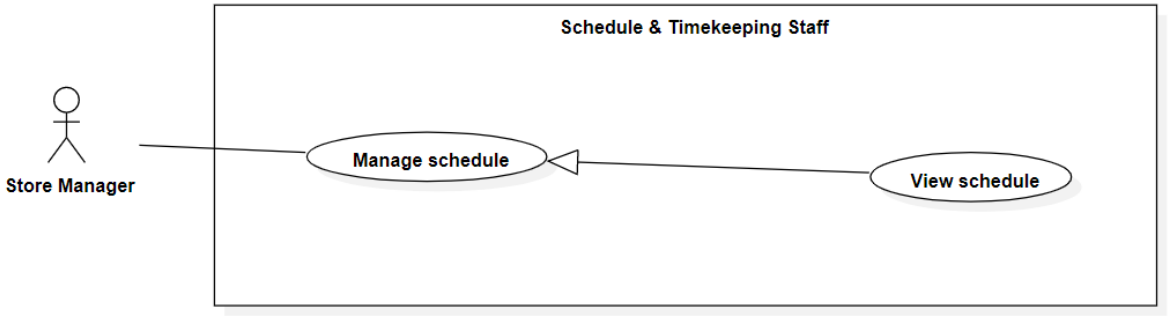


Figure 27 <Use Case> View schedule

<table>
<tr><td>UC ID and Name:</td><td colspan="3">MD4.9 View Schedule</td></tr>
<tr><td>Created By:</td><td>CuongLV</td><td>Date Created:</td><td>23/06/2021</td></tr>
<tr><td>Primary Actor:</td><td>Store Manager</td><td>Secondary Actors:</td><td>N/A</td></tr>
<tr><td>Trigger:</td><td colspan="3">The actor requested to view the schedule.</td></tr>
<tr><td>Description:</td><td colspan="3">Store Manager wants to view schedule</td></tr>
<tr><td>Preconditions:</td><td colspan="3">PRE-1: Users has logged in as store manager role
PRE-2: User stay in Schedule detail screen</td></tr>
<tr><td>Post-conditions:</td><td colspan="3">POST-1: List of Shift Assignments is displayed on schedule.</td></tr>
<tr><td>Normal Flow:</td><td colspan="3"><table><tr><th>Step</th><th>Actor Action</th><th>System Response</th></tr><tr><td>1</td><td>Store manager click on “Result” tab</td><td>System response list Shift Assignment plan</td></tr></table></td></tr>
<tr><td>Alternative Flows:</td><td colspan="3">N/A</td></tr>
<tr><td>Exceptions:</td><td colspan="3">N/A</td></tr>
<tr><td>Priority:</td><td colspan="3">High</td></tr>
<tr><td>Frequency of Use:</td><td colspan="3">High</td></tr>
<tr><td>Business Rules:</td><td colspan="3">N/A</td></tr>
<tr><td>Other Information:</td><td colspan="3">N/A</td></tr>
<tr><td>Assumptions:</td><td colspan="3">N/A</td></tr>
</table>

Table 26 <Use Case> View Schedule

2.5.10 Clone Schedule

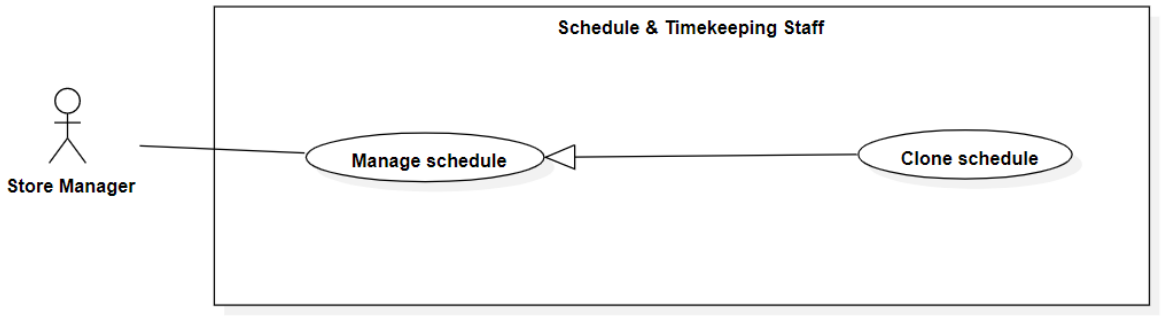


Figure 28 <Use Case> Clone Schedule

<table>
<tr><td>UC ID and Name:</td><td colspan="3">MD4.10 Clone Schedule</td></tr>
<tr><td>Created By:</td><td>MaiVTN</td><td>Date Created:</td><td>21/06/2021</td></tr>
<tr><td>Primary Actor:</td><td>Store Manager</td><td>Secondary Actors:</td><td>N/A</td></tr>
<tr><td>Trigger:</td><td colspan="3">Store Manager select schedule plan want to clone and click on clone icon button</td></tr>
<tr><td>Description:</td><td colspan="3">The Store Manager requested to clone a new schedule from the existing schedule plan.</td></tr>
<tr><td>Preconditions:</td><td colspan="3">PRE-1: Users has logged in as store manager role</td></tr>
<tr><td>Post-conditions:</td><td colspan="3">POST-1: New clone schedule plan is created and display on table</td></tr>
<tr><td>Normal Flow:</td><td colspan="3">
<table>
<tr><th>Step</th><th>Actor Action</th><th>System Response</th></tr>
<tr><td>1</td><td>Click the “clone” icon on the schedule plan to be cloned.</td><td>Pop up confirm message.</td></tr>
<tr><td>2</td><td>Click the “Confirm” button.</td><td>System response new schedule is cloned from this schedule.</td></tr>
</table>
</td></tr>
<tr><td>Alternative Flows:</td><td colspan="3">N/A</td></tr>
<tr><td>Exceptions:</td><td colspan="3">N/A</td></tr>
<tr><td>Priority:</td><td colspan="3">High</td></tr>
<tr><td>Frequency of Use:</td><td colspan="3">Medium</td></tr>
<tr><td>Business Rules:</td><td colspan="3">BR7-14, BR7-16</td></tr>
</table>

Other Information:	N/A
Assumptions:	N/A

Table 27 <Use Case> Clone Schedule

2.5.11 Publish schedule

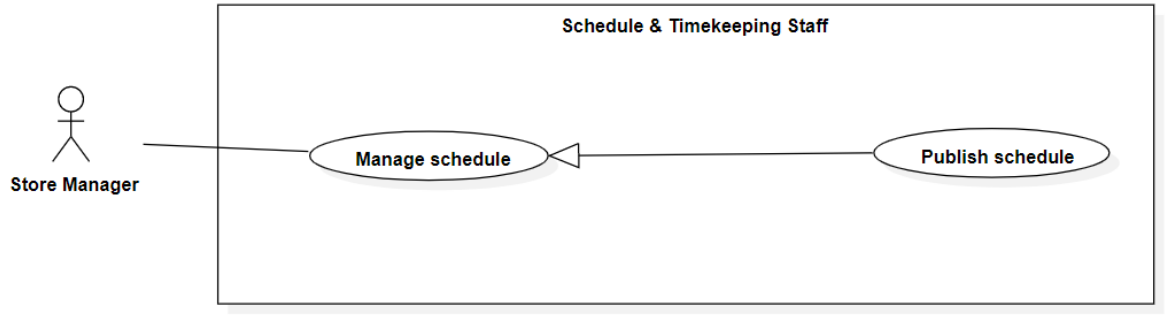


Figure 29 <Use Case> Publish schedule

UC ID and Name:	**MD4.11 Publish schedule.**		
Created By:	CuongLV	Date Created:	21/06/2021
Primary Actor:	Store Manager	Secondary Actors:	Staff
Trigger:	Store Manager clicks on “Publish” button		
Description:	The actor wants to publish the schedule to all staff in the store.		
Preconditions:	PRE-1: Users has logged in as store manager role PRE-2: Current screen is View Schedule		
Post-conditions:	POST-1: Success: Schedule is sent to all staff. Fail: null		
Normal Flow:			

Step	Actor Action	System Response
1	Store Manager clicks on “Publish” button	System takes the current schedule and saves it. After, system send notification to Staff

Alternative Flows:	N/A
Exceptions:	N/A
Priority:	High

Frequency of Use:	High
Business Rules:	BR7-05, BR7-17
Other Information:	N/A
Assumptions:	N/A

Table 28 <Use Case> Publish schedule

2.5.12 Delete Schedule Plan

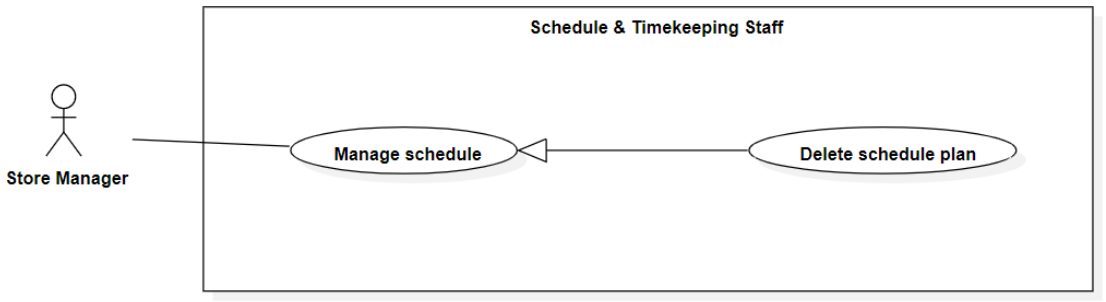


Figure 30 <Use Case> Delete schedule plan

<table>
<tr><td>UC ID and Name:</td><td colspan="3">MD4.12 Delete Schedule Plan</td></tr>
<tr><td>Created By:</td><td>MaiVTN</td><td>Date Created:</td><td>21/06/2021</td></tr>
<tr><td>Primary Actor:</td><td>Store Manager</td><td>Secondary Actors:</td><td>N/A</td></tr>
<tr><td>Trigger:</td><td colspan="3">The actor requests to delete a schedule.</td></tr>
<tr><td>Description:</td><td colspan="3">The Store manager wants to delete a schedule plan.</td></tr>
<tr><td>Preconditions:</td><td colspan="3">PRE-1: Users has logged in as store manager role</td></tr>
<tr><td>Post-conditions:</td><td colspan="3">POST-1: Delete schedule plan.</td></tr>
<tr><td>Normal Flow:</td><td colspan="3"><table><tr><th>Step</th><th>Actor Action</th><th>System Response</th></tr><tr><td>1</td><td>Click the “Delete” icon in the schedule plan page.</td><td>Pop up confirmation message.</td></tr><tr><td>2</td><td>Click the “Confirm” button.</td><td>Delete schedule plan</td></tr></table></td></tr>
<tr><td>Alternative Flows:</td><td colspan="3">N/A</td></tr>
<tr><td>Exceptions:</td><td colspan="3">N/A</td></tr>
<tr><td>Priority:</td><td colspan="3">High</td></tr>
</table>

Frequency of Use:	Low
Business Rules:	N/A
Other Information:	N/A
Assumptions:	N/A

Table 29 <Use Case> Delete Schedule

2.5.13 View staff's attendance

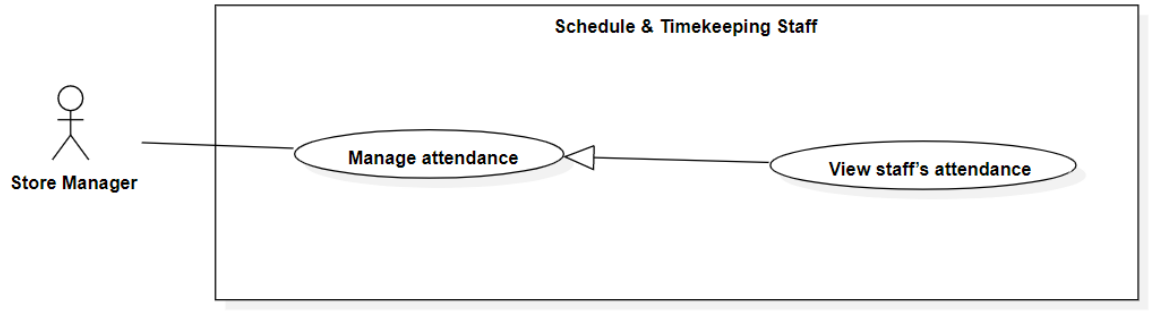


Figure 31 <Use Case> View staff's attendance

UC ID and Name:	**MD4.13 View staff's attendance**		
Created By:	MaiVTN	Date Created:	23/06/2021
Primary Actor:	Store Manager	Secondary Actors:	N/A
Trigger:	The actor requested to view the staff's attendance.		
Description:	Store managers review staff attendance times to make statistics and correct if there are errors.		
Preconditions:	PRE-1: Users have logged in as store manager roles.		
Post-conditions:	POST-1: Show staff attendance.		

Normal Flow:

Step	Actor Action	System Response
1	Click "Timekeeping" → "Attendance" in the dashboard.	Staff attendance and detailed information for each timekeeping.

Alternative Flows:	N/A
Exceptions:	N/A
Priority:	High
Frequency of Use:	High
Business Rules:	N/A
Other Information:	N/A
Assumptions:	N/A

Table 30 <Use Case> View staff's attendance

2.5.14 Add attendance manually

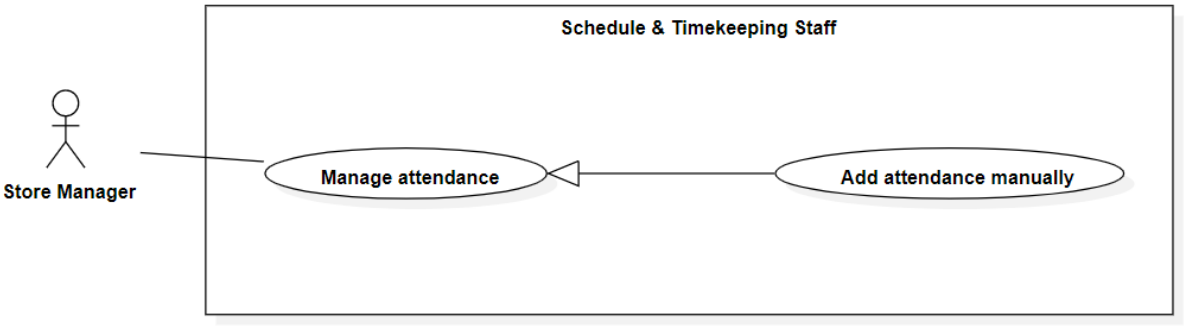


Figure 32 <Use Case> Add attendance manually

UC ID and Name:	**MD4.14 Add Attendance manually**		
Created By:	MaiVTN	Date Created:	24/06/2021
Primary Actor:	Store Manager	Secondary Actors:	N/A
Trigger:	The actor requested to add attendance manually.		
Description:	In case the attendance is wrong due to the system or the employee does not follow the attendance process correctly, the attendance will be conducted manually.		
Preconditions:	PRE-1: Users has logged in as store manager role		
Post-conditions:	POST-1: Record attendance of newly added staff manually.		
Normal Flow:			

Step	Actor Action	System Response
1	Click "Attendance" in the dashboard.	Show attendance page

	2	Click “Add Manually” in the top right of the page.	Popup “Add manually”
	3	Fill in all information include: “Staff Name”, “Date”, “Time check”, “Note” and click “Save”	Record new attendance.
Alternative Flows:	N/A		
Exceptions:	N/A		
Priority:	Medium		
Frequency of Use:	Medium		
Business Rules:	N/A		
Other Information:	N/A		
Assumptions:	N/A		

Table 31 <Use Case> Add Attendance manually

2.5.15 View staff request

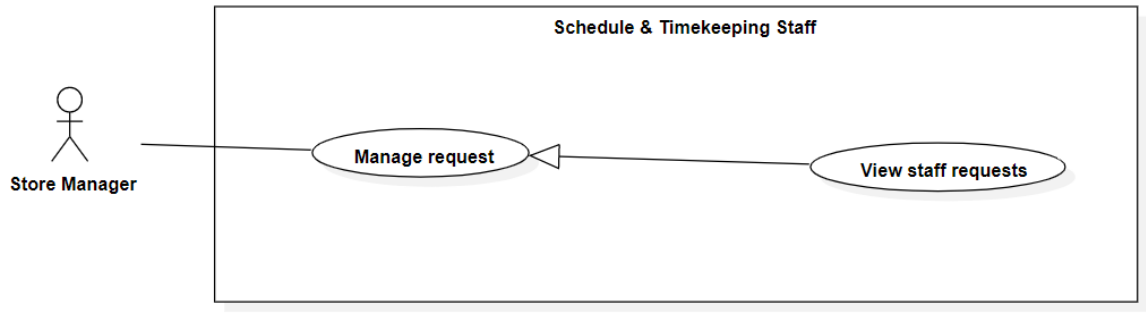


Figure 33 <Use Case> View staff request

UC ID and Name:	**MD4.15 View staff request**		
Created By:	MaiVTN	Date Created:	24/06/2021
Primary Actor:	Store Manager	Secondary Actors:	N/A
Trigger:	The actor requested to view notification of staff request.		

<table>
<tr><td>Description:</td><td>Store managers see notifications of staff requests when employees want to take time off.</td></tr>
<tr><td>Preconditions:</td><td>PRE-1: Users has logged in as store manager role</td></tr>
<tr><td>Post-conditions:</td><td>POST-1: Show staff request.</td></tr>
<tr><td>Normal Flow:</td><td><table><tr><th>Step</th><th>Actor Action</th><th>System Response</th></tr><tr><td>1</td><td>Click “Notification” on the dashboard. Then click on “Request”.</td><td>Show staff request</td></tr></table></td></tr>
<tr><td>Alternative Flows:</td><td>N/A</td></tr>
<tr><td>Exceptions:</td><td>N/A</td></tr>
<tr><td>Priority:</td><td>High</td></tr>
<tr><td>Frequency of Use:</td><td>Sometime</td></tr>
<tr><td>Business Rules:</td><td>N/A</td></tr>
<tr><td>Other Information:</td><td>N/A</td></tr>
<tr><td>Assumptions:</td><td>N/A</td></tr>
</table>

Table 32 <Use Case> View staff request

2.5.16 Approve staff absence request

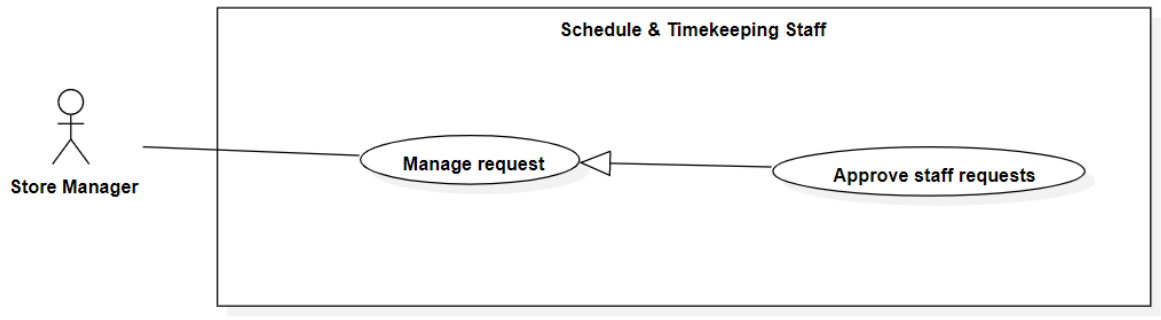


Figure 34 <Use Case> Approve staff requests

UC ID and Name:	**MD4.16 Approve staff absence request**		
Created By:	MaiVTN	Date Created:	25/06/2021
Primary Actor:	Store Manager	Secondary Actors:	N/A
Trigger:	The actor requested to approve a staff absence request.		

<table>
<tr><td>Description:</td><td>After the store manager views a staff request for leave, they will have the right to accept or not accept the request.</td></tr>
<tr><td>Preconditions:</td><td>PRE-1: Users have logged in as store manager roles.
PRE-2: Use Case MD4.15 is successed.
PRE-3: Request type is absence request.</td></tr>
<tr><td>Post-conditions:</td><td>POST-1: success: Staff absence request is approved.
fail: staff absence request isn’t approved.</td></tr>
<tr><td>Normal Flow:</td><td><table>
<tr><th>Step</th><th>Actor Action</th><th>System Response</th></tr>
<tr><td>1</td><td>Click the “Approve” button.</td><td>Show detail of request</td></tr>
<tr><td>2</td><td>Click “Approve” to approve and “Reject” to reject the request.</td><td>Send notification that request is successful or not.</td></tr>
</table></td></tr>
<tr><td>Alternative Flows:</td><td>N/A</td></tr>
<tr><td>Exceptions:</td><td>N/A</td></tr>
<tr><td>Priority:</td><td>Low</td></tr>
<tr><td>Frequency of Use:</td><td>Low</td></tr>
<tr><td>Business Rules:</td><td>N/A</td></tr>
<tr><td>Other Information:</td><td>N/A</td></tr>
<tr><td>Assumptions:</td><td>N/A</td></tr>
</table>

Table 33 <Use Case> Approve staff absence request

2.5.17 Approve staff change shift request

UC ID and Name:	**MD4.17 Approve staff change shift request**		
Created By:	CuongLV	Date Created:	26/06/2021
Primary Actor:	Store Manager	Secondary Actors:	N/A
Trigger:	The staff sent a change shift request.		
Description:	Staff want to change their shift and store managers need to approve or reject this request.		
Preconditions:	PRE-1: Users has logged in as store manager role PRE-2: Use Case MD4.15 is successed. PRE-3: Request type is change shift request.		

<table>
<tr><td>Post-conditions:</td><td>POST-1: success: Staff change shift request is approved.
fail: staff change shift request isn’t approved.</td></tr>
<tr><td>Normal Flow:</td><td>
<table>
<tr><th>Step</th><th>Actor Action</th><th>System Response</th></tr>
<tr><td>1</td><td>Click the “Approve” button.</td><td>Show detail of request</td></tr>
<tr><td>2</td><td>Click “Approve” to approve and “Reject” to reject the request.</td><td>Send notification that request is successful or not.</td></tr>
</table>
</td></tr>
<tr><td>Alternative Flows:</td><td>N/A</td></tr>
<tr><td>Exceptions:</td><td>N/A</td></tr>
<tr><td>Priority:</td><td>Low</td></tr>
<tr><td>Frequency of Use:</td><td>Low</td></tr>
<tr><td>Business Rules:</td><td>N/A</td></tr>
<tr><td>Other Information:</td><td>N/A</td></tr>
<tr><td>Assumptions:</td><td>N/A</td></tr>
</table>

Table 34 <Use Case> Approve staff change shift request

2.5.18 Report staff performance

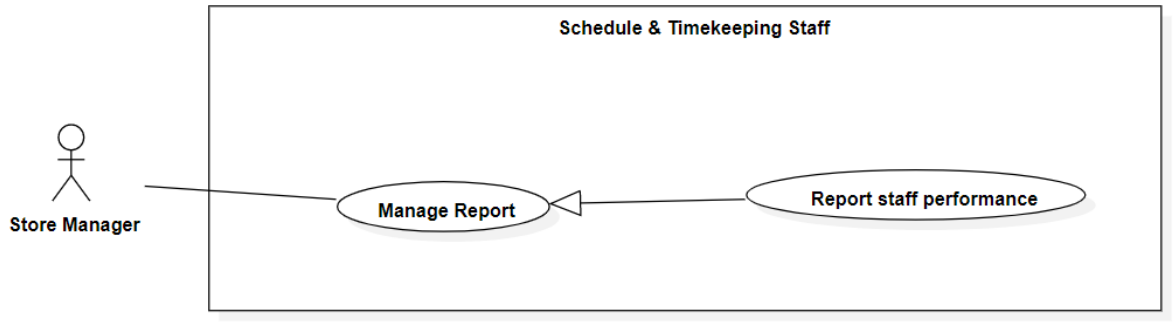


Figure 35 <Use Case> Report staff performance

UC ID and Name:	**MD4.18 Report staff performance**		
Created By:	DaoPM	Date Created:	21/06/2021
Primary Actor:	Store Manager	Secondary Actors:	N/A
Trigger:	Store Manager requests to view report of staff		
Description:	Store Manager wants to view report of staff		

<table>
<tr><td>Preconditions:</td><td>PRE-1: User has logged in as role store manager.</td></tr>
<tr><td>Post-conditions:</td><td>POST-1: display report staff performance on table</td></tr>
<tr><td>Normal Flow:</td><td>
<table>
<tr><th>Step</th><th>Actor Action</th><th>System Response</th></tr>
<tr><td>1</td><td>Store Manager clicks on the “Staff” item on the “Report” menu item.</td><td>System return default list of record of first staff from begin of current week to current day</td></tr>
<tr><td>2</td><td>Store Manager choose staff and time range to view the report</td><td>System return list of record by date and display it on table</td></tr>
</table>
</td></tr>
<tr><td>Alternative Flows:</td><td>N/A</td></tr>
<tr><td>Exceptions:</td><td>N/A</td></tr>
<tr><td>Priority:</td><td>High</td></tr>
<tr><td>Frequency of Use:</td><td>Medium</td></tr>
<tr><td>Business Rules:</td><td>BR9-16</td></tr>
<tr><td>Other Information:</td><td>N/A</td></tr>
<tr><td>Assumptions:</td><td>N/A</td></tr>
</table>

Table 35 <Use Case> Report staff performance

2.5.19 Report store performance

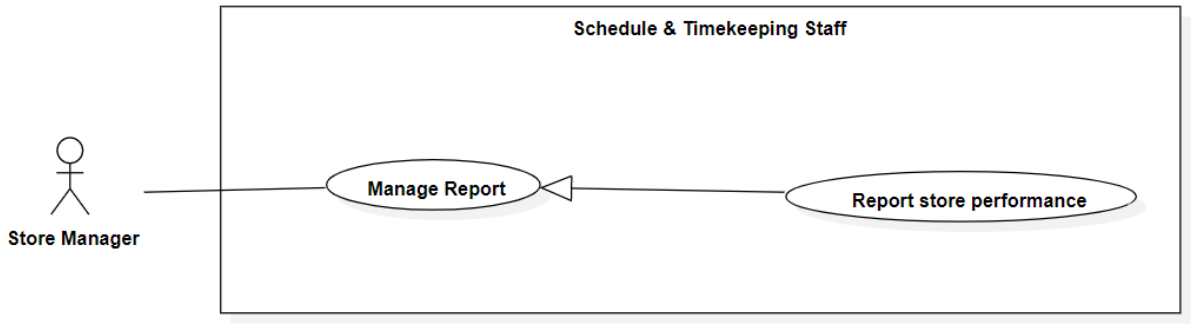


Figure 36 <Use Case> Report store performance

<table>
<tr><td>UC ID and Name:</td><td colspan="3">MD4.19 Report store performance</td></tr>
<tr><td>Created By:</td><td>DaoPM</td><td>Date Created:</td><td>21/06/2021</td></tr>
<tr><td>Primary Actor:</td><td>Store Manager</td><td>Secondary Actors:</td><td>N/A</td></tr>
<tr><td>Trigger:</td><td colspan="3">Store Manager requests to view report of the store.</td></tr>
<tr><td>Description:</td><td colspan="3">Store Manager wants to view report of the store</td></tr>
<tr><td>Preconditions:</td><td colspan="3">PRE-1: User has logged in as store manager role</td></tr>
<tr><td>Post-conditions:</td><td colspan="3">POST-1: display report store performance on table</td></tr>
<tr><td>Normal Flow:</td><td colspan="3"><table>
<tr><th>Step</th><th>Actor Action</th><th>System Response</th></tr>
<tr><td>1</td><td>The Store Manager clicks on “Store” item on “Report” menu item.</td><td>Redirect to store report page.</td></tr>
<tr><td>2</td><td>Store Manager choose a time range</td><td>System response list of record by staff and display it on table</td></tr>
</table></td></tr>
<tr><td>Alternative Flows:</td><td colspan="3">N/A</td></tr>
<tr><td>Exceptions:</td><td colspan="3">N/A</td></tr>
<tr><td>Priority:</td><td colspan="3">High</td></tr>
<tr><td>Frequency of Use:</td><td colspan="3">Medium</td></tr>
<tr><td>Business Rules:</td><td colspan="3">BR9-17</td></tr>
<tr><td>Other Information:</td><td colspan="3">N/A</td></tr>
<tr><td>Assumptions:</td><td colspan="3">N/A</td></tr>
</table>

Table 36 <Use Case> Report store performance

2.5.20 Config store operating hours

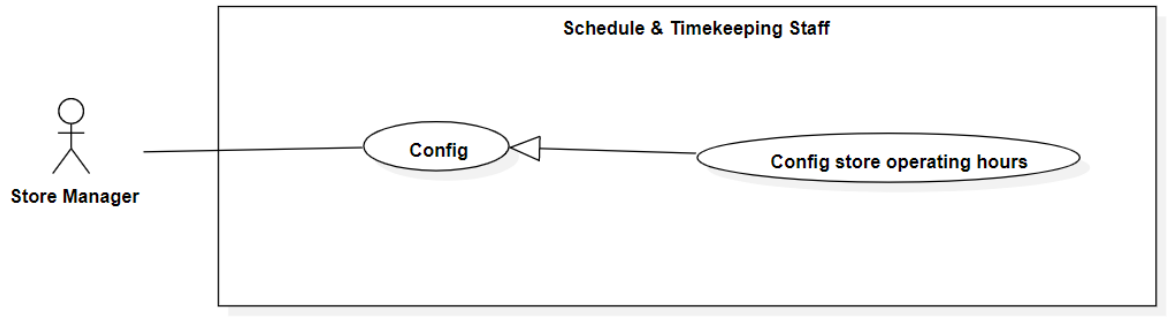


Figure 37 <Use Case> Config store operating hours

UC ID and Name:	**MD4.20 Config store operating hours**		
Created By:	CuongLV	Date Created:	21/06/2021
Primary Actor:	Store Manager	Secondary Actors:	N/A
Trigger:	Store Manager click on “Schedule Config” menu item in “Config” menu in sidebar		
Description:	The Store Manager wants to configure the store operating hours.		
Preconditions:	PRE-1: Users has logged in as store manager role		
Post-conditions:	POST-1: Success: store operating hours are saved by the system. Fail: null		
Normal Flow:			

Step	Actor Action	System Response
1	The Store Manager clicks on the “operating time” tab on the tab bar.	System response current the store’s operating hours
2	Store Manager fill information need to change on tabpanel	
3	Store Manager clicks on “Save” button	System save new the store’s operating hours

Alternative Flows:	N/A
Exceptions:	N/A

Priority:	Medium
Frequency of Use:	Low
Business Rules:	BR7-11, BR7-15
Other Information:	N/A
Assumptions:	N/A

Table 37 <Use Case> Config store operating hours

2.5.21 Config store constraints

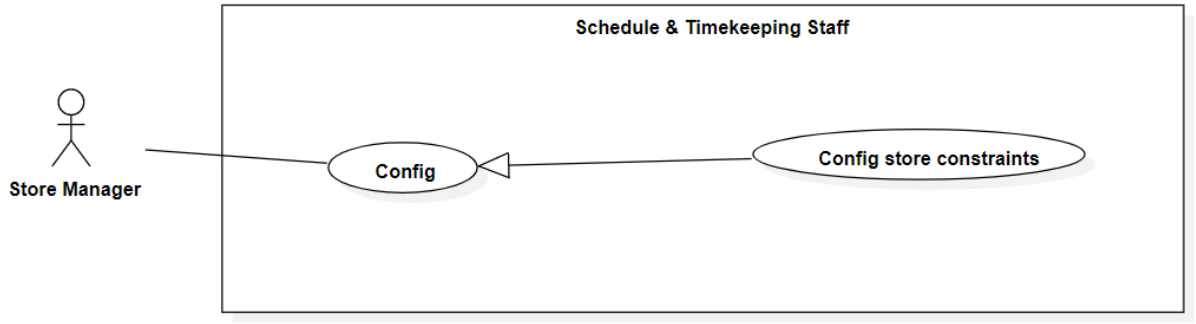


Figure 38 <Use Case> Config store constraints

UC ID and Name:	**MD4.22 Config store constraints**		
Created By:	DaoPM	Date Created:	21/06/2021
Primary Actor:	Store Manager	Secondary Actors:	N/A
Trigger:	Store Manager requests to configure store constraints.		
Description:	The Store Manager wants to configure the store constraints default values.		
Preconditions:	PRE-1: Users has logged in as store manager role		
Post-conditions:	POST-1: Success: store's constraints default values are saved by the system.		

Normal Flow:

Step	Actor Action	System Response
1	The Store Manager clicks on the "constraints" tab on the tab bar.	System response default of the store's constraints
2	Store Manager fill information need to change on tabpanel	

	3	Store Manager clicks on “Save” button	System save new the store’s constraint values
Alternative Flows:	N/A		
Exceptions:	N/A		
Priority:	Medium		
Frequency of Use:	Medium		
Business Rules:	BR7-08, BR7-09, BR7-11, BR7-13, BR7-15		
Other Information:	N/A		
Assumptions:	N/A		

Table 38 <Use Case> Config store constraints

2.5.22 Config store demands

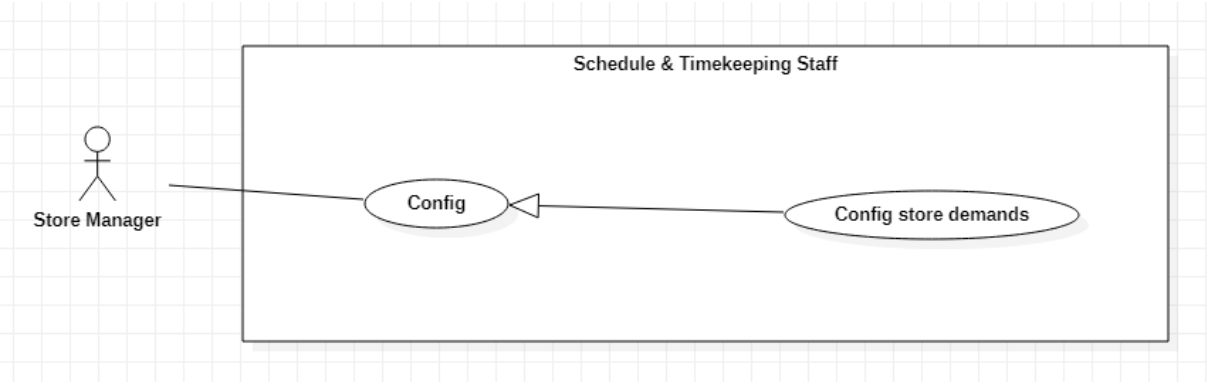


Figure 39 <Use Case> Config store demands

UC ID and Name:	**MD4.22 Config store demands**		
Created By:	DaoPM	Date Created:	21/06/2021
Primary Actor:	Store Manager	Secondary Actors:	N/A
Trigger:	Store Manager requests to configure the demand template.		
Description:	The Store Manager wants to configure the store demand template.		
Preconditions:	PRE-1: Users has logged in as store manager role		

<table>
<tr><td>Post-conditions:</td><td>POST-1: Success: store schedule demand hours are saved by the system.</td></tr>
<tr><td>Normal Flow:</td><td>
<table>
<tr><th>Step</th><th>Actor Action</th><th>System Response</th></tr>
<tr><td>1</td><td>The Store Manager clicks on the “demands” tab on the tab bar.</td><td></td></tr>
<tr><td>2</td><td>The Store Manager choose template to config</td><td>System response template of the store’s demand</td></tr>
<tr><td>3</td><td>Fill in the demands for template</td><td></td></tr>
</table>
</td></tr>
<tr><td>Alternative Flows:</td><td>N/A</td></tr>
<tr><td>Exceptions:</td><td>N/A</td></tr>
<tr><td>Priority:</td><td>Medium</td></tr>
<tr><td>Frequency of Use:</td><td>Medium</td></tr>
<tr><td>Business Rules:</td><td>BR7-08, BR7-09, BR7-11, BR7-13, BR7-15</td></tr>
<tr><td>Other Information:</td><td>N/A</td></tr>
<tr><td>Assumptions:</td><td>N/A</td></tr>
</table>

Table 39 <Use Case> Config store demands

2.5.23 Compute scheduling

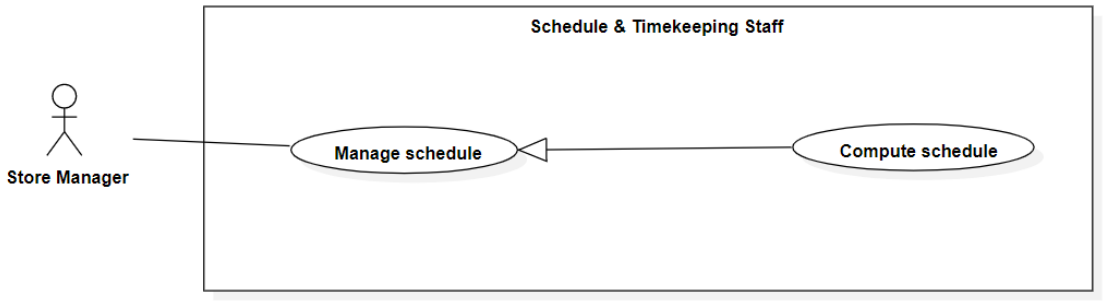


Figure 40 <Use Case> Compute Scheduling

<table>
<tr><td>UC ID and Name:</td><td colspan="3">MD4.23 Compute schedule</td></tr>
<tr><td>Created By:</td><td>CuongLV</td><td>Date Created:</td><td>21/06/2021</td></tr>
<tr><td>Primary Actor:</td><td>Store manager</td><td>Secondary Actors:</td><td>N/A</td></tr>
<tr><td>Trigger:</td><td colspan="3">The Store manager wants to compute the schedule.</td></tr>
<tr><td>Description:</td><td colspan="3">After preparing data for the computing schedule. Store Manage sends a computing schedule request.</td></tr>
<tr><td>Preconditions:</td><td colspan="3">PRE-1: Users has logged in as store manager role
PRE-2: Data for input is ready.
PRE-3: User at Schedule result page.</td></tr>
<tr><td>Post-conditions:</td><td colspan="3">POST-1: Success: Algorithm return result.
Fail: The system shows the error message and logs it</td></tr>
<tr><td>Normal Flow:</td><td colspan="3">
<table>
<tr><th>Step</th><th>Actor Action</th><th>System Response</th></tr>
<tr><td>1</td><td>Store manager clicks the “Compute Schedule” button to trigger computing.</td><td>System response “id” and send request computing with data to Algorithm</td></tr>
<tr><td>2</td><td>Algorithm compute completely and send request to save result</td><td>System response result to Store manager</td></tr>
</table>
</td></tr>
</table>

<table>
<tr><td>Alternative Flows:</td><td>
<table>
<tr><th>Step</th><th>Actor Action</th><th>System Response</th></tr>
<tr><td>1</td><td>Store manager clicks the “Compute Schedule” button to trigger computing.</td><td>System response “id” and send request computing with data to Algorithm</td></tr>
<tr><td>2</td><td>Algorithm compute unsuccessful send request with empty result</td><td>System show alert can’t compute to Store Manage</td></tr>
</table>
</td></tr>
<tr><td>Exceptions:</td><td>N/A</td></tr>
<tr><td>Priority:</td><td>High</td></tr>
<tr><td>Frequency of Use:</td><td>High</td></tr>
<tr><td>Business Rules:</td><td>BR7-02, BR7-06, BR7-07, BR7-08, BR7-09</td></tr>
<tr><td>Other Information:</td><td>N/A</td></tr>
<tr><td>Assumptions:</td><td>N/A</td></tr>
</table>

Table 40 <Use Case> Compute schedule

2.6 MD5. Staff Module

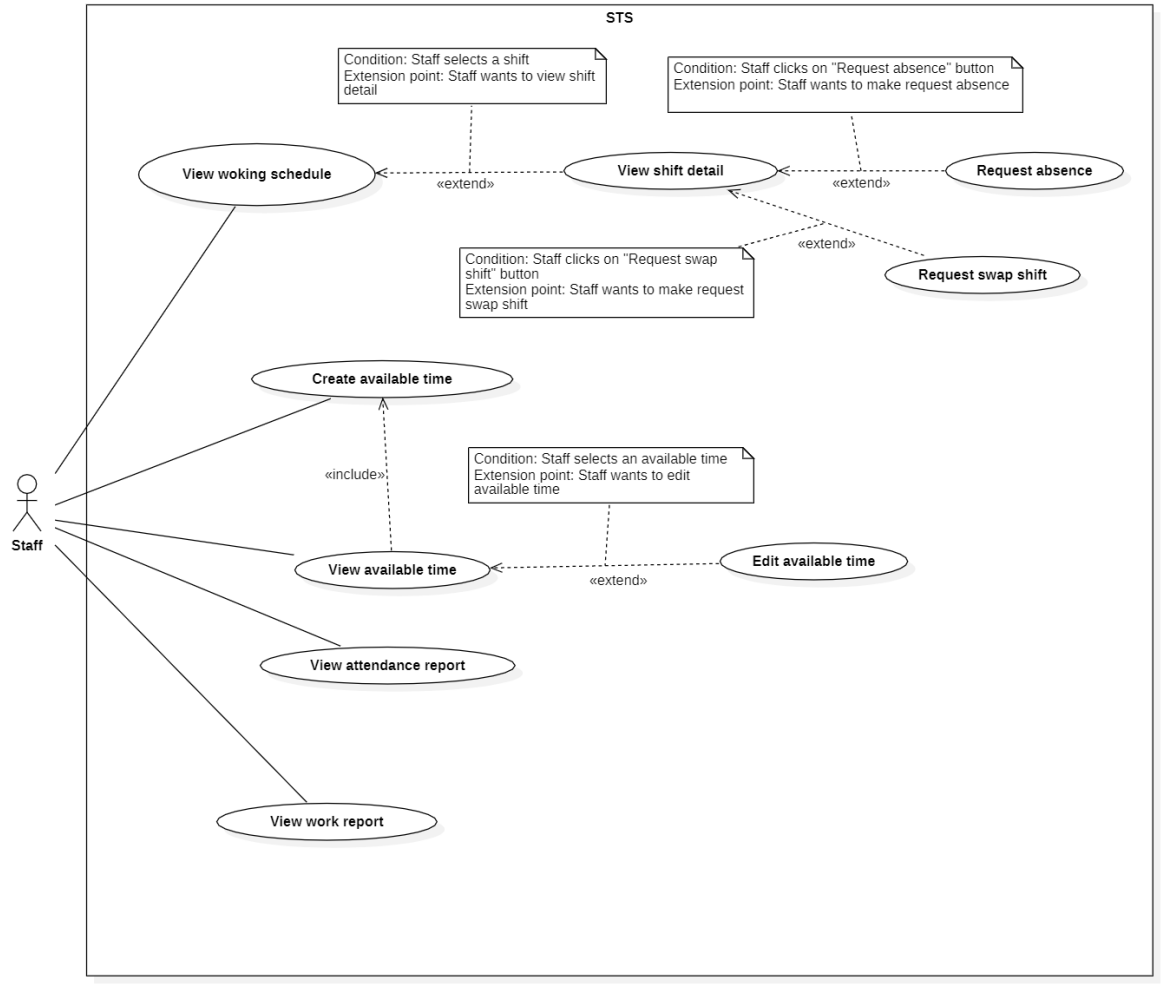


Figure 41 <Use Case Diagram> Staff Use Case Diagram

2.6.1 View working schedule

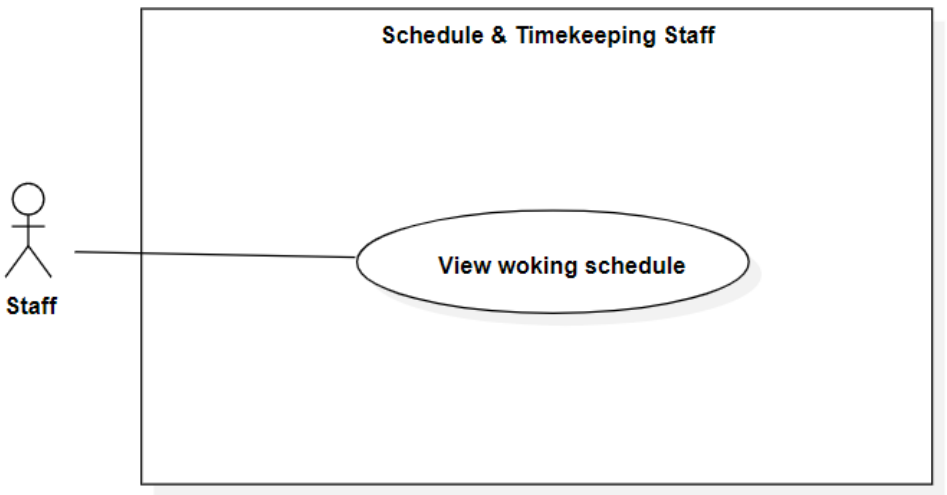


Figure 42 <Use Case> View working schedule

<table>
<tr><td>UC ID and Name:</td><td colspan="3">MD5.01 View working schedule</td></tr>
<tr><td>Created By:</td><td>TrungDQ</td><td>Date Created:</td><td>12/06/2021</td></tr>
<tr><td>Primary Actor:</td><td>Staff</td><td>Secondary Actors:</td><td>N/A</td></tr>
<tr><td>Trigger:</td><td colspan="3">Staff request to get a working schedule.</td></tr>
<tr><td>Description:</td><td colspan="3">Staff views working schedule.</td></tr>
<tr><td>Preconditions:</td><td colspan="3">PRE-1: Staff has logged into the application.
PRE-2: Current screen is “Home” screen.</td></tr>
<tr><td>Post-conditions:</td><td colspan="3">POST-1: Working schedule will be displayed.</td></tr>
<tr><td>Normal Flow:</td><td colspan="3"><table><tr><th>Step</th><th>Actor Action</th><th>System Response</th></tr><tr><td>1</td><td>Staff clicks on the “Schedule” button in the bottom navigation bar.</td><td>System navigates to the “Schedule” screen, displaying a working schedule.</td></tr></table></td></tr>
<tr><td>Alternative Flows:</td><td colspan="3">N/A</td></tr>
<tr><td>Exceptions:</td><td colspan="3">N/A</td></tr>
<tr><td>Priority:</td><td colspan="3">High</td></tr>
<tr><td>Frequency of Use:</td><td colspan="3">High</td></tr>
<tr><td>Business Rules:</td><td colspan="3">N/A</td></tr>
<tr><td>Other Information:</td><td colspan="3">Working schedule of the current week is displayed by default.</td></tr>
<tr><td>Assumptions:</td><td colspan="3">N/A</td></tr>
</table>

Table 41 <Use Case> View working schedule

2.6.2 View shift detail

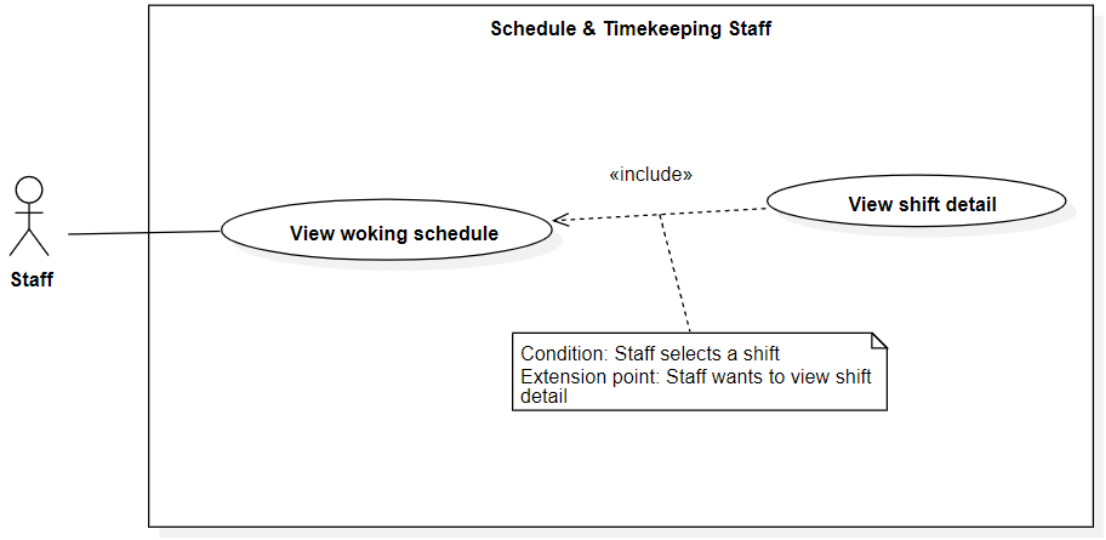


Figure 43 <Use Case> View shift detail

UC ID and Name:	**MD5.02 View shift detail**		
Created By:	TrungDQ	Date Created:	12/06/2021
Primary Actor:	Staff	Secondary Actors:	N/A
Trigger:	Staff request to get details of a shift.		
Description:	Staff views detailed information of selected shift.		
Preconditions:	PRE-1: Staff has logged into the application. PRE-2: Current screen is “Schedule” screen.		
Post-conditions:	POST-1: Detailed information of shift will be displayed.		
Normal Flow:			

Step	Actor Action	System Response
1	Staff selects a shift.	System navigates to the “Shift Detail” screen, displaying detailed information of selected shift including location, skill, time,...

Alternative Flows:	N/A

Exceptions:	N/A
Priority:	Medium
Frequency of Use:	Medium
Business Rules:	N/A
Other Information:	N/A
Assumptions:	N/A

Table 42 <Use Case> View shift detail

2.6.3 Request absence

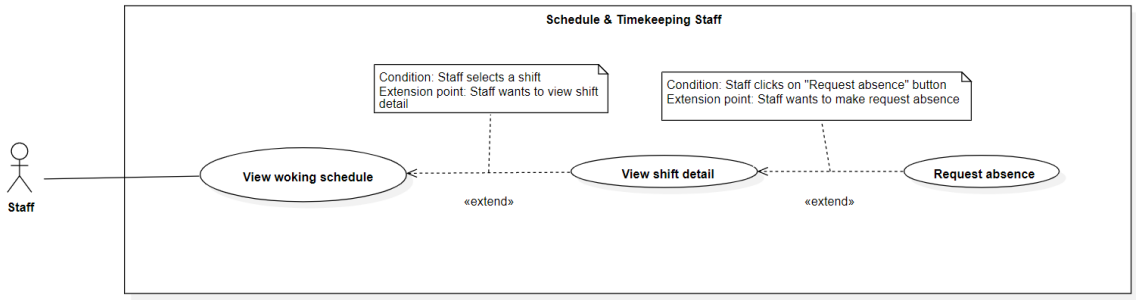


Figure 44 <Use Case> Request absence

UC ID and Name:	**MD4.3 Request absence**		
Created By:	TrungDQ	Date Created:	12/06/2021
Primary Actor:	Staff	Secondary Actors:	N/A
Trigger:	Staff requests to make request absence.		
Description:	Staff makes request absence for selected shift.		
Preconditions:	PRE-1: Staff has logged into the application. PRE-2: Current screen is “Shift Detail” screen. PRE-3: Selected shift has not started.		
Post-conditions:	POST-1: The request is saved to the system.		
Normal Flow:			

Step	Actor Action	System Response
1	Staff clicks on the “Request absence” button.	System displays confirm dialog.

<table>
<tr><td></td><td>
<table>
<tr><td>2</td><td>Staff clicks the confirm button.</td><td>Request is saved to the system.</td></tr>
<tr><td></td><td></td><td>System shows successful message.</td></tr>
</table>
</td></tr>
<tr><td>Alternative Flows:</td><td>N/A</td></tr>
<tr><td>Exceptions:</td><td>
<table>
<tr><th>No</th><th>Cause</th><th>System Response</th></tr>
<tr><td>1</td><td>“Reason” text field is empty.</td><td>System disables “Confirm” button.</td></tr>
<tr><td></td><td></td><td></td></tr>
</table>
</td></tr>
<tr><td>Priority:</td><td>Low</td></tr>
<tr><td>Frequency of Use:</td><td>Low</td></tr>
<tr><td>Business Rules:</td><td>N/A</td></tr>
<tr><td>Other Information:</td><td>N/A</td></tr>
<tr><td>Assumptions:</td><td>N/A</td></tr>
</table>

Table 43 <Use Case> Request absence

2.6.4 Request swap shift

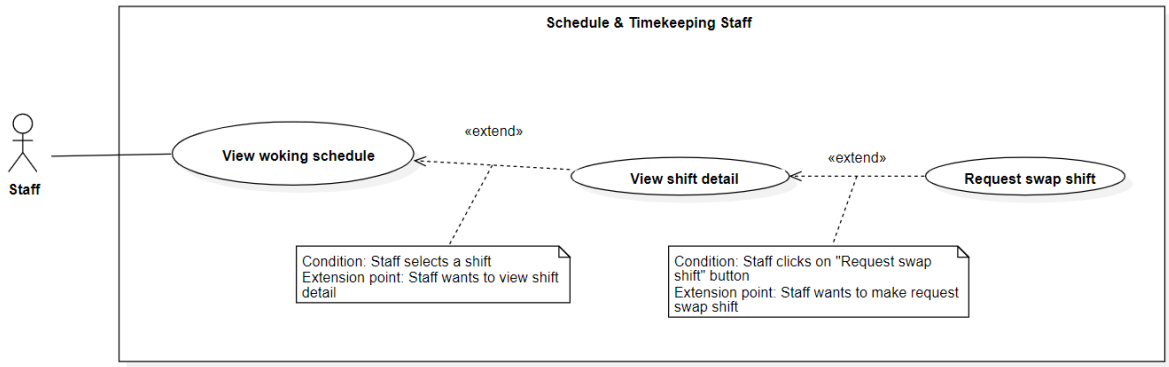


Figure 45 <Use Case> Request swap shift

<table>
<tr><td>UC ID and Name:</td><td colspan="3">MD5.04 Request swap shift</td></tr>
<tr><td>Created By:</td><td>TrungDQ</td><td>Date Created:</td><td>12/06/2021</td></tr>
<tr><td>Primary Actor:</td><td>Staff</td><td>Secondary Actors:</td><td>N/A</td></tr>
<tr><td>Trigger:</td><td colspan="3">Staff requests to make request swap shift.</td></tr>
<tr><td>Description:</td><td colspan="3">Staff makes request swap shift for selected shift.</td></tr>
<tr><td>Preconditions:</td><td colspan="3">PRE-1: Staff has logged into the application.
PRE-2: Current screen is “Shift Detail” screen.
PRE-3: Selected shift has not started.</td></tr>
<tr><td>Post-conditions:</td><td colspan="3">POST-1: The request is saved to the system.</td></tr>
<tr><td>Normal Flow:</td><td colspan="3"><table>
<tr><th>Step</th><th>Actor Action</th><th>System Response</th></tr>
<tr><td>1</td><td>Staff clicks on the “Request swap shift” button.</td><td>System displays confirm dialog.</td></tr>
<tr><td>2</td><td>Staff clicks the confirm button.</td><td>Request is saved to the system.</td></tr>
<tr><td></td><td></td><td>System shows successful message.</td></tr>
</table></td></tr>
<tr><td>Alternative Flows:</td><td colspan="3">N/A</td></tr>
<tr><td>Exceptions:</td><td colspan="3">N/A</td></tr>
<tr><td>Priority:</td><td colspan="3">Low</td></tr>
<tr><td>Frequency of Use:</td><td colspan="3">Low</td></tr>
<tr><td>Business Rules:</td><td colspan="3">N/A</td></tr>
<tr><td>Other Information:</td><td colspan="3">N/A</td></tr>
<tr><td>Assumptions:</td><td colspan="3">N/A</td></tr>
</table>

Table 44 <Use Case> Request swap shift

2.6.5 View available time

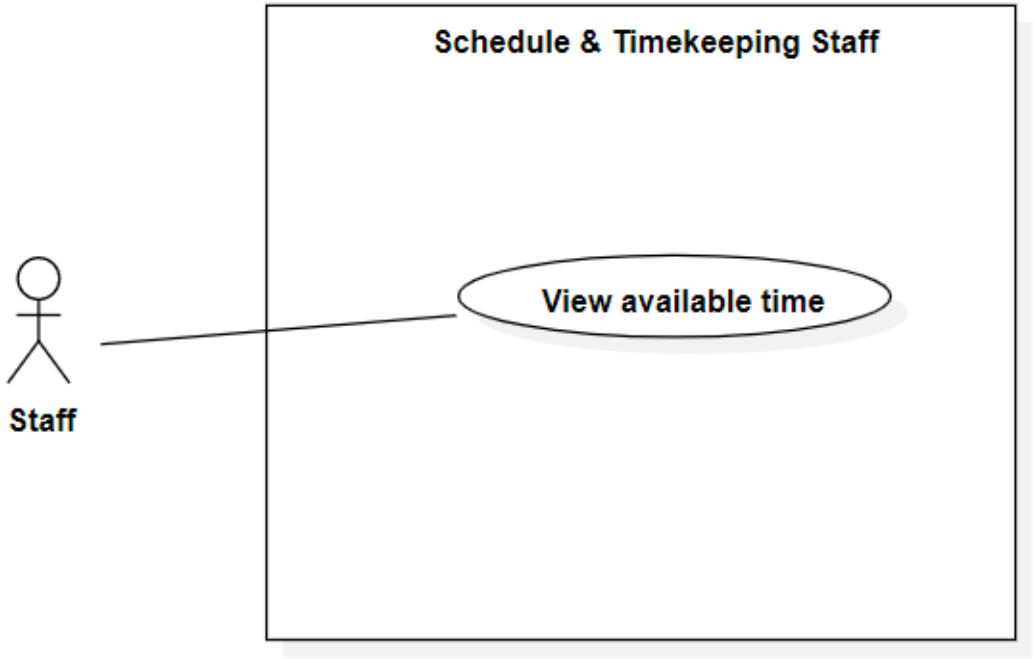


Figure 46 <Use Case> View available time

UC ID and Name:	**MD5.05 View available time**		
Created By:	TrungDQ	Date Created:	21/06/2021
Primary Actor:	Staff	Secondary Actors:	N/A
Trigger:	Staff request to get available time.		
Description:	Staff views available time that has been created before.		
Preconditions:	PRE-1: Staff has logged into the application. PRE-2: Current screen is “Home” screen.		
Post-conditions:	POST-1: List of available time will be displayed.		
Normal Flow:			

Step	Actor Action	System Response
1	Staff clicks on the “Register” button in the bottom navigation bar.	System navigates to the “Register” screen, displaying available time.

Alternative Flows:	N/A
Exceptions:	N/A

Priority:	High
Frequency of Use:	High
Business Rules:	N/A
Other Information:	List of available time of next week is displayed by default.
Assumptions:	N/A

Table 45 <Use Case> View available time

2.6.6 Create available time

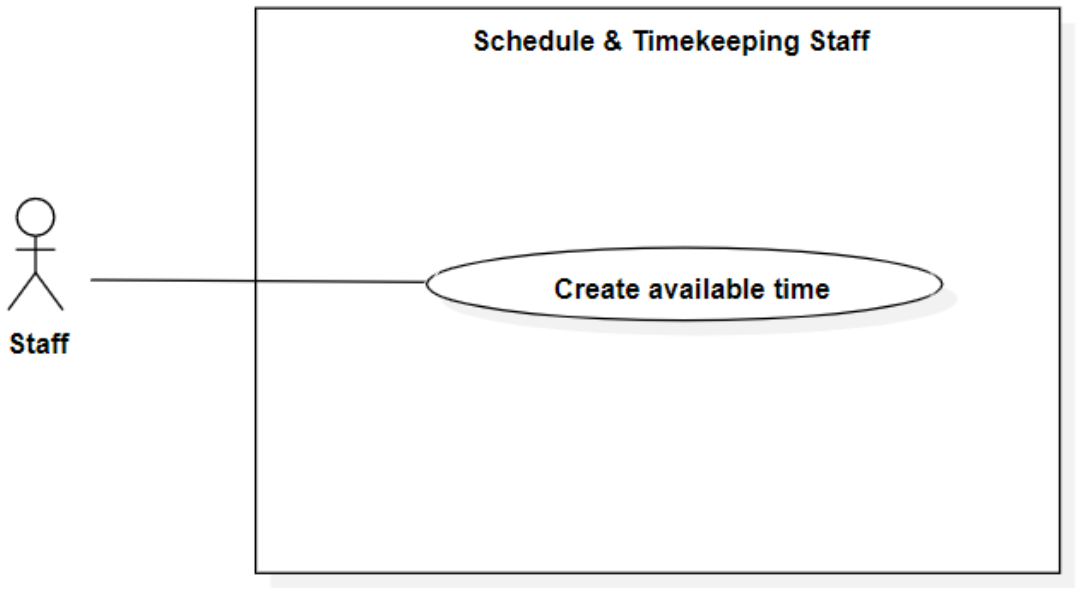


Figure 47 <Use Case> Create available time

UC ID and Name:	**MD5.06 Create available time**		
Created By:	TrungDQ	Date Created:	21/06/2021
Primary Actor:	Staff	Secondary Actors:	N/A
Trigger:	Staff requests to create new available time.		
Description:	Staff creates available time that he/she can work on next week.		
Preconditions:	PRE-1: Staff has logged into the application. PRE-2: Current screen is “Register” screen. PRE-3: Selected week must be in the future.		

<table>
<tr><td>Post-conditions:</td><td>POST-1: New available time is added to list.</td></tr>
<tr><td>Normal Flow:</td><td>
<table>
<tr><th>Step</th><th>Actor Action</th><th>System Response</th></tr>
<tr><td>1</td><td>Staff clicks on the “+” button.</td><td>System displays create available time dialog.</td></tr>
<tr><td>2</td><td>Staff select the time range and clicks “Save” button.
[Exception 1]</td><td>New available time is added to the system</td></tr>
<tr><td></td><td></td><td>System reloads available time list and displays.</td></tr>
</table>
</td></tr>
<tr><td>Alternative Flows:</td><td>N/A</td></tr>
<tr><td>Exceptions:</td><td>
<table>
<tr><th>No</th><th>Cause</th><th>System Response</th></tr>
<tr><td>1</td><td>Selected time range is not valid</td><td>System displays an error message.</td></tr>
</table>
</td></tr>
<tr><td>Priority:</td><td>High</td></tr>
<tr><td>Frequency of Use:</td><td>High</td></tr>
<tr><td>Business Rules:</td><td>Can not register duplicate time.</td></tr>
<tr><td>Other Information:</td><td>N/A</td></tr>
<tr><td>Assumptions:</td><td>N/A</td></tr>
</table>

Table 46 <Use Case> View available time

2.6.7 Edit available time

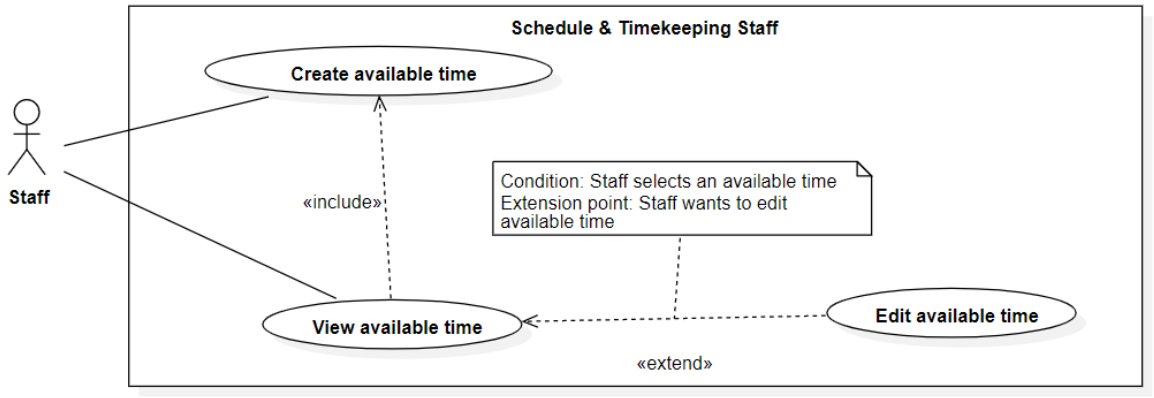


Figure 48 <Use Case> Edit available time

UC ID and Name:	**MD5.07 Edit available time**		
Created By:	TrungDQ	Date Created:	21/06/2021
Primary Actor:	Staff	Secondary Actors:	N/A
Trigger:	Staff requests to edit available time.		
Description:	Staff edits available time in list that he/she can work on next week.		
Preconditions:	PRE-1: Staff has logged into the application. PRE-2: Current screen is “Register” screen. PRE-3: Selected week must be in the future.		
Post-conditions:	POST-1: Selected available time is edited.		
Normal Flow:			

Step	Actor Action	System Response
1	Staff selects time he/she wants to edit.	System displays edit available time dialog.
2	Staff selects the time range and clicks the “Update” button to update available time, or clicks the “Delete” button to delete available time.. [Exception 1]	Request is saved to the system.
		System reloads available time list and displays.

<table>
<tr><td></td><td></td></tr>
<tr><td>Alternative Flows:</td><td>N/A</td></tr>
<tr><td>Exceptions:</td><td>
<table>
<tr><th>No</th><th>Cause</th><th>System Response</th></tr>
<tr><td>1</td><td>Selected time range is not valid</td><td>System displays error dialog.</td></tr>
</table>
</td></tr>
<tr><td>Priority:</td><td>High</td></tr>
<tr><td>Frequency of Use:</td><td>Low</td></tr>
<tr><td>Business Rules:</td><td>Can not edit to duplicate time.</td></tr>
<tr><td>Other Information:</td><td>N/A</td></tr>
<tr><td>Assumptions:</td><td>N/A</td></tr>
</table>

Table 47 <Use Case> Edit available time

2.6.8 View attendance report

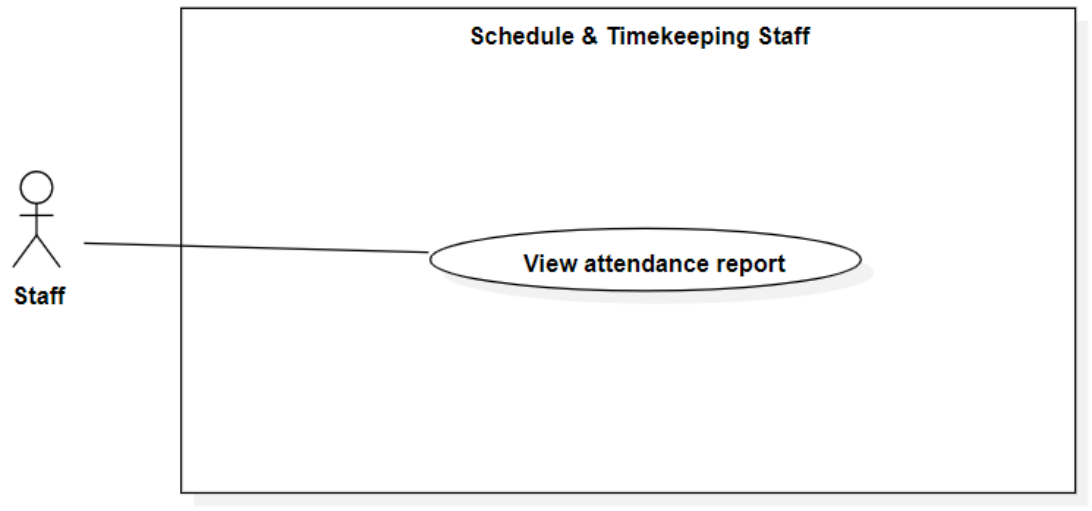


Figure 49 <Use Case> View attendance report

UC ID and Name:	**MD5.08 View attendance report**		
Created By:	TrungDQ	Date Created:	21/06/2021
Primary Actor:	Staff	Secondary Actors:	N/A
Trigger:	Staff requests to get attendance report.		

<table>
<tr><td>Description:</td><td>Staff views attendance report.</td></tr>
<tr><td>Preconditions:</td><td>PRE-1: Staff has logged into the application.
PRE-2: Current screen is “Profile” screen.</td></tr>
<tr><td>Post-conditions:</td><td>POST-1: Attendance report will be displayed.</td></tr>
<tr><td>Normal Flow:</td><td>
<table>
<tr><th>Step</th><th>Actor Action</th><th>System Response</th></tr>
<tr><td>1</td><td>Staff clicks on the “Attendance Report” button..</td><td>System navigates to “Attendance report” screen and displays attendance report..</td></tr>
</table>
</td></tr>
<tr><td>Alternative Flows:</td><td>N/A</td></tr>
<tr><td>Exceptions:</td><td>N/A</td></tr>
<tr><td>Priority:</td><td>Medium</td></tr>
<tr><td>Frequency of Use:</td><td>High</td></tr>
<tr><td>Business Rules:</td><td>N/A</td></tr>
<tr><td>Other Information:</td><td>N/A</td></tr>
<tr><td>Assumptions:</td><td>N/A</td></tr>
</table>

Table 48 View attendance report

2.6.9 View work report

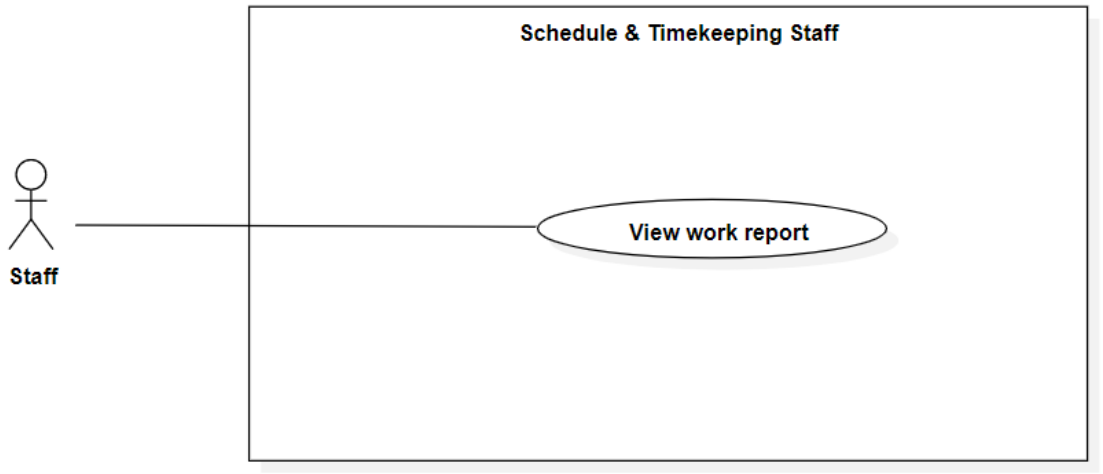


Figure 50 <Use Case> View work report

UC ID and Name:	**MD5.09 View work report**		
Created By:	TrungDQ	Date Created:	21/06/2021
Primary Actor:	Staff	Secondary Actors:	N/A
Trigger:	Staff requests to get work report.		
Description:	Staff views work report.		
Preconditions:	PRE-1: Staff has logged into the application. PRE-2: Current screen is “Profile” screen.		
Post-conditions:	POST-1: Work report will be displayed.		
Normal Flow:			

Step	Actor Action	System Response
1	Staff clicks on the “Work report” button.	System navigates to the “Work report” screen and displays work report..

Alternative Flows:	N/A
Exceptions:	N/A

Priority:	Medium
Frequency of Use:	Medium
Business Rules:	N/A
Other Information:	N/A
Assumptions:	N/A

Table 49 <Use Case> View attendance report

2.7 MD6. Timekeeper

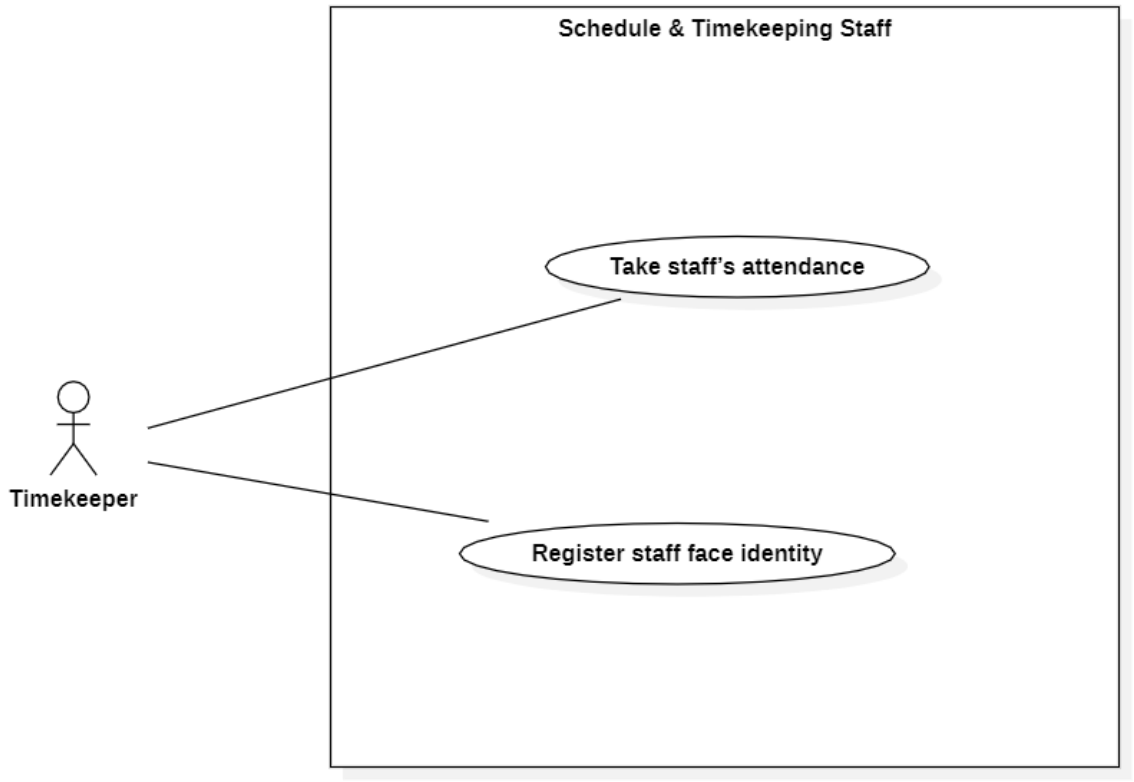


Figure 51 <Use Case Diagram> Timekeeper Use Case Diagram

2.7.1 Take staff's attendance

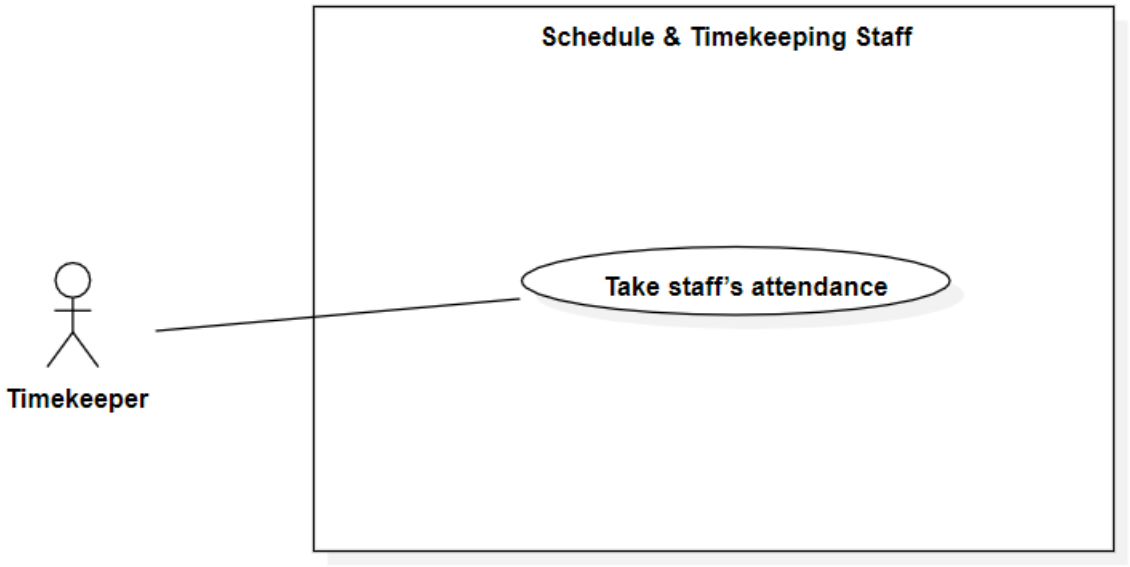


Figure 52 <Use Case> Take staff's attendance

UC ID and Name:	**MD6.1 Take staff's attendance**		
Created By:	DaoPM	Date Created:	12/06/2021
Primary Actor:	Staff	Secondary Actors:	N/A
Trigger:	Staff requests to take attendance.		
Description:	The staff wants to take attendance.		
Preconditions:	PRE-1: Staff use store face recognize device		
Post-conditions:	POST-1: Success: attendance recorded. Fail: return error message		
Normal Flow:			

Step	Actor Action	System Response
1	Press "Take attendance" button	Redirect to take attendance page.
2	Take photo of staff face	Send photo to the server for processing and return result.

Alternative Flows:	N/A
Exceptions:	<table><tr><th>No</th><th>Cause</th><th>System Response</th></tr><tr><td>1</td><td>Can not identify staff.</td><td>System returns error message.</td></tr></table>
Priority:	High
Frequency of Use:	High
Business Rules:	BR8-01, BR8-02, BR8-03
Other Information:	N/A
Assumptions:	N/A

Table 50 <Use Case> Take staff's attendance

2.7.2 Register staff face identity

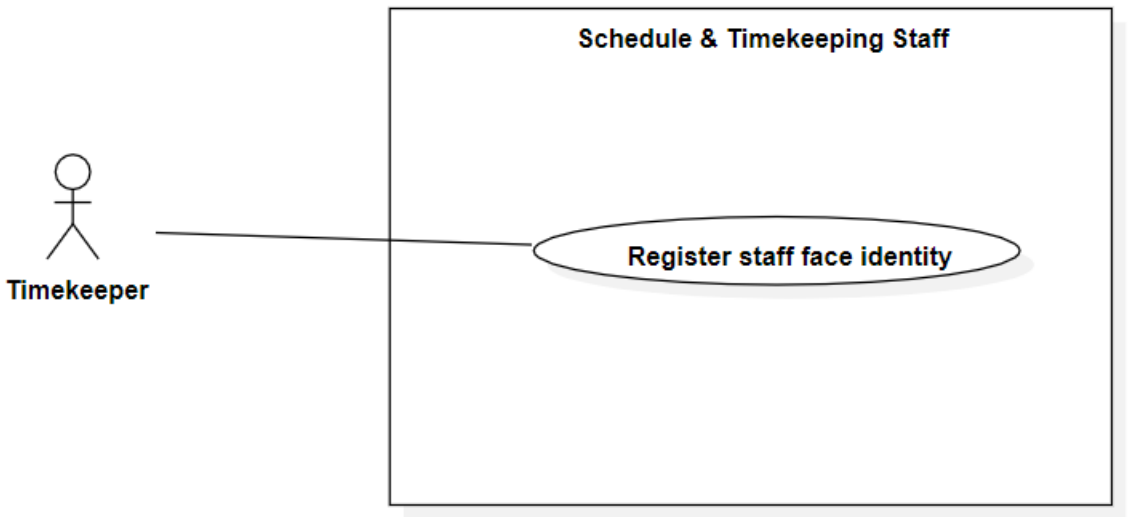


Figure 53 <Use Case> Register staff face identity

UC ID and Name:	**MD6.2 Register staff face identity**		
Created By:	DaoPM	Date Created:	12/06/2021
Primary Actor:	Staff	Secondary Actors:	N/A
Trigger:	The staff request to register face identity		

<table>
<tr><td>Description:</td><td>The staff register his/her face identity in store face recognition device.</td></tr>
<tr><td>Preconditions:</td><td>PRE-1: Staff use store recognition device</td></tr>
<tr><td>Post-conditions:</td><td>POST-1: Success: Face identity registered.</td></tr>
<tr><td>Normal Flow:</td><td>
<table>
<tr><th>Step</th><th>Actor Action</th><th>System Response</th></tr>
<tr><td>1</td><td>Press “Register” button</td><td>Redirect to register page.</td></tr>
<tr><td>2</td><td>Enter username & press “Record video” button</td><td>Redirect to record video page</td></tr>
<tr><td>3</td><td>Record video and press “save”</td><td>Video sent to server for processing & training data.</td></tr>
</table>
</td></tr>
<tr><td>Alternative Flows:</td><td>N/A</td></tr>
<tr><td>Exceptions:</td><td>N/A</td></tr>
<tr><td>Priority:</td><td>High</td></tr>
<tr><td>Frequency of Use:</td><td>Medium</td></tr>
<tr><td>Business Rules:</td><td>BR8-04</td></tr>
<tr><td>Other Information:</td><td>N/A</td></tr>
<tr><td>Assumptions:</td><td>Staff at the store with good light conditions.</td></tr>
</table>

Table 51 <Use Case> Register staff face identity

2.8 MD7. Admin

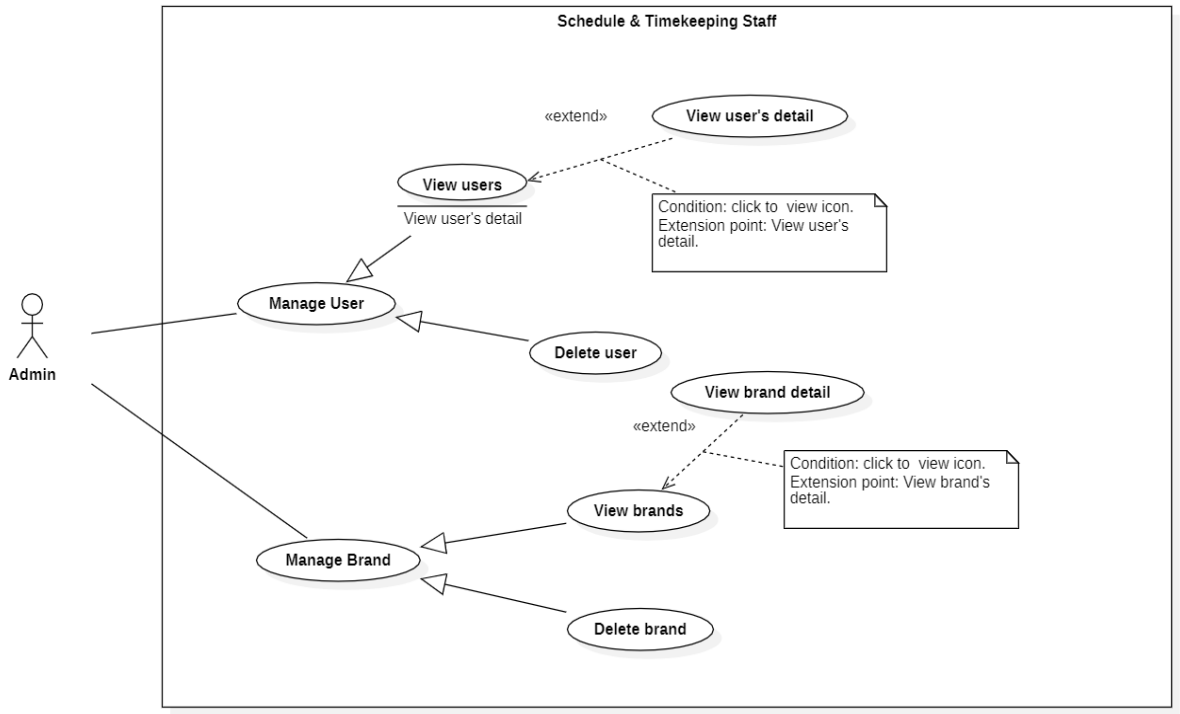


Figure 54 <Use Case Diagram> Admin Use Case Diagram

2.8.1 View users.

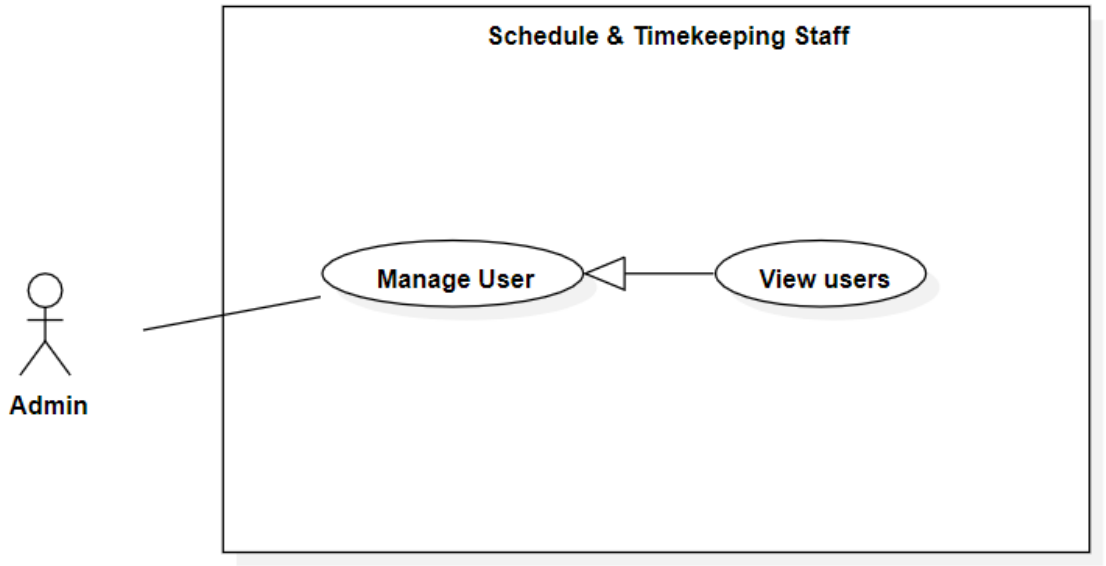


Figure 55 <Use Case> View users

<table>
<tr><td>UC ID and Name:</td><td colspan="3">MD7.1 View users</td></tr>
<tr><td>Created By:</td><td>DaoPM</td><td>Date Created:</td><td>24/05/2021</td></tr>
<tr><td>Primary Actor:</td><td>Admin</td><td>Secondary Actors:</td><td>N/A</td></tr>
<tr><td>Trigger:</td><td colspan="3">The Admin requested to view users.</td></tr>
<tr><td>Description:</td><td colspan="3">The Admin wants to view a list of users.</td></tr>
<tr><td>Preconditions:</td><td colspan="3">PRE-1: Users have logged in as role admin.</td></tr>
<tr><td>Post-conditions:</td><td colspan="3">POST-1: Success: list of users return with paging</td></tr>
<tr><td>Normal Flow:</td><td colspan="3"><table><tr><th>Step</th><th>Actor Action</th><th>System Response</th></tr><tr><td>1</td><td>Admin click “User” item on menu</td><td>Return page to view user list</td></tr></table></td></tr>
<tr><td>Alternative Flows:</td><td colspan="3">N/A</td></tr>
<tr><td>Exceptions:</td><td colspan="3">N/A</td></tr>
<tr><td>Priority:</td><td colspan="3">High</td></tr>
<tr><td>Frequency of Use:</td><td colspan="3">High</td></tr>
<tr><td>Business Rules:</td><td colspan="3">N/A</td></tr>
<tr><td>Other Information:</td><td colspan="3">N/A</td></tr>
<tr><td>Assumptions:</td><td colspan="3">N/A</td></tr>
</table>

Table 52 <Use Case> View users

2.8.2 Add user.

UC ID and Name:	**MD7.2 Add user**		
Created By:	DaoPM	Date Created:	24/05/2021
Primary Actor:	Admin	Secondary Actors:	N/A
Trigger:	The Admin requested to add a user.		
Description:	Admin want to create a new user		
Preconditions:	PRE-1: User has to log in as role admin. PRE-2: User in view users tab.		

<table>
<tr><td>Post-conditions:</td><td>POST-1: success: new user created.
fail: show error message.</td></tr>
<tr><td>Normal Flow:</td><td>
<table>
<tr><th>Step</th><th>Actor Action</th><th>System Response</th></tr>
<tr><td>1</td><td>Admin click “ADD USER” button</td><td>Pop up Add User form</td></tr>
<tr><td>2</td><td>Fill in the information in the form: “Email”, “Username”, “Is admin”, “Password”, “Confirm”</td><td></td></tr>
<tr><td>3</td><td>Click “Confirm”</td><td>- The system checks if the user input is valid.
- Create new user.</td></tr>
</table>
</td></tr>
<tr><td>Alternative Flows:</td><td>N/A</td></tr>
<tr><td>Exceptions:</td><td>
<table>
<tr><th>No</th><th>Cause</th><th>System Response</th></tr>
<tr><td>1</td><td>Fields are required.</td><td>System returns error message label that fields is required</td></tr>
<tr><td>2</td><td>Username must be unique</td><td>System returns an error message “Username already exists”.</td></tr>
<tr><td>3</td><td>“Password” does not match with “Confirm”</td><td>System returns error message “Confirm must match Password”</td></tr>
</table>
</td></tr>
<tr><td>Priority:</td><td>Medium</td></tr>
<tr><td>Frequency of Use:</td><td>Low</td></tr>
<tr><td>Business Rules:</td><td>Email: email form
Username: text input, required</td></tr>
</table>

	Is Admin: check box Password: password input Confirm: password input
Other Information:	N/A
Assumptions:	N/A

Table 53 <Use Case> Add user

2.8.3 View user detail.

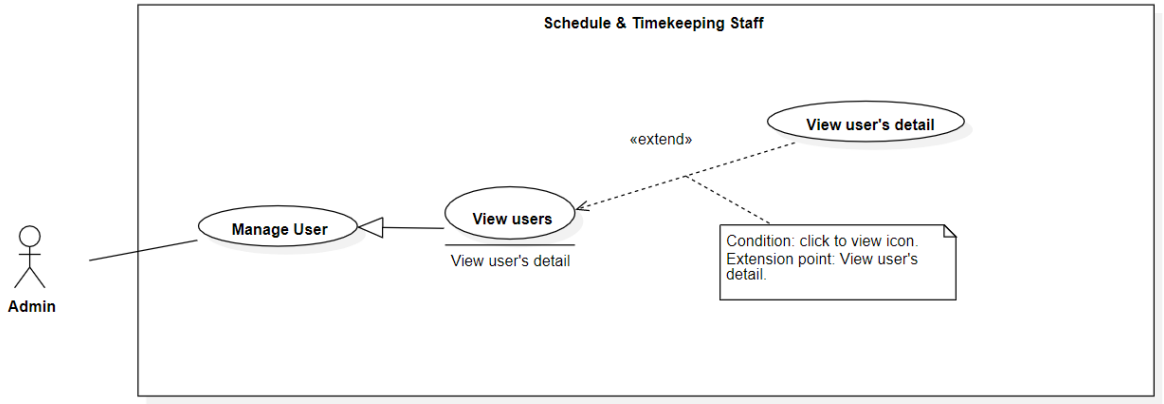


Figure 56 <Use Case> View User Detail

UC ID and Name:	**MD7.3 View user detail**		
Created By:	DaoPM	Date Created:	24/05/2021
Primary Actor:	Admin	Secondary Actors:	N/A
Trigger:	The Admin requested to view user details.		
Description:	The Admin wants to view details of the user.		
Preconditions:	PRE-1: Users have logged in as role admin. PRE-2: User in view users tab.		
Post-conditions:	POST-1: Success: return user details information		
Normal Flow:			

Step	Actor Action	System Response
1	Admin click the eye icon on actions column in user row	redirect page to view user detail

Alternative Flows:	N/A
Exceptions:	N/A
Priority:	Medium
Frequency of Use:	Medium
Business Rules:	N/A
Other Information:	N/A
Assumptions:	N/A

Table 54 <Use Case> View user detail

2.8.4 Delete user.

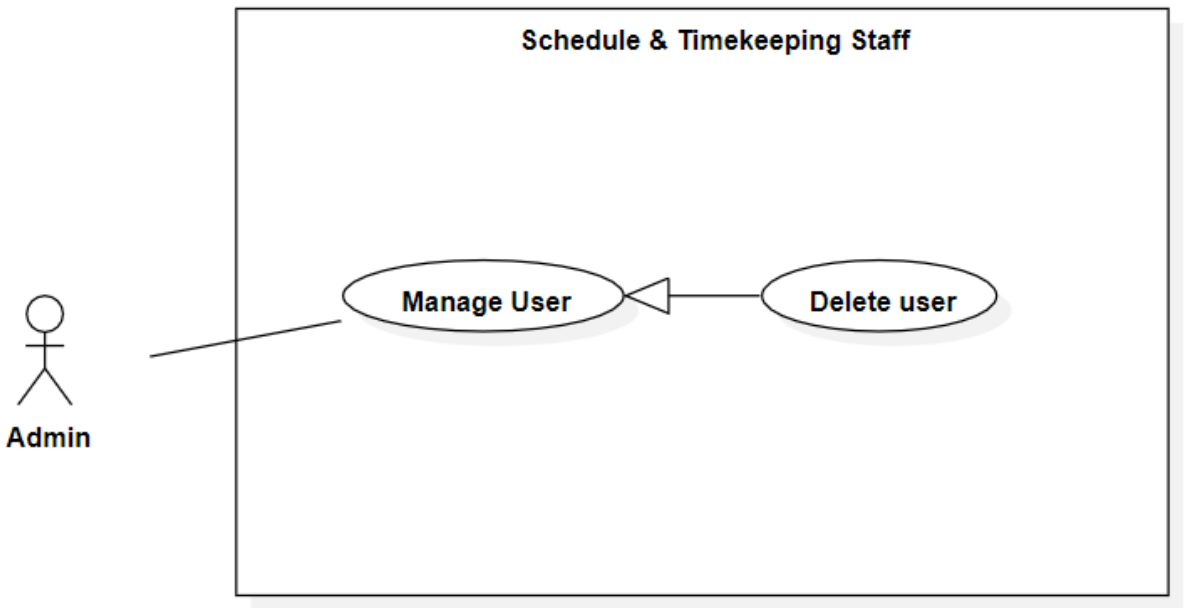


Figure 57 <Use Case> Delete user

UC ID and Name:	**MD7.4 Delete user**		
Created By:	DaoPM	Date Created:	24/05/2021
Primary Actor:	Admin	Secondary Actors:	N/A
Trigger:	The Admin requested to delete a user.		
Description:	The Admin delete user		
Preconditions:	PRE-1: User has to log in as role admin.		

<table>
<tr><td></td><td>PRE-2: User in “User” tab.</td></tr>
<tr><td>Post-conditions:</td><td>POST-1: success: deactivate user.</td></tr>
<tr><td>Normal Flow:</td><td>
<table>
<tr><th>Step</th><th>Actor Action</th><th>System Response</th></tr>
<tr><td>1</td><td>Admin click the red “X” icon at the Actions column in the user row</td><td>Pop up “Do you want to delete user: <<username>>”</td></tr>
<tr><td>2</td><td>Admin click “Confirm”</td><td>User deactivated</td></tr>
</table>
</td></tr>
<tr><td>Alternative Flows:</td><td>N/A</td></tr>
<tr><td>Exceptions:</td><td>N/A</td></tr>
<tr><td>Priority:</td><td>High</td></tr>
<tr><td>Frequency of Use:</td><td>Low</td></tr>
<tr><td>Business Rules:</td><td>N/A</td></tr>
<tr><td>Other Information:</td><td>N/A</td></tr>
<tr><td>Assumptions:</td><td>N/A</td></tr>
</table>

Table 55 <Use Case> Delete user

2.8.5 View brands.

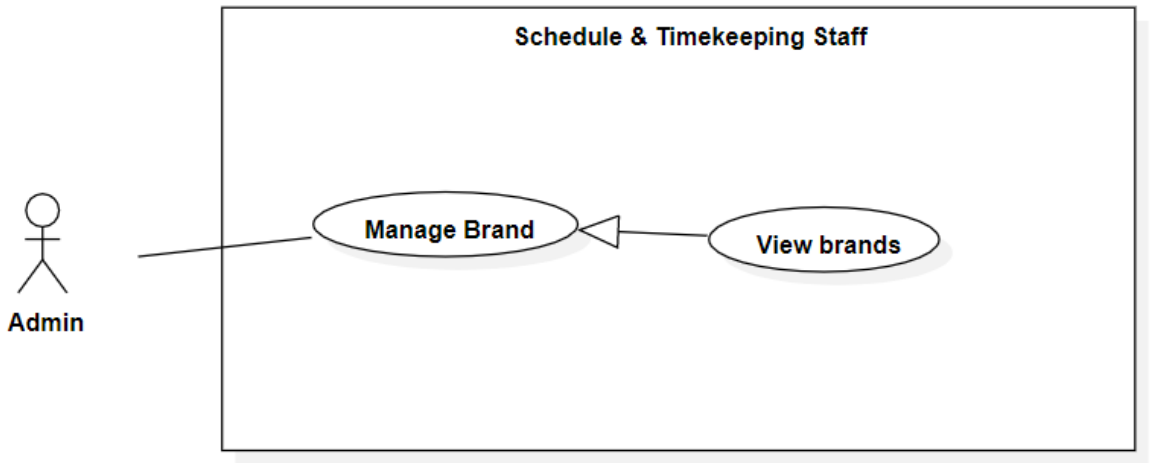


Figure 58 <Use Case> View brands

<table>
<tr><td>UC ID and Name:</td><td colspan="3">MD7.5 View brands</td></tr>
<tr><td>Created By:</td><td>DaoPM</td><td>Date Created:</td><td>24/05/2021</td></tr>
<tr><td>Primary Actor:</td><td>Admin</td><td>Secondary Actors:</td><td>N/A</td></tr>
<tr><td>Trigger:</td><td colspan="3">The Admin requested to view brands.</td></tr>
<tr><td>Description:</td><td colspan="3">The Admin view the brands</td></tr>
<tr><td>Preconditions:</td><td colspan="3">PRE-1: Users have logged in as role admin.</td></tr>
<tr><td>Post-conditions:</td><td colspan="3">POST-1: Success: list of brands return with paging</td></tr>
<tr><td>Normal Flow:</td><td colspan="3"><table><tr><th>Step</th><th>Actor Action</th><th>System Response</th></tr><tr><td>1</td><td>Admin click “Company” item on menu</td><td>Return page to view brand list</td></tr></table></td></tr>
<tr><td>Alternative Flows:</td><td colspan="3">N/A</td></tr>
<tr><td>Exceptions:</td><td colspan="3">N/A</td></tr>
<tr><td>Priority:</td><td colspan="3">High</td></tr>
<tr><td>Frequency of Use:</td><td colspan="3">High</td></tr>
<tr><td>Business Rules:</td><td colspan="3">N/A</td></tr>
<tr><td>Other Information:</td><td colspan="3">N/A</td></tr>
<tr><td>Assumptions:</td><td colspan="3">N/A</td></tr>
</table>

Table 56 <Use Case> View brands

2.8.6 View brand detail.

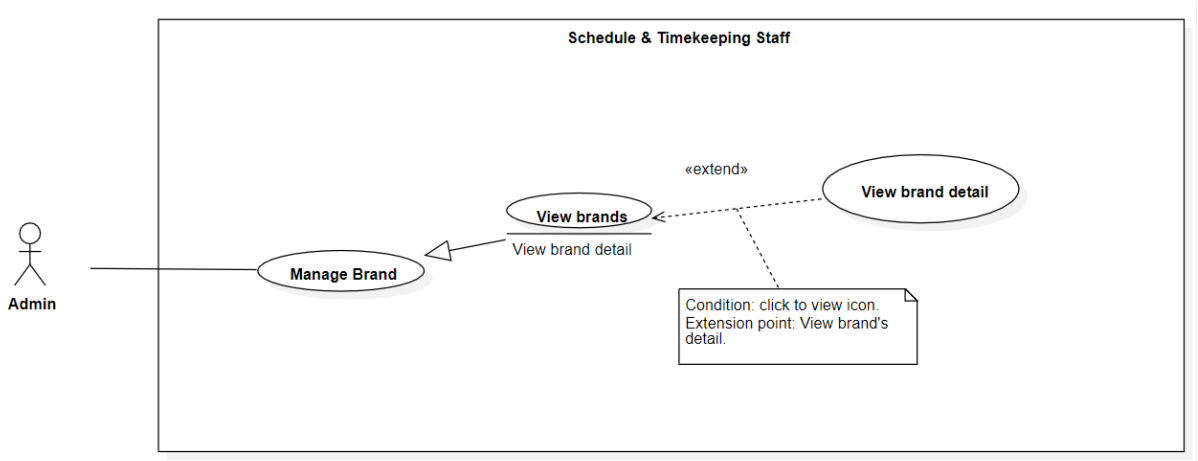


Figure 59 <Use Case> View brand detail

UC ID and Name:	**MD7.6 View brand detail**		
Created By:	DaoPM	Date Created:	24/05/2021
Primary Actor:	Admin	Secondary Actors:	N/A
Trigger:	The Admin requested to view brand details.		
Description:	The Admin wants to view details of the brand.		
Preconditions:	PRE-1: Users have logged in as role admin. PRE-2: User in view brands tab.		
Post-conditions:	POST-1: Success: return brand details information		
Normal Flow:	<table><tr><th>Step</th><th>Actor Action</th><th>System Response</th></tr><tr><td>1</td><td>Admin click the eye icon on actions column in brand row</td><td>redirect page to view brand detail</td></tr></table>		
Alternative Flows:	N/A		
Exceptions:	N/A		
Priority:	Medium		
Frequency of Use:	Medium		
Business Rules:	N/A		

Other Information:	N/A
Assumptions:	N/A

Table 57 <Use Case> View brand detail

2.8.7 Delete brand.

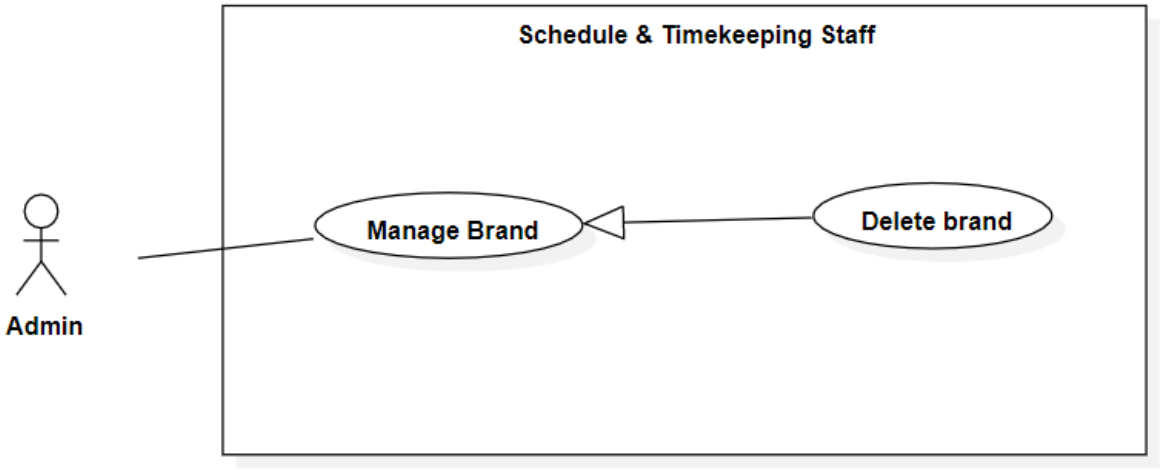


Figure 60 <Use Case> Delete brand

UC ID and Name:	**MD7.7 Delete brand**		
Created By:	DaoPM	Date Created:	24/05/2021
Primary Actor:	Admin	Secondary Actors:	N/A
Trigger:	The Admin requested to delete a brand.		
Description:	The Admin delete brand.		
Preconditions:	PRE-1: Users have logged in as role admin. PRE-2: User in “Company” tab.		
Post-conditions:	POST-1: success: brand deleted.		
Normal Flow:			

Step	Actor Action	System Response
1	Admin click the red “X” icon at the Actions column in the brand row	Pop up “Do you want to delete brand: <<brand name>>?”
2	Admin click “Confirm”	Brand deleted.

Alternative Flows:	N/A
Exceptions:	N/A
Priority:	Medium
Frequency of Use:	Medium
Business Rules:	N/A
Other Information:	N/A
Assumptions:	N/A

Table 58 <Use Case> Delete brand.

3. Functional Requirements

3.1 System Functional Overview

a. Screen Flow

- **Web admin Application Screen Flow**

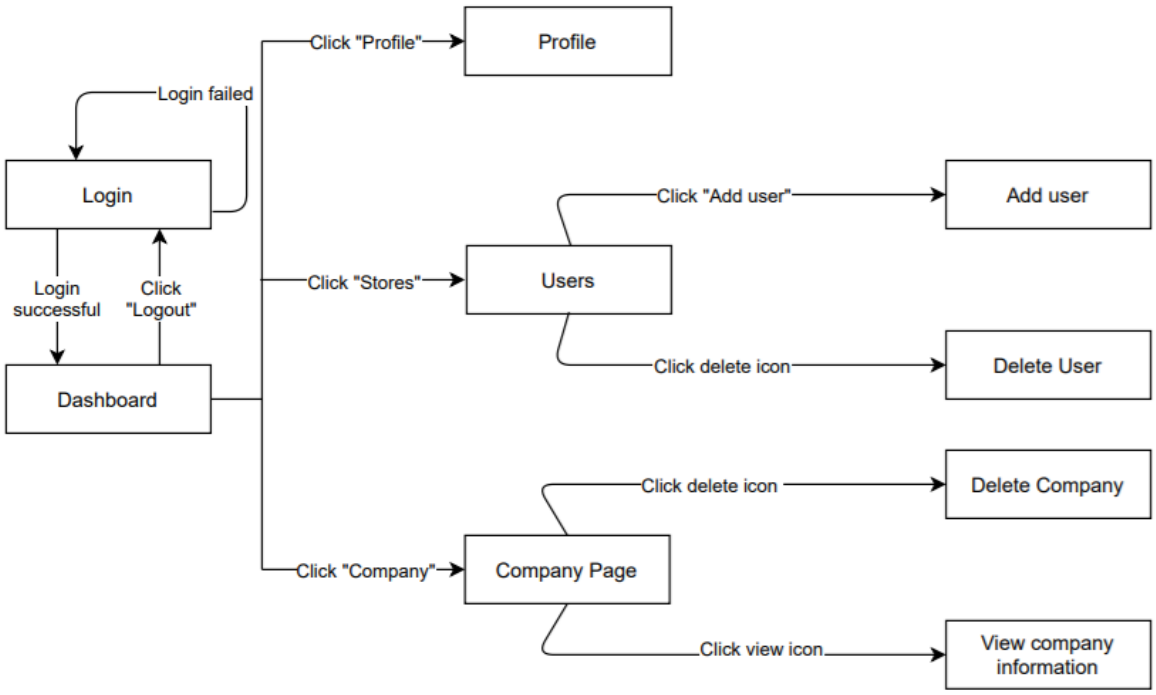


Figure 61 <Screenflow> Web Admin Application Screen Flow

- **Brand Manager Web Screen Flow**

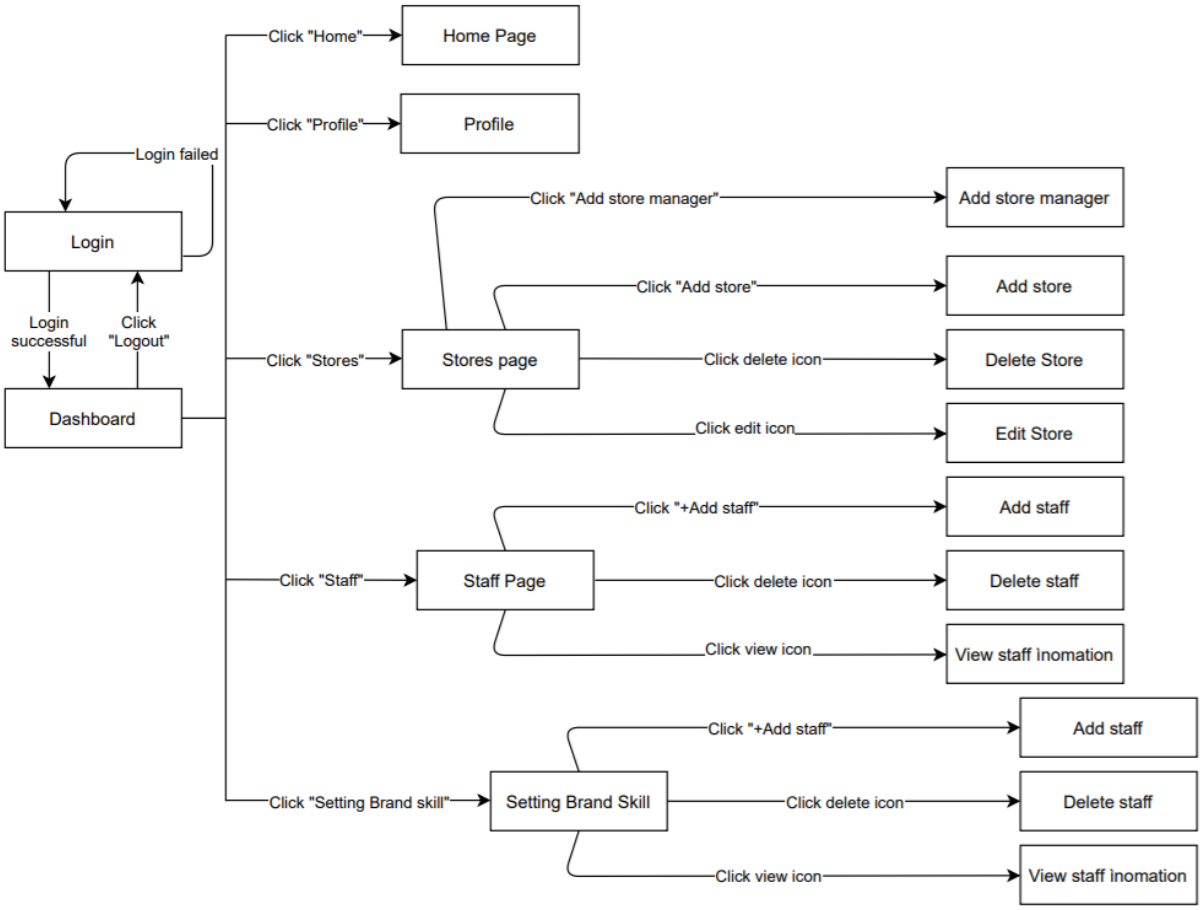


Figure 62 <Screenflow> Brand Manager Web Screen Flow

- **Store Manager Web Screen Flow**

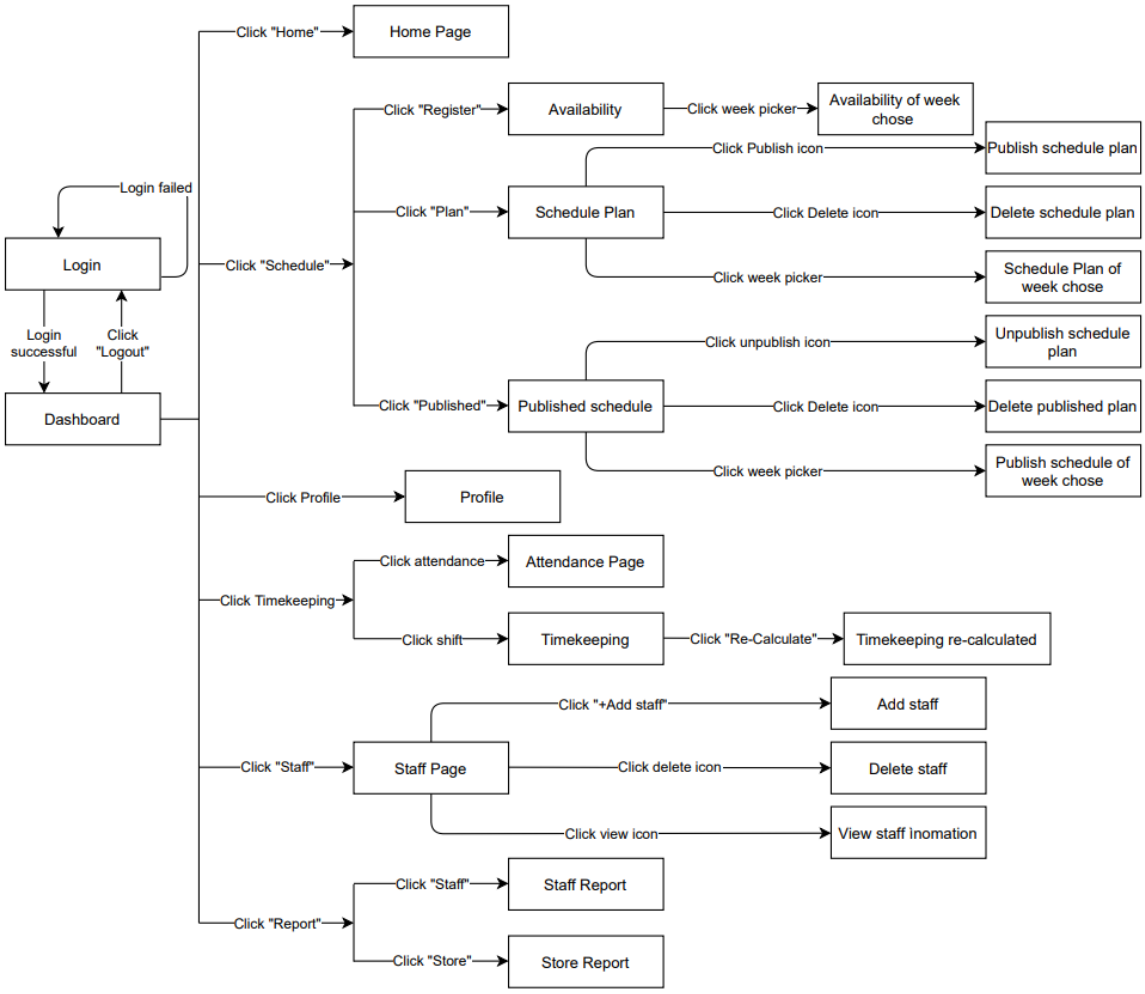


Figure 63 <Screenflow> Web Store Manager Application Screen Flow

- **Staff Mobile Application Screen Flow**

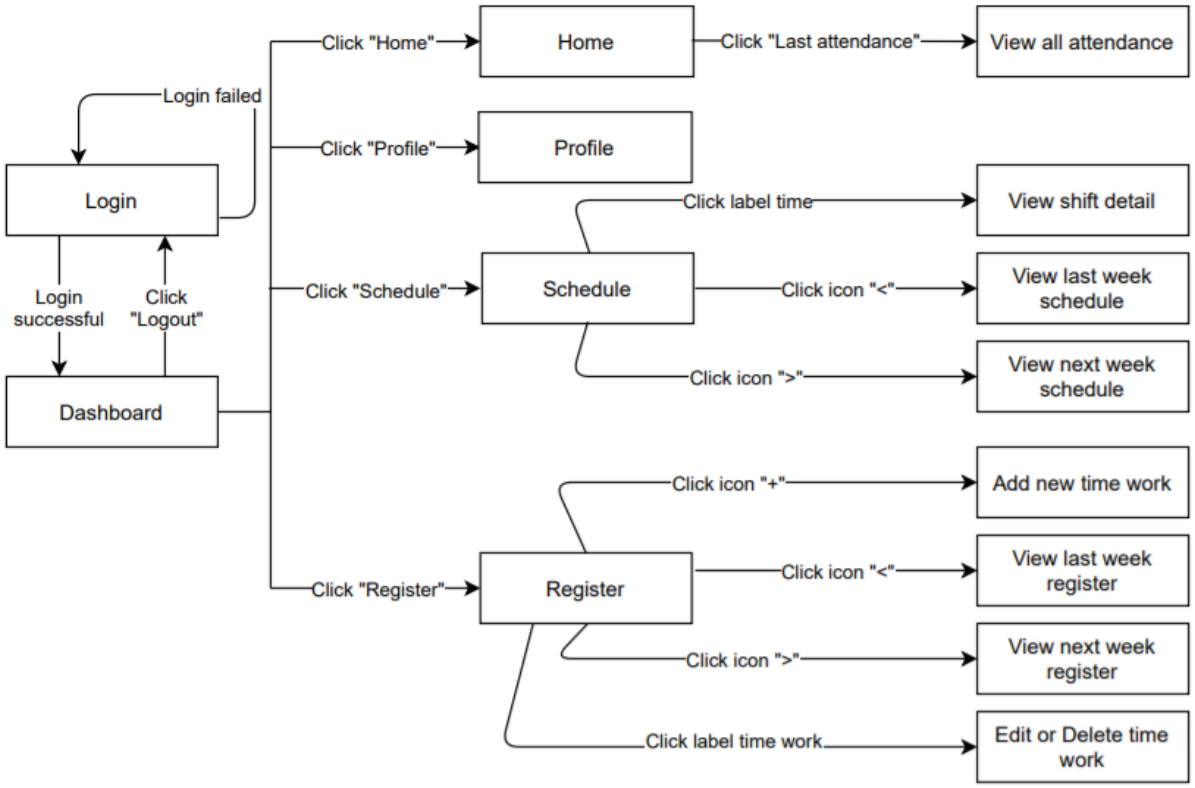


Figure 64 <Screen Flow> Staff Mobile Application Screen Flow

b. Screen Details

- **Web admin Application Screen Flow**

Id	Screen	Description
01	Login	Login into system
02	Profile	View personal information
03	Staff Page	View all staff and some information about it in tabular form.
04	Company Page	View all brands that have been registered into the system.

- **Brand Manager Web Screen Flow**

Id	Screen	Description
01	Login	Login into system
02	Home Page	View statistics about attendance of all stores and staff.
03	Profile	View personal information

04	Stores Page	View all stores and some information about it in tabular form.
05	Staffs Page	View all staff and some information about it in tabular form.
06	Setting Brand Skill	View all brand skills that apply to the whole system. The brand manager can edit or delete by clicking the icon in this page.
07	Add Store Manager	Add a new store manager and assign them to store needs.
08	Add Store	Add a new store in the chain.
09	Edit Store	Click the edit icon in the Store Page. The brand manager can change store information.
10	Add Staff	Add new staff and assign them to store needs.
11	View detail Staff	View staff's personal information in more detail.

- **Store Manager Web Screen Flow**

Id	Screen	Description
01	Login	Login into system
02	Home Page	View statistics about attendance of all stores and staff.
03	Availability	All registered staff hours information is displayed here
04	Schedule Plan	Store manager set many schedule plans for next week.
05	Published schedule	After setting the schedule, the store manager publishes the schedule for next week.
06	Profile	View personal information
07	Attendance Page	All attendance employee information including photos, equipment, hours, accuracy are recorded and displayed in the form of a table.
08	Timekeeping Page	Statistics of the total number of working hours, the number of times of being late and leaving early of staff.
09	Staff Page	View staff in the store.
10	Staff Report	View staff report
11	Store Report	View store report

- **Staff Mobile Application Screen Flow**

Id	Screen	Description
01	Login	Login into system.
02	Home	View last attendance, shift upcoming, total working time.
03	Profile	View user information.
04	Schedule	View shift assignments.
05	Register	View and register available time.

06	View all attendance	View attendances.
07	View shift detail	View shift detail.

c. Screen Authorization

Screen	Unauthenticated User	Admin	Brand Manager	Store Manager	Staff
Login	X				
Profile	X				
Change password	X				
Stores Page			X		
Add Store			X		
Delete Store			X		
Edit Store			X	X	
Staff Page			X	X	
Add Staff			X	X	
View detail			X	X	X
Delete Staff			X	X	
User Page		X			
View detail		X			
Delete User		X			
Company Page		X			
Edit Company		X	X		
Delete Company		X			
Home Page			X	X	
Setting Brand Skill			X		
Add new skill			X		
Delete skill			X		
Update skill			X		
Add Manager Store			X		
Availability				X	
Schedule Plan				X	
Constraints				X	
Demands				X	
View				X	

Published schedule				X	
Constraints				X	
Demands				X	
View				X	
Attendance Page			X	X	
Timekeeping Page			X	X	
View detail			X	X	
Staff Report			X	X	
Store report			X	X	

d. Non-Screen Functions

#	System Function	Description
1	Compute schedule	Use the algorithm using constraints to output the result of the schedule.
2	Face recognition	Use the app at the store device to scan your face and record attendance.

e. Entity Relationship Diagram

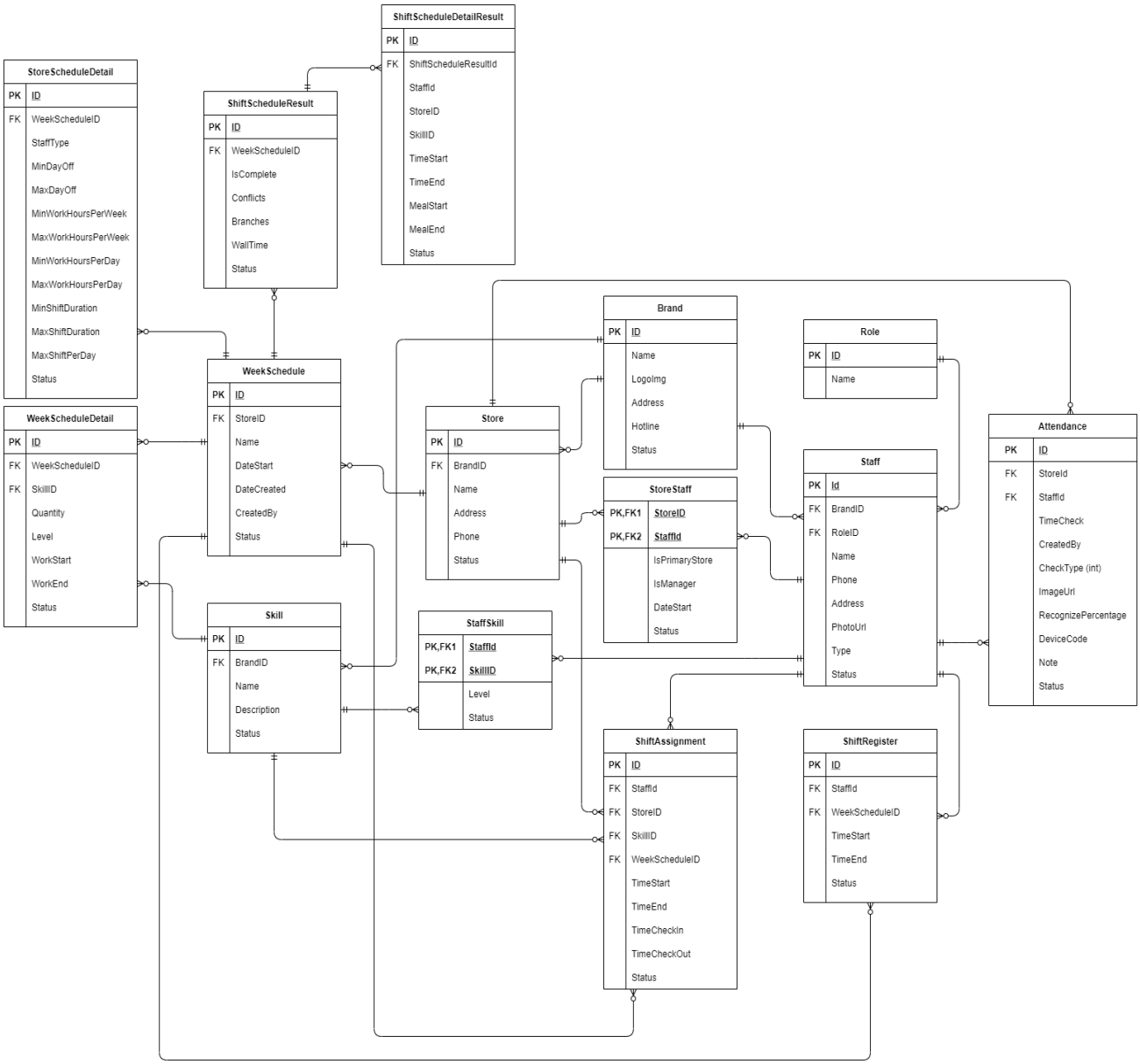


Figure 65 Entity Relationship Diagram

#	Entity	Description
1	Role	Contains roles for staff, use for application authorization.
2	Staff	Contains personal information of staff. Such as name, phone, address and its role in the system.

3	Brand	Contains information about the brand: Name, Logo, Address,...
4	Store	Contains information about the store: Name, Phone, Address,...
5	StoreStaff	Represent stores staff work at. One staff at work at many stores, but only one primary store.
6	Skill	Represent skill description at specific brand.
7	StaffSkill	Represent skills that staff have and its level.
8	ShiftAssignment	Contains information about one shift: Which staff work, at when, where and which skill,...
9	WeekSchedule	Contains information of one week schedule, Its date start, its status (unpublished, published),...
10	ShiftScheduleResult	Contains information of algorithm result overview: Conflicts, is it complete or not,...
11	ShiftScheduleDetailResult	Contains information of algorithm result detail: Specific information for each shift, includes staff id, store id, skill id,...
12	StoreScheduleDetail	Define store’s week schedule constraints, full time staff or part time staff.
13	WeekScheduleDetail	Define the store's week schedule demands, the needs at specific times.
14	Attendance	Records at the time which staff took attendance. Contains information such as staff id, time check, store id,...

4. Non-Functional Requirements

4.1 External Interfaces

a. User Interfaces

UI-1: The web application in the system shall follow Material-UI design.

UI-2: The mobile application in the system shall follow Neumorphic design.

UI-3: The UI must be friendly and easy to use.

b. Software Interfaces

SI-1: SQL Server 2016: used for relational data storage.

- The database server is secured with a SQL Server account, so the .NET client must use a specific username/password to connect to the server.

SI-2: Use an Email API to send email.

c. Hardware Interfaces

Time attendance equipment at the store must have a camera.

d. Communications Interfaces

CI-1: The STS shall send an email to the user whenever they change their password or create a new account.

4.2 Quality Attributes

a. Usability

- The UI designed by Material Design for ease to use.
- Managers and staff can use their application/web application frequently after 1- day training.
- It's easier for staff to take attendance with facial recognition.

b. Reliability

- STS provides manual actions for the scheduling process to prevent situations: staff not register schedule for next week, the algorithm does not solve,...
- Prevent employees from taking attendance for others.
- Ensure staff's actual working time is not cheated through the attendance control system.

c. Performance

- The running time of the scheduling algorithm is no more than 5 minutes.
- Face recognition time to take attendance should not exceed 30 seconds.
- Accuracy rate of face recognition over 80%.

d. Dependability

d1. Security

- Each user is only allowed to access the functions of its own role.
- The password will be hashed when saved to the database.

e. Supportability

- Support Android devices from version 8 and above.

f. Design Constraints

- The system uses TypeScripts and JavaScripts to increase maintainability.
- Web service is developed using Node.js, using its architecture advantages.
- Web modules are developed using ReactJS library.
- Mobile applications use Flutter and Neumophic to design.

h. Purchased Components

- Firebase Storage: limit 5GB storage total, 50000 requests or 1GB bandwidth, per day
- AWS: limit 15G server per month.

IV. Software Design Description

1. Overall Description

1.1 Assumptions

This system is designed basing on these following assumptions:

- Windows 10 MacOS Big Sur
- Google Chrome 92
- Android 9 / iOS 14
- SQL Server 2016
- User have stable internet connection

1.2 Design Constraints

This system should be complied with following items:

- Coding language:
    - .Net Core for Restful API server
    - Flask for face recognize server

    - Flutter for staff mobile app and store device
    - ReactJS for front-end web app
- Clients include web app and mobile devices must be connected to the internet to use the system
- Most processes in the system must be less than 10 seconds
- Scheduling Algorithm must be less than 5 minutes
- Normal interacts between servers through HTTP/HTTPS requests
- Data for schedule algorithm is sent to message queue server and consumed by algorithm server

2. System Architecture Design

2.1 Overall Architecture

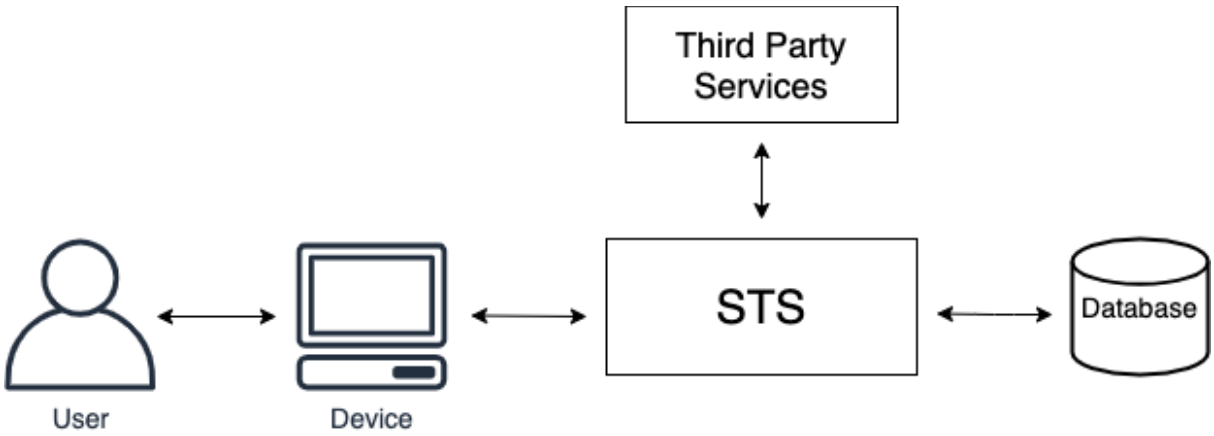


Figure 66 Overall Architecture

2.2 System Architecture

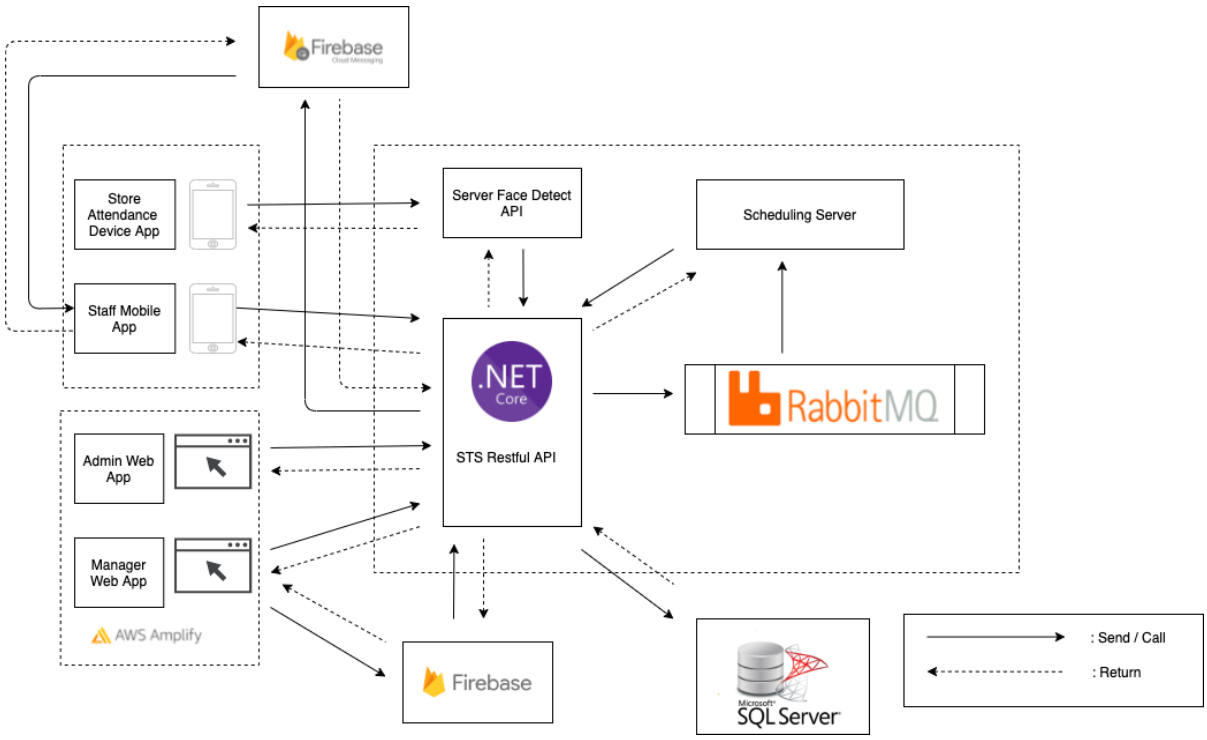


Figure 67 System Architecture

Description:

- **Store Attendance Device App**: Mobile app at store for staff to take attendance and register face identity. It interacts with the Face Detect API server through HTTP Protocol.
- **Staff Mobile App**: Mobile app for staff, interacts with STS Restful API Server through HTTP Protocol.
- **Admin Web App & Manager Web App**: Front-end web application, interact with STS Restful API server through HTTP Protocol.
- **Firebase Cloud Messaging**: Cloud Messaging Service for sending notifications.
- **Firebase**: Google Cloud Service for storage, realtime database.
- **RabbitMQ**: Message Queue Server to store algorithm data waiting to be processed.
- **Scheduling Server**: Consume data from RabbitMQ to process, after finished, processed data are sent to STS Restful API Server through HTTP Protocol.
- **Face Detect Server**: Receive data from Store Attendance Device to process and return results.
- **STS Restful API Server**: Process HTTP requests from clients.

- **Database**: MS SQL Server, an Relational Database that is connected with the server through TCP/IP protocol.

2.3 Package Diagram

2.3.1 Front-end Web Manager

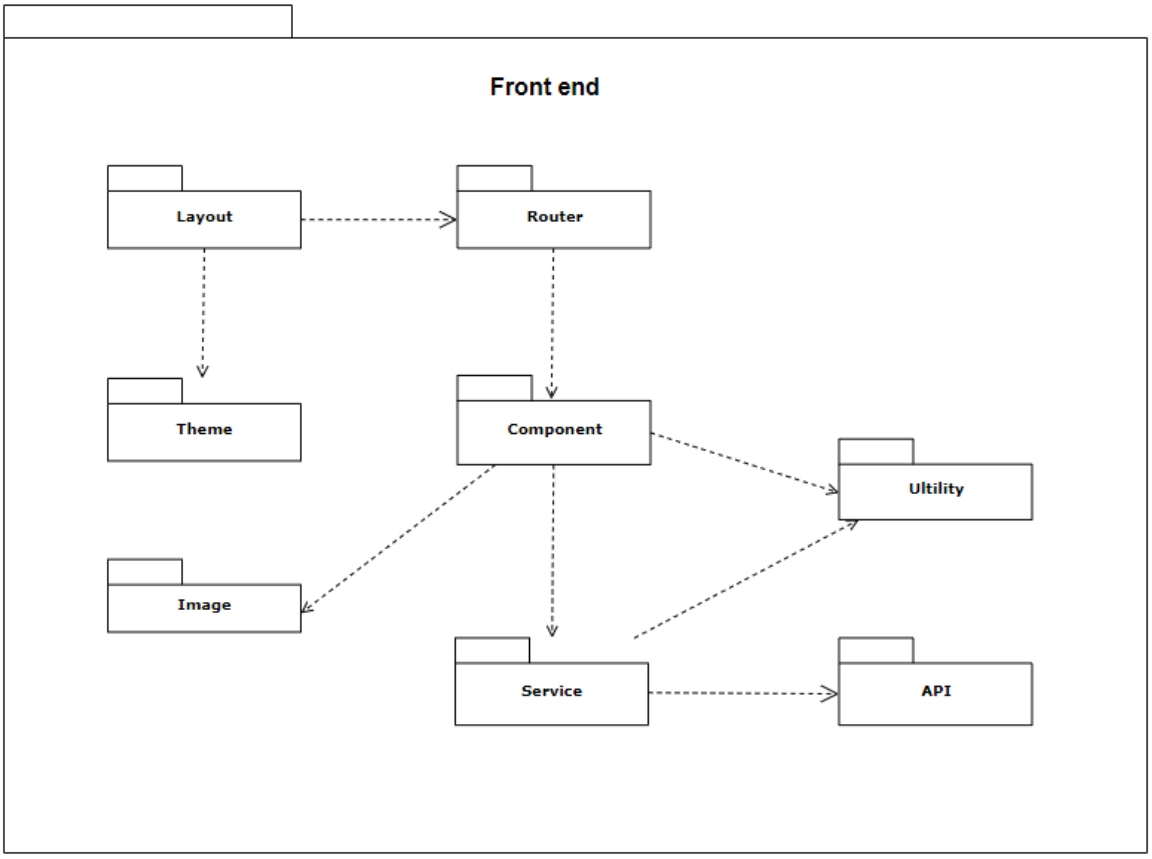


Figure 68 <Package Diagram> Front-end Web Manager

No	Package	Description
01	Layout	Package contains general Component is used for most pages
02	Theme	Package contains functions for configuring a common UI for all Components
03	Router	Package contains functions that link a URL to the appropriate page

04	Component	Package contains functions to handle presentation
05	Image	Package contains Images file is used for inclusion in the Component
06	Service	Package contains functions to handle business logic, connect API with Component
07	Utility	Package contains functions to handle repetitive computations
08	API	Package contains functions to interact with the backend,...

2.3.2 STS Restful API

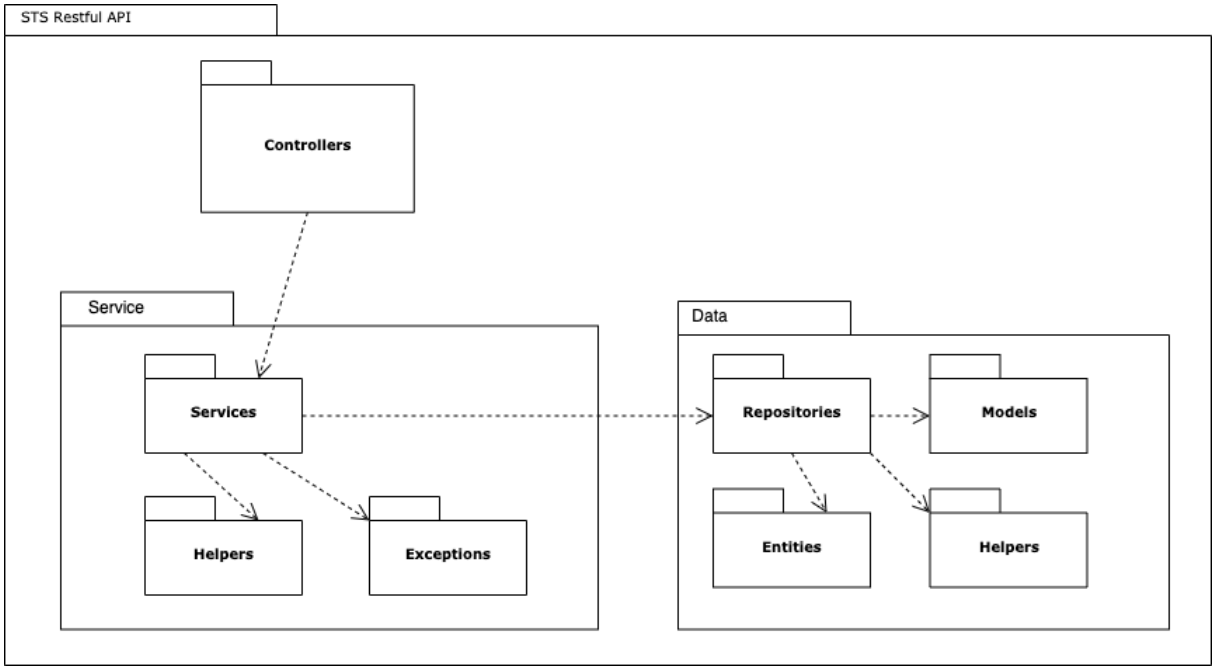


Figure 69 <Package Diagram> STS Restful API Server

No	Package	Description

01	Controllers	Package contains Controller classes to handle HTTP requests and responses
02	Service.Services	Package contains classes to handle business logic, interact with data,...
03	Service.Helpers	Package contains utility classes for services
04	Service.Exceptions	Package contains Exceptions may happen in application
05	Data.Repositories	Package contains classes to query and update data from database
06	Data.Entities	Package contains entities classes that mapping with database tables
07	Data.Models	Package contains model classes that can be mapped between entities and models. Use to make request & return response from API
08	Data.Helpers	Package contains utility classes for data package

3. System Detailed Design

3.1 Class Diagram

3.1.1 Class Diagram

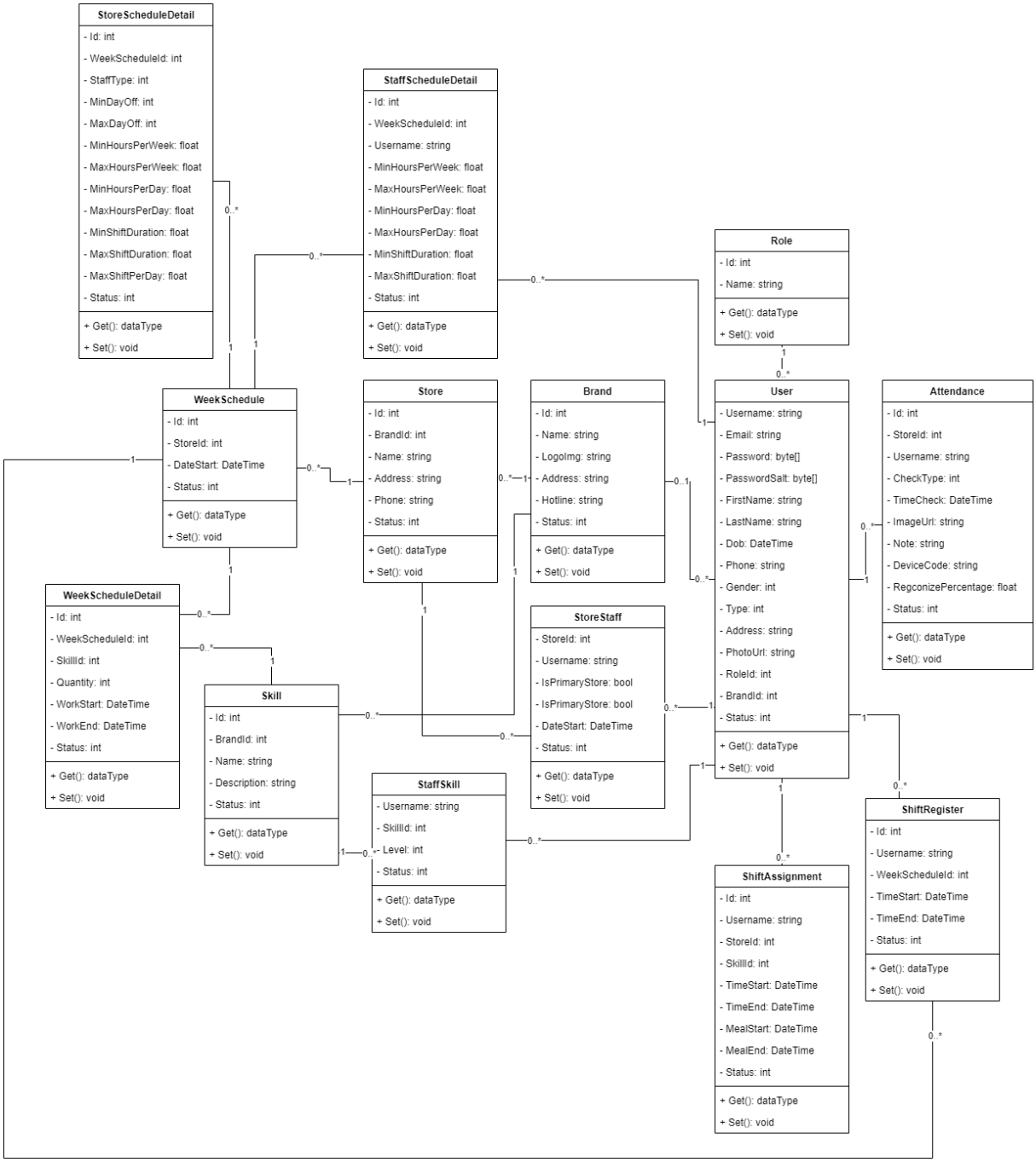


Figure 70 Class Diagram

3.1.2 Class Specification

3.1.2.1 Role Class

Field Name	Type	Description
Id	int	The id of role in the system
Name	string	The role name in the system: admin, brand manager, store manager, staff

Table 59 <Class Diagram Attributes> Role

Method	Type	Description
Get	data type	Get the attribute value
Set	void	Set the attribute value

Table 60 <Class Diagram Method> Role

3.1.2.2 User Class

Field Name	Type	Description
Username	string	The id of user
Name	string	The role name in the system: admin, brand manager, store manager, staff
Email	string	The email of this user
Password	byte[]	The hashed password of this user
PasswordSalt	byte[]	The password salt of this user
FirstName	string	The first name of this user
LastName	string	The last name of this user
Dob	DateTime	The date of birth of this user
Phone	string	The phone number of this user
Gender	int	The gender of this user: Female, Male, Other
Type	int	The type of this user: full time, part time
Address	string	The address of this user
PhotoUrl	string	The photo url of this user
RoleId	int	The role id of this user
BrandId	int	The brand id of this user
Status	int	The status of this user: deleted, active

Table 61 <Class Diagram Attributes>User

Method	Type	Description
Get	data type	Get the attribute value
Set	void	Set the attribute value

Table 62 <Class Diagram Method> User

3.1.2.3 Brand Class

Field Name	Type	Description
Id	int	The id of brand in the system
Name	string	The brand name in the system
LogoImg	string	The link to brand's logo image
Address	string	The address of brand
Hotline	string	The brand's hotline
Status	int	The status of this brand: deleted, active

Table 63 <Class Diagram Attributes> Brand

Method	Type	Description
Get	data type	Get the attribute value
Set	void	Set the attribute value

Table 64 <Class Diagram Method> Brand

3.1.2.4 Store Class

Field Name	Type	Description
Id	int	The id of store in the system
Name	string	The store name in the system
BrandId	int	The brandId related to this store
Address	string	The address of store
Phone	string	The store's phone number
Status	int	The status of this store: deleted, active

Table 65 <Class Diagram Attributes> Store

Method	Type	Description
Get	data type	Get the attribute value
Set	void	Set the attribute value

Table 66 <Class Diagram Method> Store

3.1.2.5 StoreStaff Class

Field Name	Type	Description
StoreId	int	The id of store in the system
Username	string	The user's username in the system
IsPrimaryStore	bool	The value define if this is user's main store
IsManager	bool	The value define the user is manager of this store or not
DateStart	DateTime	The date time staff start working here
Status	int	The status of this StoreStaff: deleted, active

Table 67 <Class Diagram Attributes> StoreStaff

Method	Type	Description
Get	data type	Get the attribute value
Set	void	Set the attribute value

Table 68 <Class Diagram Method> StoreStaff

3.1.2.6 Skill Class

Field Name	Type	Description
Id	int	The id of skill in the system
BrandId	int	The brandId related to this skill
Name	string	The skill's name
Description	string	The skill's description
Status	int	The status of this Skill: deleted, active

Table 69 <Class Diagram Attributes> Skill

Method	Type	Description
Get	data type	Get the attribute value
Set	void	Set the attribute value

Table 70 <Class Diagram Method> Skill

3.1.2.7 StaffSkill Class

Field Name	Type	Description
Username	string	The username of user in this system
SkillId	int	The skillId that the user have
Level	int	The skill's level of user
Status	int	The status of this StaffSkill: deleted, active

Table 71 <Class Diagram Attributes> StaffSkill

Method	Type	Description
Get	data type	Get the attribute value
Set	void	Set the attribute value

Table 72 <Class Diagram Method> StaffSkill

3.1.2.8 ShiftAssignment Class

Field Name	Type	Description
Id	int	The id of ShiftAssignment in this system
Username	string	The Username of user assigned to this shift
WeekScheduleId	int	The Id of WeekSchedule related to this ShiftAssignment
StoreId	int	The Id of Store that assigned to this shift
SkillId	int	The Id of Skill that assigned to this shift
TimeStart	DateTime	The time shift begin
TimeEnd	DateTime	The time shift end
TimeCheckIn	DateTime	The time staff check in
TimeCheckOut	DateTime	The time staff check out
Status	int	The status of this ShiftAssignment: deleted, active

Table 73 <Class Diagram Attributes> ShiftAssignment

Method	Type	Description
Get	data type	Get the attribute value

Set	void	Set the attribute value

Table 74 <Class Diagram Method> ShiftAssignment

3.1.2.9 WeekSchedule Class

Field Name	Type	Description
Id	int	The id of ShiftAssignment in this system
StoreId	int	The Id of Store that assigned to this shift
Name	string	The week schedule name
CreatedBy	string	The name of user who created this week schedule
DateStart	DateTime	The time week schedule begins
DateCreated	DateTime	The time week schedule created
Status	int	The status of this ShiftAssignment: deleted, active

Table 75 <Class Diagram Attributes> WeekSchedule

Method	Type	Description
Get	data type	Get the attribute value
Set	void	Set the attribute value

Table 76 <Class Diagram Method> WeekSchedule

3.1.2.10 ShiftScheduleResult Class

Field Name	Type	Description
Id	int	The id of ShiftScheduleResult in this system
IsComplete	bool	The flag to mark if the schedule is completed
Conflicts	long	The number of conflicts the algorithm return
Branches	long	The number of branches the algorithm return
WallTime	double	The number of seconds algorithm took to solve
WeekScheduleId	int	The Id of WeekSchedule this ShiftScheduleResult related to
Status	int	The status of this ShiftScheduleResult: deleted, active

Table 77 <Class Diagram Attributes> ShiftSchedule

Method	Type	Description
Get	data type	Get the attribute value

Set	void	Set the attribute value

Table 78 <Class Diagram Method> Shift Schedule

3.1.2.11 ShiftScheduleDetailResult Class

Field Name	Type	Description
Id	int	The id of ShiftScheduleDetailResult in this system
Username	string	The id of user related to this schedule result
StoreId	int	The id of store which the user assigned to
SkillId	int	The id of skill which the user assigned to
ShiftSchedule ResultId	int	The id of ShiftScheduleResult this ShiftScheduleDetailResult related to
TimeStart	DateTime	The time when the shift start
TimeEnd	DateTime	The time when the shift end
Status	int	The status of this ShiftScheduleDetailResult: deleted, active

Table 79 <Class Diagram Attributes> ShiftScheduleDetailResult Class

Method	Type	Description
Get	data type	Get the attribute value
Set	void	Set the attribute value

Table 80 <Class Diagram Method> ShiftScheduleDetailResult Class

3.1.2.12 StoreScheduleDetail Class

Field Name	Type	Description
Id	int	The id of ShiftScheduleDetailResult in this system
WeekSchedul eId	int	The Id of WeekSchedule this ShiftScheduleResult related to
StaffType	StaffType	The enum class represent type of staff: FullTime, PartTime, All, Undefine
MinDayOff	int	The min day in week the staff do not need to work
MaxDayOff	int	The max day in week the staff do not need to work

MinHoursPerWeek	float	The min hours in week the staff need to work
MaxHoursPerWeek	float	The max hours in week the staff can work
MinHoursPerDay	float	The min hours per day the staff need to work
MaxHoursPerDay	float	The max hours per day the staff can work
MinShiftDuration	float	The min hours per shift that the staff need to work
MaxShiftDuration	float	The max hours per shift that the staff can work
MaxShiftPerDay	int	The max number of shifts the staff can work in one day
Status	int	The status of this StoreScheduleDetail: deleted, active

Table 81 <Class Diagram Attributes > StoreScheduleDetail Class

Method	Type	Description
Get	data type	Get the attribute value
Set	void	Set the attribute value

Table 82 <Class Diagram Method> StoreScheduleDetail Class

3.1.2.13 WeekScheduleDetail Class

Field Name	Type	Description
Id	int	The id of ShiftScheduleDetailResult in this system
WeekScheduleId	int	The Id of WeekSchedule this WeekScheduleDetail related to
SkillId	int	The id of skill this WeekScheduleDetail need
Level	int	The level this WeekScheduleDetail need
Quantity	int	The staff quantity needed in this WeekScheduleDetail
WorkStart	DateTime	The time when this shift start
WorkEnd	DateTime	The time when this shift end

Status	int	The status of this ShiftScheduleDetailResult: deleted, active

Table 83 <Class Diagram Attributes> WeekScheduleDetail

Method	Type	Description
Get	data type	Get the attribute value
Set	void	Set the attribute value

Table 84 <Class Diagram Method> WeekScheduleDetail

3.1.2.14 Attendance Class

Field Name	Type	Description
Id	int	The id of ShiftScheduleDetailResult in this system
StoreId	int	The Id of Store this Attendance related to
Username	string	The id of user who took attendance
TimeCheck	DateTime	The time user took attendance
CreateBy	string	The username of staff who took attendance or the manager who created it manually.
CheckeType	int	The method type when took attendance: 0 - face recognize, 1 - QR Code, 2 - finger print, 3 - manual
ImageUrl	string	The Image Url of the picture when staff take attendance
Note	string	The note of the attendance
RecognizePercentage	float	The percentage number compares to the base record in the model.
DeviceCode	string	The device code of the device which took the attendance.
Status	int	The status of this ShiftScheduleDetailResult: deleted, active

Table 85 <Class Diagram Attributes> Attendance

Method	Type	Description
Get	data type	Get the attribute value
Set	void	Set the attribute value

Table 86 <Class Diagram Method> Attendance

3.1.2.15 ShiftRegister Class

Field Name	Type	Description

Id	int	The id of ShiftRegister in this system
WeekScheduleId	int	The Id of WeekSchedule this Shift Register related to
Username	string	The id of user who register this shift
TimeStart	DateTime	The time staff available begin
TimeEnd	DateTime	The time staff available end
Status	int	The status of this ShiftRegister: deleted, active

Table 87 <Class Diagram Attributes> ShiftRegister

Method	Type	Description
Get	data type	Get the attribute value
Set	void	Set the attribute value

Table 88 <Class Diagram Method> ShiftRegister

3.1.3 Sequence Diagrams

3.1.3.1 Compute Schedule

Summary: This diagram illustrates the interaction that occurs in the system when the store manager presses the “Compute Schedule” button.

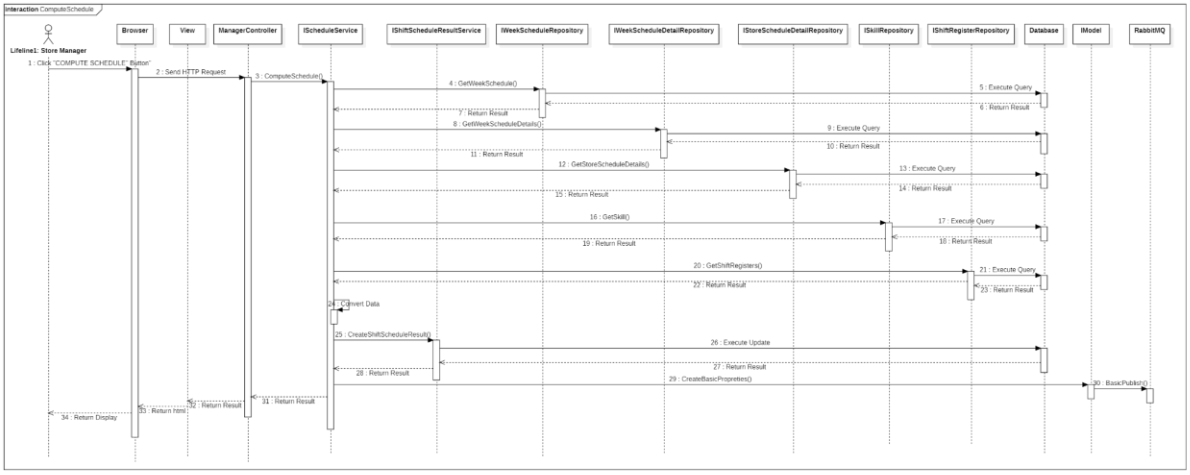


Figure 71 <Sequence Diagram> Compute Schedule

3.1.3.2 Publish Schedule

Summary: This diagram illustrates the interaction that occurs in the system when the store manager presses the “Publish” button.

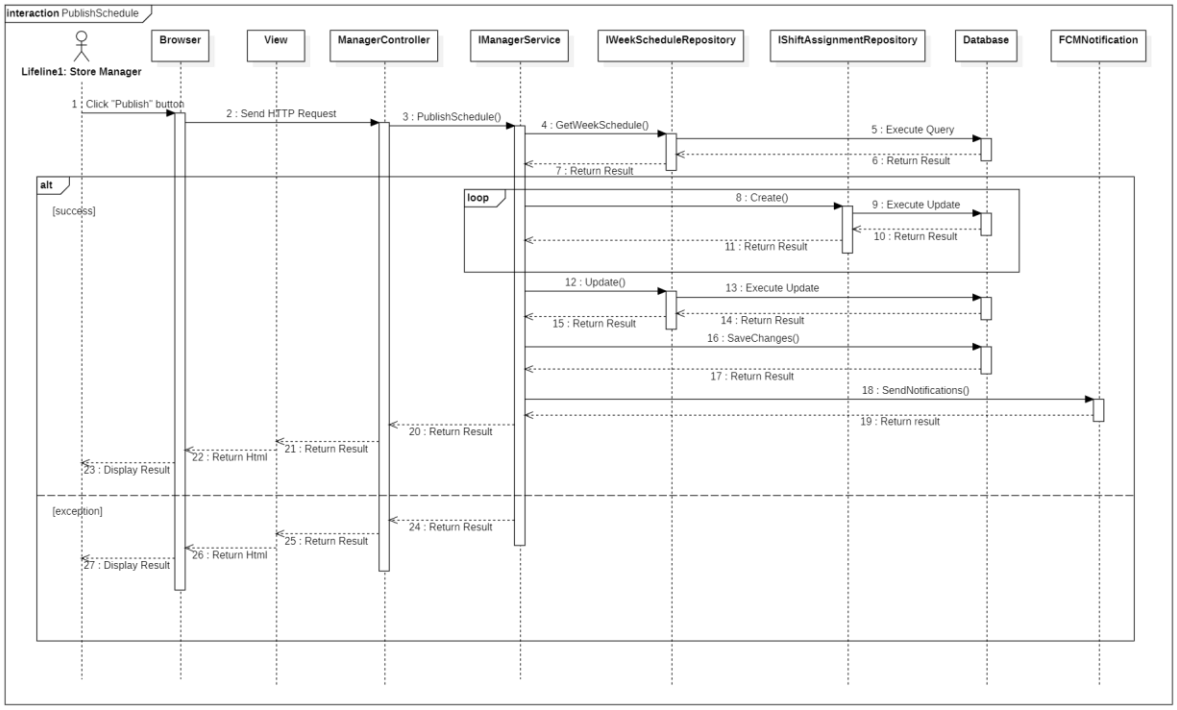


Figure 72 <Sequence Diagram> Publish Schedule

3.1.3.3 Calculate timekeeping

Summary: This diagram illustrates the interaction that occurs in the system when the store manager press the “RE-CALCULATE” button.

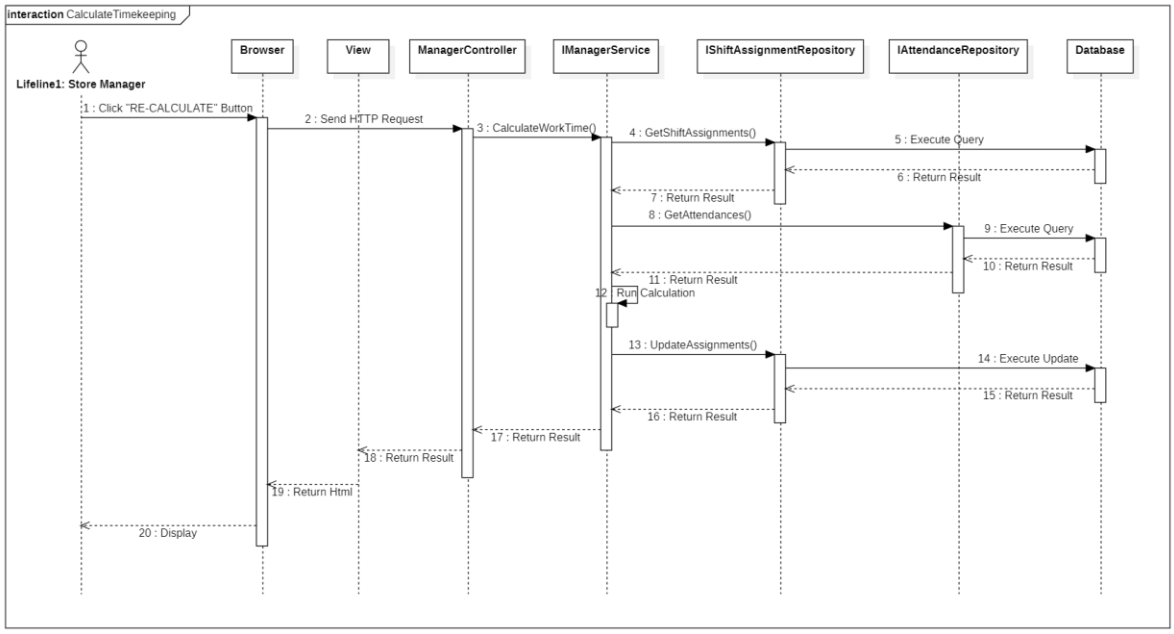


Figure 73 <Sequence Diagram> Calculate Timekeeping

3.1.3.4 Take attendance

Summary: This diagram illustrates the interaction that occurs in the system when the staff trigger takes attendance event.

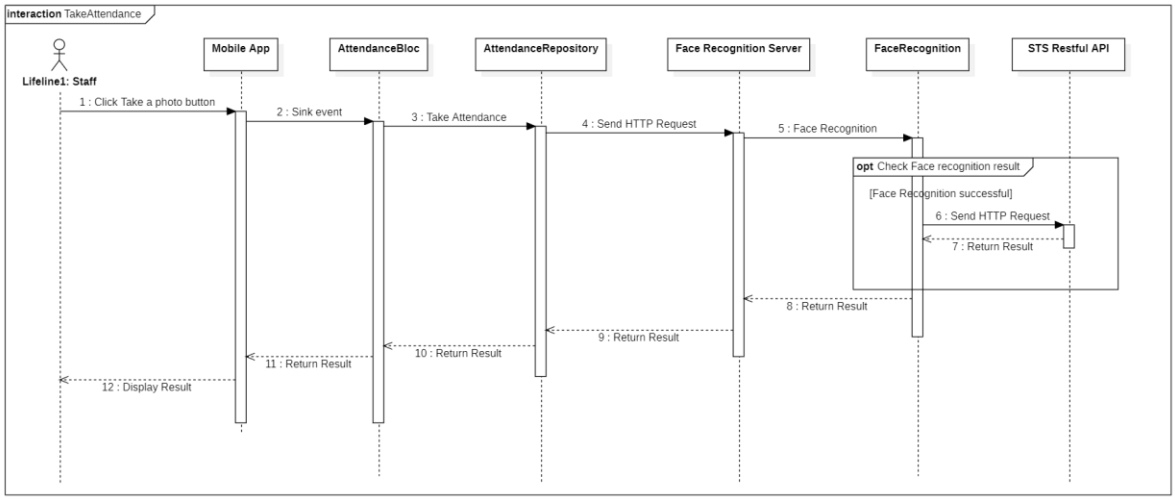


Figure 74 <Sequence Diagram> Take attendance

4. Data & Database Design

4.1 Database Design

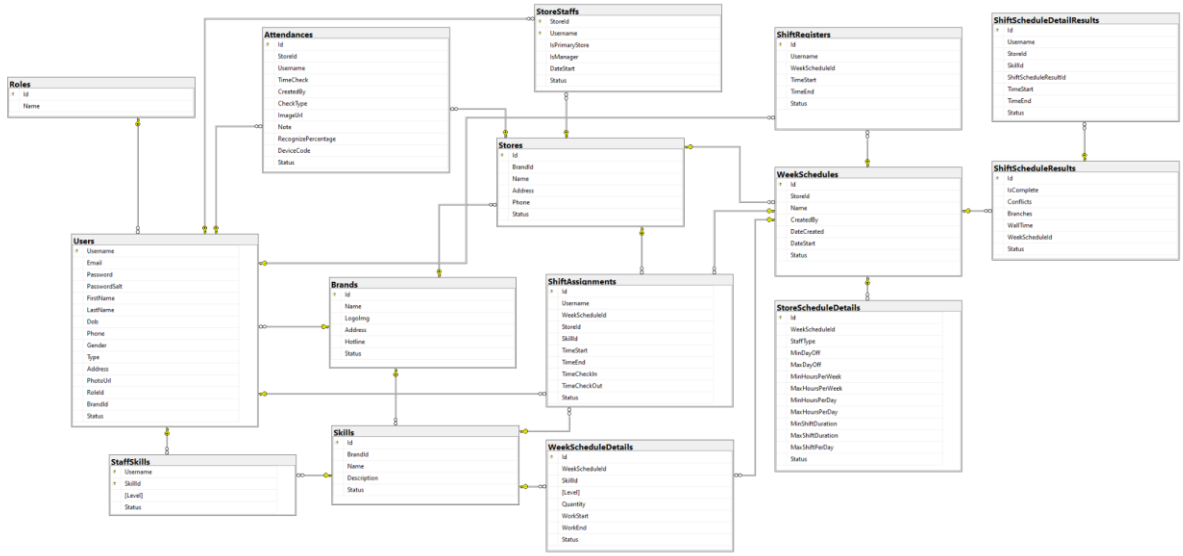


Figure 75 Physical Diagram

4.1.1 Role Table

Field name	Type	Size	Unique	Not Null	PK/FK	Notes
Id	int		x	x	PK	The id of the role
Name	varchar	50				The name of the role

Table 89 <Physical Diagram> Role

4.1.2 Users Table

Field name	Type	Size	Unique	Not Null	PK/FK	Notes
Username	varchar	50	x	x	PK	The id of the user

Email	varchar	100				The email of the user
Password	varbinary	MAX				The hashed password of user using SHA512 hash algorithm
PasswordSalt	varbinary	MAX				The password salt of user
FirstName	varchar	50				The first name of user
LastName	varchar	50				The last name of user
Dob	datetime2					The Date of birth of user
Phone	varchar	15				The phone number of user
Address	varchar	200				The address of user
PhotoUrl	varchar	MAX				The link to user's photo image
RoleId	int			x	FK	Permission for user can use specific actions in system
BrandId	int				FK	The id of the brand this user related to
Type	int			x		The type of user: 0 - Full time, 1 - Part time, 2 - All, 3 - Undefined
Gender	int			x		The gender of this user: 0 - female, 1 - male, 2 - other
Status	int			x		The status of this user: 0 - deleted, 1 - active

Table 90 <Physical Diagram> Users

4.1.3 Brands Table

Field name	Type	Size	Unique	Not Null	PK/FK	Notes
Id	int			x	PK	The id of this brand
Name	varchar	50		x		The name of this brand
LogoImg	varchar	MAX				The link to to logo image of this brand
Address	varchar	200				The address of this brand
Hotline	varchar	20				The hotline number of this Brand
Status	int			x		The status of this brand

Table 91 <Physical Diagram> Brands

4.1.4 Stores Table

Field name	Type	Size	Unique	Not Null	PK/FK	Notes
Id	int		x	x	PK	The id of store
BrandId	int			x	FK	The id of brand related to this store
Name	varchar	50		x		The name of this store
Address	varchar	200				The address of this store
Phone	varchar	20				The phone number of this store

Status	int			x		The status of this store

Table 92 <Physical Diagram> Stores

4.1.5 StoreStaffs Table

Field name	Type	Size	Unique	Not Null	PK/FK	Notes
StoreId	int		x	x	PK	The id of the store
Username	varchar	50		x	PK	The username of user work here
IsPrimaryStore	bit			x		Is the staff work mainly here
IsManager	bit			x		Is the user manager of this store
DateStart	datetime2			x		The time staff start working here
Status	int			x		The status of this StoreStaff

Table 93 <Physical Diagram> StoreStaffs

4.1.6 Skills Table

Field name	Type	Size	Unique	Not Null	PK/FK	Notes
Id	int		x	x	PK	The id of this skill
BrandId	int			x	FK	The id of brand this skill related to
Name	varchar	50		x		The name of this skill
Description	varchar	200				The description of this skill
Status	int			x		The status of this skill

Table 94 <Physical Diagram> Skills

4.1.7 StaffSkills Table

Field name	Type	Size	Unique	Not Null	PK/FK	Notes
Username	varchar	50	x	x	PK	The username of staff have this skill
SkillId	int			x	PK	The id of skill staff have
Level	int			x		The level staff have in this skill
Status	int			x		The status of this StaffSkill

Table 95 <Physical Diagram> StaffSkills

4.1.8 ShiftAssignments Table

Field name	Type	Size	Unique	Not Null	PK/FK	Notes
Id	int		x	x	PK	The id of this ShiftAssignment
Username	varchar	50		x	FK	The username of staff work in this shift
StoreId	int			x	FK	The id of the store the staff work in this shift

SkillId	int			x	FK	The id of the skill work in this shift
WeekScheduleId	int			x	FK	The id of the WeekSchedule related to this ShiftAssignment
TimeStart	datetime2			x		The time when the shift start
TimeEnd	datetime2			x		The time when the shift end
TimeCheckIn	datetime2			x		The time staff Check in
TimeCheckOut	datetime2			x		The time staff Check out
Status	int			x		The Status of this ShiftAssignment

Table 96 <Physical Diagram> ShiftAssignment

4.1.9 WeekSchedules Table

Field name	Type	Size	Unique	Not Null	PK/FK	Notes
Id	int			x	PK	The id of this WeekSchedule
StoreId	int			x	FK	The id of the store related to this WeekSchedule
Name	varchar	50				The Name of the WeekSchedule
DateStart	datetime2			x		The date start this WeekSchedule
DateCreated	datetime2			x		The date created this WeekSchedule
CreatedBy	varchar	50				The username of user who created this WeekSchedule
Status	int			x		The status of this WeekSchedule: 0 - Deleted, 2 - Published, 3 - Unpublished, 4 - Register

Table 97 <Physical Diagram> WeekSchedule

4.1.10 ShiftScheduleResults Table

Field name	Type	Size	Unique	Not Null	PK/FK	Notes
Id	int		x	x	PK	The id of this ShiftScheduleResult
WeekScheduleId	int			x	FK	The id of the WeekSchedule this Shift ScheduleResult related to
IsComplete	bit			x		Is the schedule done computing or not
Conflicts	int			x		The number of conflicts the algorithm return
Branches	int			x		The number of branches the algorithm return
WallTime	float			x		The time the algorithm compute

Status	int			x		The status of this ShiftScheduleResult

Table 98 <Physical Diagram> ShiftScheduleResults Table

4.1.11 ShiftScheduleDetailResults Table

Field name	Type	Size	Unique	Not Null	PK/FK	Notes
Id	int		x	x	PK	The id of this ShiftScheduleDetailResult
ShiftScheduleResultId	int			x	FK	The id of ShiftScheduleResult
Username	varchar	50		x		The username of staff work in this shift
StoreId	int			x		The id of store in this shift
SkillId	int			x		The id of skill in this shift
TimeStart	datetime2			x		The time when the shift start
TimeEnd	datetime2			x		The time when the shift end
Status	int			x		The status of this ShiftScheduleDetailResult

Table 99 <Physical Diagram> ShiftScheduleDetailResults

4.1.12 StoreScheduleDetails Table

Field name	Type	Size	Unique	Not Null	PK/FK	Notes
Id	int		x	x	PK	The id of this StoreScheduleDetail
WeekScheduleId	int			x	FK	The id of WeekSchedule related to this WeekSchedule
MinDayOff	int			x		The min day off in week
MaxDayOff	int			x		The max day off in week
MinHoursPerWeek	float			x		The min hours staff need to work in this week
MaxHoursPerWeek	float			x		The max hours staff can work in this week
MinHoursPerDay	float			x		The min hours staff need to work in one day
MaxHoursPerDay	float			x		The max hours staff can work in one day
MinShiftDuration	float			x		The min length of one shift
MaxShiftDuration	float			x		The max length of one shift
MaxShiftPerDay	int			x		The max number of shift staff can work in one day
StaffType	int			x		The type of staff applied these constraints: 0 - full time, 1 - part time

Status	int			x		The status of this StoreScheduleDetail

Table 100 <Physical Diagram> StoreScheduleDetails

4.1.13 WeekScheduleDetails Table

Field name	Type	Size	Unique	Not Null	PK/FK	Notes
Id	int		x	x	PK	The id of this WeekScheduleDetail
WeekScheduleId	int			x	FK	The id of WeekSchedule
SkillId	int			x		The id of skill need for this time frame
Level	int			x		The level of skill need for this timeframe
Quantity	int			x		The quantity of staff need for this timeframe
WorkStart	datetime2			x		The time start the shift
WorkEnd	datetime2			x		The time end the shift
Status	int			x		The status of this WeekScheduleDetail

Table 101 <Physical Diagram> WeekScheduleDetails

4.1.14 Attendances Table

Field name	Type	Size	Unique	Not Null	PK/FK	Notes
Id	int		x	x	PK	The id of this attendance
StoreId	int			x	FK	The id of the store where staff took attendance
Username	varchar	50		x	FK	The username of the staff who took attendance
TimeCheck	datetime2			x		The time when staff took attendance
CreatedBy	varchar	50		x		The Attendance created by
Note	varchar	200				The note for this attendance
CheckType	int			x		The method staff use for take attendance: 0 - face recognize, 1 - QR Code, 2 - finger print, 3 - manual
ImageUrl	varchar	MAX				The link to the image recorded when the staff took attendance
RecognizePercentage	float					The Recognize Percentage compare to the base model

DeviceCode	varchar	100				The Device Code which use to took this attendance
Status	int			x		The status of this attendance

Table 102 <Physical Diagram> Attendances

4.1.15 ShiftRegister Table

Field name	Type	Size	Unique	Not Null	PK/FK	Notes
Id	int		x	x	PK	The id of this ShiftRegister
WeekScheduleId	int			x	FK	The id of the WeekScheduleId this shift register related to
Username	varchar	50		x	FK	The username of the staff who register this shift
TimeStart	datetime2			x		The time when staff available start
TimeEnd	datetime2			x		The time when staff available end
Status	int			x		The status of this ShiftRegister

Table 103 <Physical Diagram> ShiftRegister

5. Algorithm

5.1 Scheduling algorithm

- **Abstract:** This research introduces an integrated **Employee scheduling** problem that considers various real-life problems such as varying **employee demand**, different **employee working conditions**. The **proposed model** is a **constraint programming** (CP) model used to solve the problem with varied constraints. Results show that the proposed model generates **optimal and feasible solutions** for weekly employee schedules that satisfies certain **hard constraints** and minimizes an objective function defined as weighted sum of **soft constraint** violations with a single goal of **reducing understaffing**.
- **Limit**: Because time is limited and it takes time to develop and exploit functions to be able to combine algorithms and systems. These studies still do not fully reflect current realistic organizational problems such as **individual preferences**, **breaking time**, **overstaffing**. We focus on **basic constraints** to construct a basic schedule to meet basic requirements. We consulted many coffee shops about overtime of staff and they replied that no employees are working more than 8 hours/day so we will **ignore optimizing cost for overtime**.

- **Keywords:** Constraint programming, employee scheduling, individual preferences, breaking time, understaffing, overstaffing

5.1.1 Introduction

Employee scheduling is an important function performed by managers. Today, managers face two difficulties in scheduling:

- The number of customers varies between different days (such as weekdays and weekends and holidays); and times of the day (such as normal and peak hours). (1)

The difficulty of finding enough full-time employees to cover all staffing needs is solved by employing part-time employees. But it's also not easy because requiring part-time employees to be available during specific periods of the operating day is limited time-availability. For example “work shift starts from 10 am to 5 pm, but you are free from 12 pm to 6 pm, so we can't hire you even though we need more staff between 12 am and 2 pm which is the peak time after lunch." (2)

To be able to solve the above problem and increase flexibility so that part-time employees can conveniently register their free time and take advantage of smaller intervals to be able to increase their working time more. We will not divide a day into 2 or 3 shifts as usual, but into small 30-minute intervals that include approximately 48 periods. Shifts will consist of these small intervals continuously. This employee's shift may not be the same as another employee's depending on many factors. This makes it easier and more flexible to plan staffing needs and register employees' free time.

In addition, increase flexibility and clarity of planning or pre-scheduling corresponding to personnel needs and employee classification. Therefore, we include factors such as skills corresponding to different working positions combined with the level representing the level of work experience of the employees to meet the demand more clearly.

5.1.2 Approach

We define the employee scheduling problem as a constraint problem. Constraint Programming (CP) is a relatively modern technology for solving constraint satisfaction and constraint optimization problems. Constraint programming is the technique that is used to model the problem and Google OR- tools is the library used for the implementation.

Note: CP is a declarative form of programming for optimization problems, i.e., instead of coding how things should be done, all that is necessary is to declare what should be calculated and then a general algorithm attempts to solve the problem.

This is the process of implementing CP to solve the problem:

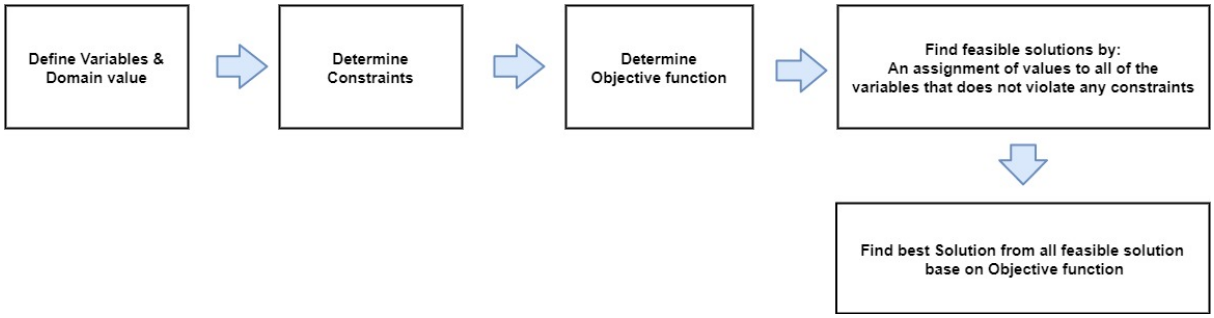


Figure 76 The process of implementing CP to solve the problem

5.1.3 Constraint Programming model

Notation

The indices, parameters, decision variables, and dependent variables and their descriptions that will be used to model the problem are listed in Tables 1 to 4, respectively:

Indices	Description
E	`Set of employees e` $\in E$ (e = 0, 1, 2, …, n-1) with n = total employee
D	Set of days $d \in D$ (d = 1, 2, …, 7)
T	Set of time periods $t \in T$ (t = 0, 1, 2, …, 47)
S	Set of skill $s \in S$ (Ex: s = 0:Bartender, 1:Cashier, 2:Waiter)
L	Set of level l $\in L$ (l= 0, 1, 2)

Table 1. Indices

Parameters	Description
minWorkingTimeOnWeek	Minimum working time of 1 employee per week
maxWorkingTimeOnWeek	Maximum working time of 1 employee per week
maxWorkingTimePerDay	Maximum working time of 1 employee per day

minShiftDuration	Minimum duration of one shift
maxShiftDuration	Maximum duration of one shift
maxShiftPerday	Maximum number of shifts of 1 employee in 1 day
minDayOff	Minimum number of day-off of 1 employee in 1 week
maxDayOff	Maximum number of day-off of 1 employee in 1 week
R_{dslt}	Number of employees needed at day *d* at time period *t* for skill *s* with higher level *l*
SE_{es}	Skill which an employee can works at. 1 if employee e can work at skill *s*; 0 otherwise
V_{edt}	Available time of employees. 1 if employee e is available on day d time t; 0 otherwise
P_B	Penalty weight for under-coverage
P_C	Penalty weight for over-coverage

Table 2. Parameters

Decision variables	Description
W_{edkt}	1 if employee e is assigned to work on day d at time t at skill k; 0 otherwise

Table 3. Decision Variables

Dependent variables	Description
G_{ed}	1 if employee *e* working on day *d;* 0 otherwise
A_{dslt}	Number of employees working at day *d* at time period *t* for skill s with level *l*

B_{dslt}	Number of employees is under-covered at day *d* at time period *t* for skill s with level *l*
C_{dslt}	Number of employees is over-covered at day *d* at time period *t* for skill s with level *l*

Table 4. Dependent Variables

Step 1: Define Variables and Domain:

First, we create a set of variables to represent the schedule in the form of matrix 4x4.

$$W_{edst} = \begin{cases} 1, & \text{when staff e works on day d at time t in skill s} \\ 0, & \text{otherwise} \end{cases}$$

Step 2: Determine constraints:

#1 Constraint for total working time per week: Total working time of each employee must be in range:

$$minWorkingTimePerWeek \leq \sum_{d \in D} \sum_{s \in S} \sum_{t \in T} W_{edst} \leq maxWorkingTimePerWeek$$

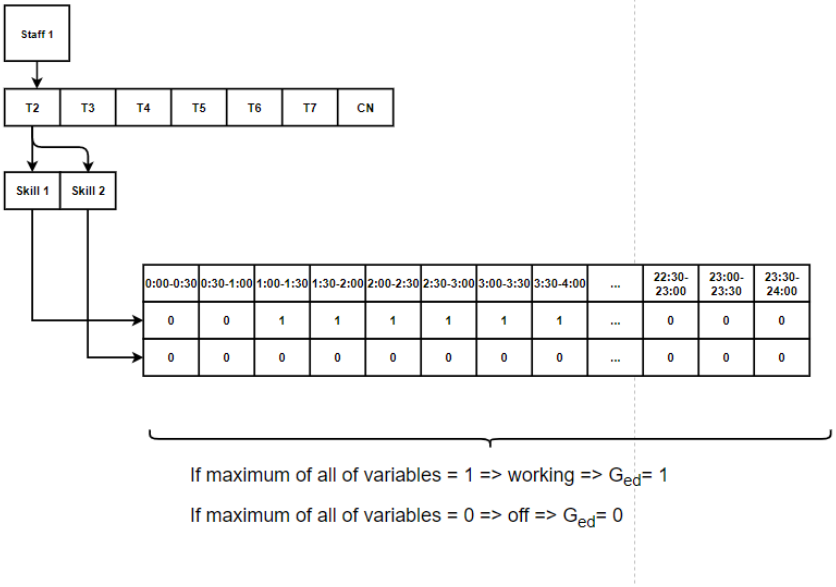
#2 Constraint for total working time per day: Total working time of each employee must be less than *maxWorkingTimePerDay:*

$$\sum_{s \in S} \sum_{t \in T} W_{edst} \leq maxWorkingTimePerDay$$

#3 Constraint for day-off per week: Total day-off of each employee must be in range (minDayOff, maxDayOff):

$$minDayOff \leq \sum_{d \in D} G_{ed} \leq maxDayOff$$

Value of G_{ed} will be defined by:



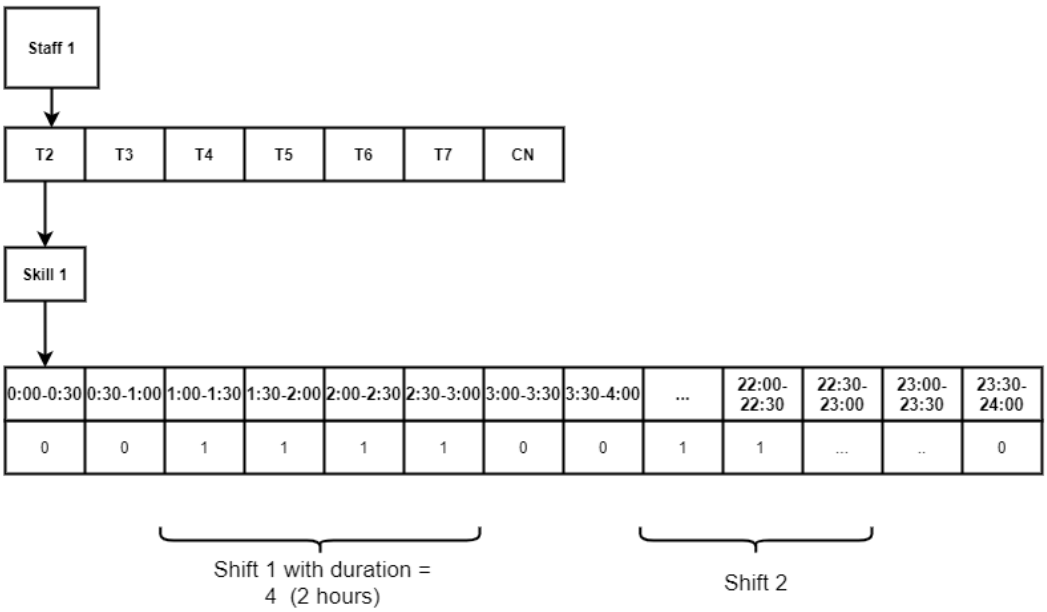
#4 Constraints for length and number of shift:
For each shift:

$$minShiftDuration \leqslant shiftDuration \leqslant maxShiftDuration$$

$$totalShiftPerday \leqslant maxShiftPerday$$

shiftDuration can be defined by counting consecutive time periods.

totalShiftPerday can be defined by count all shifts in one day;



#5 Constraint for employee working in available time: employees ***e*** only can work on available time ***V*** they were registered at day ***d*** time period ***t***.

$$W_{edst} \leq V_{edt}, \forall s \in S$$

#6 Constraints on employee skills: employees ***e*** can only work at skill ***s*** that they are pre-registered in ***SE***.

$$W_{edst} \leq SE_{es}, \forall d \in D, \forall t \in T$$

#7 Employee ***e*** only work at 1 skill at a time:

$$\sum_{s \in S} W_{edst} \leq 1, \forall d \in D, \forall t \in T$$

#8 Constraint for demand:Total employees working should meet employee demand at day ***d*** at each time period ***t*** in skill **s** and level ***l***, allowing for over-coverage (C_{dslt}) and under-coverage (B_{dslt}).

$$C_{ds(l+1)t} + A_{dslt} + B_{dslt} - C_{dslt} = R_{dslt}$$

Where

$C_{ds(l+1)t} = 0$ *with* $l = 3$: *Total employees over − coverage at level higher* current level

Step 3: The Objective Function:

The problem will be solved by minimizing the total of objective functions subject to the following constraints:

$$\text{Min } z = P_B\left(\sum_{d\in D}\sum_{t\in T}\sum_{s\in S}\sum_{l\in L} B_{dslt}\right) + P_C\left(\sum_{d\in D}\sum_{t\in T}\sum_{s\in S} C_{ds(l=0)t}\right)$$

Minimize number of under-coverage staff and Minimize number of over-coverage at level 0.

Step 4 & Step 5:

We use the CP-SAT Solver of Google OR-Tools to implement these two steps.

5.1.4 Limitations

Solving the problem is thus done through an interactive iteration between the user and the scheduler. The algorithm can only respond well in cases where the store's resources can correspond to the demand. In the case of a shortage of too many resources, there may appear periods when there are no employees working.

5.2 Face Recognition

5.2.1 Definition

To Deal with attendance problems, we figured out that there are many methods, but fingerprint recognition and face recognition are the most specific to verify the staff's identity. Since fingerprint recognition has some problems, such as staff having to clean their finger, and also cleaning the sensor before using. Moreover, staff risk spreading unwanted bacteria around, which doesn't seem pleasing at all. So we use face recognition instead.

5.2.2 Define Problem

There are many ways to approach with face recognition:

- Use an IP camera: only get video streams.
- Use a camera device: can verify before face recognition.
- Take attendance one by one: suitable for timekeeping, conference, meeting.
- Take attendance many attendees at once: suitable for schools.

5.2.3 Solution

Our scope is timekeeping, so we choose to take attendance one by one, and also choose using a camera device because if we can verify before face recognition, the performance will be increased.

In the process of researching, we found out Face Recognition package, which is based on Python programming language. This package is built using Dlib's state-of-the-art face recognition and built with deep learning.

Our face recognition process:

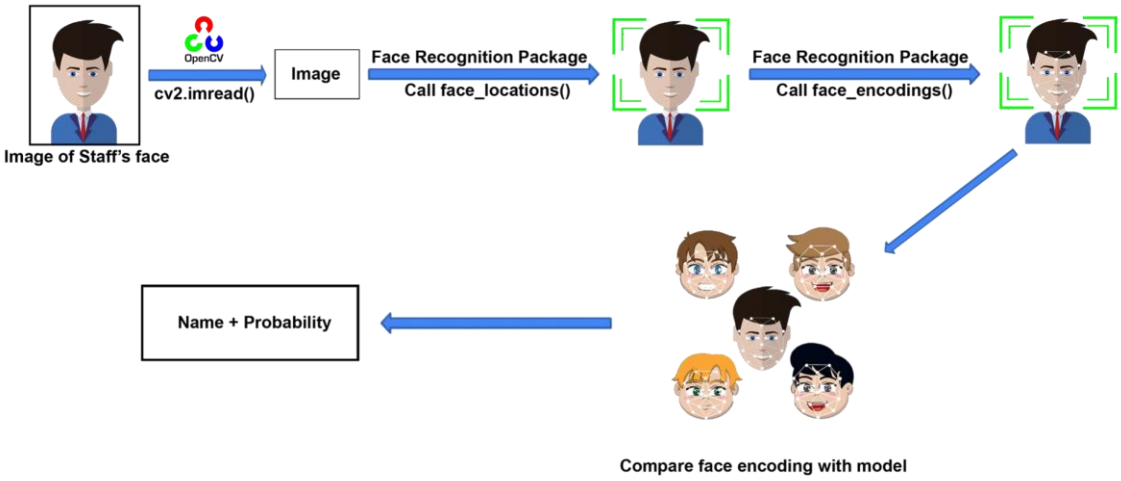


Figure 77 Face Recognition Process

- Input an image of Staff's face.
- Read image for processing.
- Detect face: we call face_locations from Face Recognition Package to find face's location.
- Encode face: we call face_encodings from Face Recognition Package to encode face to 128 dimensional vector.
- Compare face encodings: After encoding face, we have a 128 dimensional vector. Then we load pretrained model.
- After comparing face with model, we will get the name of the staff and probability.

Step 1: Input an image of Staff's face

We use the Store Timekeeping application to take a photo of staff's face. After staff clicks the button to take a photo, this image will be sent to Face recognition server. Assumption that there is only 1 face in this image.

Step 2: Read image for processing

When the server receives an image of Staff's face, this image will be saved to storage. Then we use OpenCV package to read image for processing.

Reference to OpenCV: https://opencv.org

Step 3: Detect face

We call face_locations from the Face Recognition package to find face's location. This method uses a method invented in 2005 called Histogram of Oriented Gradients - or just HOG for short.

Reference to HOG: http://lear.inrialpes.fr/people/triggs/pubs/Dalal-cvpr05.pdf

How this method works:

- The original image is turned into a HOG representation that captures the major features of the image regardless of image brightness.
- With HOG image, find the part of the image that looks the most similar to a known HOG pattern that was extracted from a bunch of other training faces, then we have the location of the face in the image.

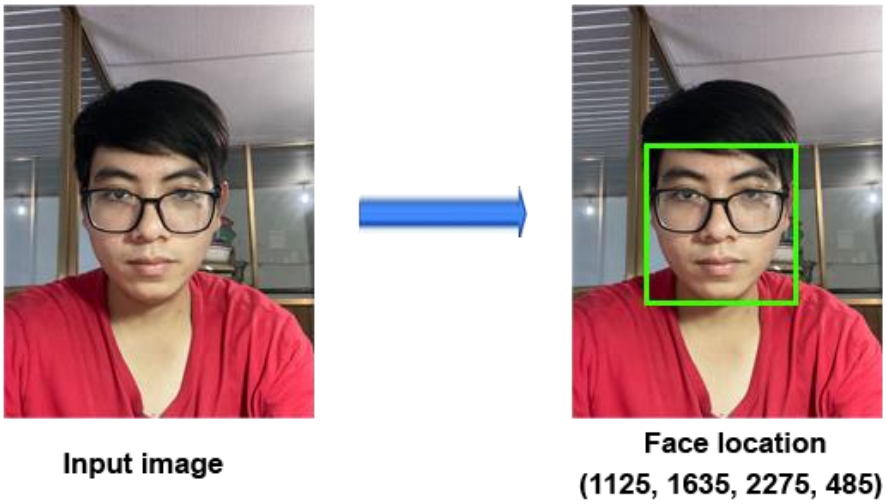


Figure 78 <Face detection> Detect face

Step 4: Encode face

From step 3 we have the location of the face. Then we call face_encodings from the Face Recognition package. This method uses a pre-trained network to get the 128 measurements for the face.

Face location

[-1.00380503e-01, 8.55414867e-02, 2.33170688e-02, -3.52578349e-02,
-6.31213784e-02, -7.19021410e-02, -5.03488891e-02, -1.44731000e-01,
1.62536919e-01, -1.07897148e-01, 2.35749751e-01, -3.98353785e-02,
-1.67808205e-01, -8.10519308e-02, -2.36569941e-02, 1.13952920e-01,
-2.44163185e-01, -4.10131961e-02, 1.26227392e-02, 6.04514033e-03,
1.19786739e-01, 1.19727990e-02, 7.83408433e-02, 8.12924504e-02,
-9.75858867e-02, -3.62402052e-01, -9.43632424e-02, -1.15093634e-01,
2.45414879e-02, -4.83753979e-02, -1.54015794e-02, -1.84272975e-03,
-1.28621370e-01, -1.75544471e-02, 5.56711853e-03, 1.90051310e-02,
-4.00691107e-02, -4.58130464e-02, 2.05937222e-01, -6.71697631e-02,
-2.38199174e-01, 1.18863732e-02, 4.63477075e-02, 2.05398500e-01,
1.93472892e-01, 1.07629679e-01, -1.73629671e-02, -1.28463224e-01,
1.60113066e-01, -1.99606493e-01, 1.60524677e-02, 1.37756303e-01,
3.95094138e-03, -6.44154847e-03, -1.42730922e-02, -1.59120113e-01,
2.21035480e-02, 7.29974657e-02, -1.78411096e-01, 3.02061103e-02,
8.21049586e-02, -1.21583775e-01, 4.66142632e-02, 2.46732011e-02,
2.15501457e-01, 7.91719705e-02, -1.25020698e-01, -1.28281802e-01,
8.57459977e-02, -2.08457977e-01, -3.21398675e-02, 7.09578395e-02,
-1.41071573e-01, -2.48251855e-01, -2.81718165e-01, -6.43317401e-03,
4.26202953e-01, 1.21069252e-01, -9.09870267e-02, 1.68684684e-02,
-8.44466090e-02, 3.26819718e-05, 1.66234672e-01, 1.57107651e-01,
-3.64071727e-02, -6.46005645e-02, -1.41722530e-01, 3.94119322e-03,
2.24435091e-01, -6.63642362e-02, -5.05987406e-02, 2.52168030e-01,
-4.25083935e-03, 1.17837556e-01, -2.65950430e-02, 1.94235444e-02,
-1.15525588e-01, 6.72915280e-02, -1.06007539e-01, 2.86284983e-02,
2.26102471e-02, 1.40252896e-03, 2.85947174e-02, 1.42264009e-01,
-1.76678747e-01, 1.19346812e-01, -6.77741319e-03, -7.72401690e-04,
3.67728919e-02, 1.21966004e-04, -1.12261884e-01, -8.34244788e-02,
9.61202234e-02, -1.93191051e-01, 2.18047291e-01, 1.81692511e-01,
1.55341744e-01, 1.79579496e-01, 8.67729634e-02, 3.64308581e-02,
8.59810412e-03, -2.15140730e-02, -2.08516151e-01, 2.50607505e-02,
7.30481893e-02, -2.73706913e-02, 1.13414004e-01, -2.68799067e-02]

128 Measurements Generated from Image

Figure 79 Face encoding

These 128 numbers are nearly the same when generated from two different images of the same person.

Step 5: Compare face encoding with model

From step 4, we have 128 measurements of face in image. This last step is actually the easiest step in the whole process. All we have to do is find the person in our database of known people who has the closest measurements to our test image. After that, we will get the name and probability of the face in the image.

These steps base on Adam Geitgey article: https://medium.com/@ageitgey/machine-learning-is-fun-part-4-modern-face-recognition-with-deep-learning-c3cffc121d78

mystaff04, 96.21%

Figure 80 Face Recognition result

Training model

To train model, we use scikit-learn's implementation of Support Vector Machines (SVM), a common machine learning model.

Reference to scikit-learn's implementation of SVM: https://scikit-learn.org/stable/modules/svm.html

```
trungshin@Trungs-MacBook-Pro face_recog_api %  cd /Users/trungshin/Desktop/project/face_recog_api ;
ode/extensions/ms-python.python-2021.8.1105858891/pythonFiles/lib/python/debugpy/launcher 50482 -- /
l.py
[INFO] loading face embeddings...
[INFO] 470 encodings...
[INFO] training model...
[INFO] Accuracy: 100.00%
[INFO] training model successful...
```

Figure 81 Training result

Our processing:

- We use Store Timekeeping application to get video and username as label. Store manager must check that there is only one face in this video and username of staff is correct.
- Read video and detect face after every 4 frames, then align face and export image of face.
- All images of face exported from video are saved to the same folder with username as name.
- Load embeddings file or create new if this file does not exist. Encode all image of face from previous step, then add to embeddings file with username as label.
- Use LabelEncoder from Scikit-learn to encode labels from embeddings file.
- Use Linear SVM classification to train the model.
- After training the model we output the model and label encoder to disk as pickle files.

5.2.4 Limitations

Total time for face recognition is still long, about 4 - 8 seconds from taking a photo to returning the result. And there are many problems that may affect the result of face recognition.

V. Software Testing Documentation

1. Overall Description

1.1 Test Model

In STS project, we apply Agile testing because:

- One person can do many things, for example dev can test.
- Detect errors early.
- Capable of being applied to projects where customer requirements are not clear at the outset.

1.2 Testing Levels

Below are some testing levels that is applied in STS project:

- Unit test: test each small module in the system, for example scheduling, attendance, registration for working days, ...
- Integration test: individual software modules or functions that are logically integrated and tested in groups together.
- System test: test user interface for system.

2. Test Plan

2.1 Test Stages

Type of Test	**Stage of Test**			
	Unit	***Integration***	***System***	***Acceptance***
Function Test	X	X		
User Interface test			X	X

2.2 Resources

a. Human Resources

Worker/Doer	**Role**	**Specific Responsibilities/Comments**
Phạm Minh Đạo	Leader	Planning, verifying test deliverables, review
Vũ Thị Ngọc Mai	Member	Planning, do unit tests follow the plan.
Lý Văn Cường	Member	Do unit tests follow the plan.
Đỗ Quốc Trung	Member	Do unit tests follow the plan.

2.3 Test Milestones

Milestone Task	**Effort (md)**	**Start Date**	**End Date**

Test schedule module	2	20/7/2021	5/8/2021
Test timekeeping module	2	20/7/2021	5/7/2021

3. Test Cases

Detail test cases will be described detail in: Report5_Test Case Document.xlsx

4. Test Reports

Detail test reports will be described detail in: Report5_Test Case Document.xlsx

VI. Release Package & User Guides

1. Deliverable Package

1.1 Source codes & documents

No.	Items	Sub-Items	Type	Version
Code Package				
1	STS.Restful-API	Web service	New	1.0.0
2	STS.Algorithm	Web service	New	1.0.0
3	STS.FaceRecognize	Web service	New	1.0.0
4	STS.Admin	Admin Web Application	New	1.0.0
5	STS.Manager	Brand & Store Manager Web Application	New	1.0.0
6	STS.StaffMobile	Staff Mobile Application	New	1.0.0
7	STS.StoreDevice	Store Face Recognize Mobile Application	New	1.0.0
Documents				
1	Final Project Report	Final_report_STS.doc	New	1.0
2	Testing Document	Report_3_Test_casesSTS.doc	New	1.0

2. Installation Guides

2.1 System Requirements

Server	Minimum Requirements	Recommended

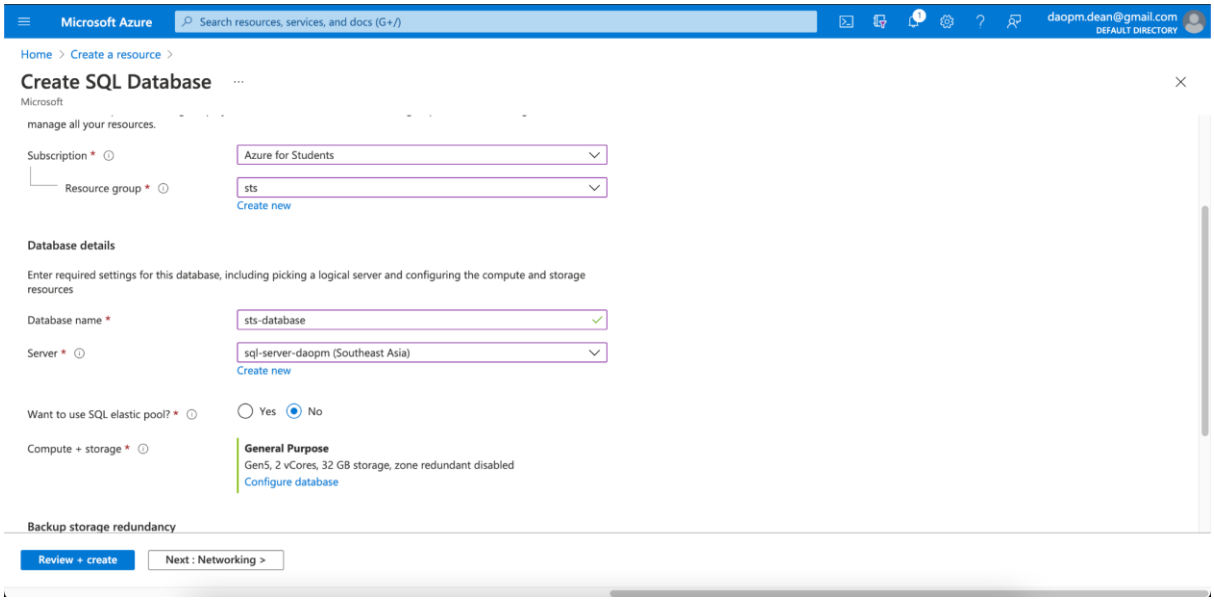
Operating System	XP, Window 7, Window 10, Ubuntu	Window 10, Ubuntu
Computer Processor	1 Core CPU	2 Core CPU or more
Computer Memory	2Gb RAM	4Gb RAM or more
Storage Space	10Gb	20Gb or more

2.3 Installation Instruction

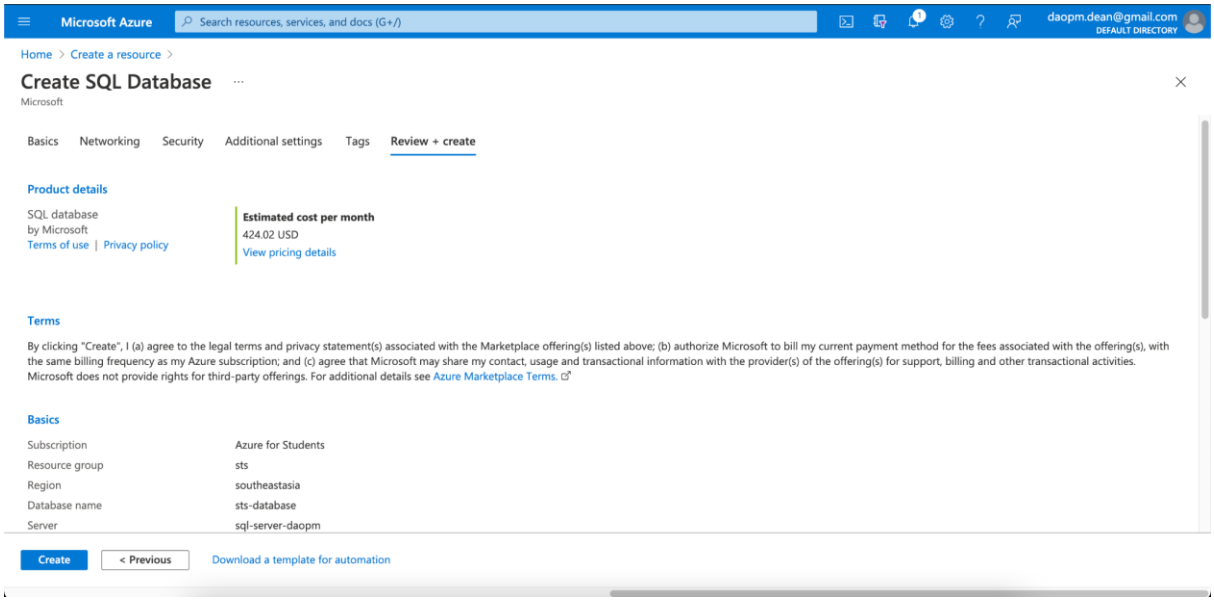
2.3.1 Deploy Database - SQL Server

This setup used Microsoft Azure cloud service for deployment.

Step 1: Fill in the configuration information you want to set up for your database. After you finish, click “Review & Create”.



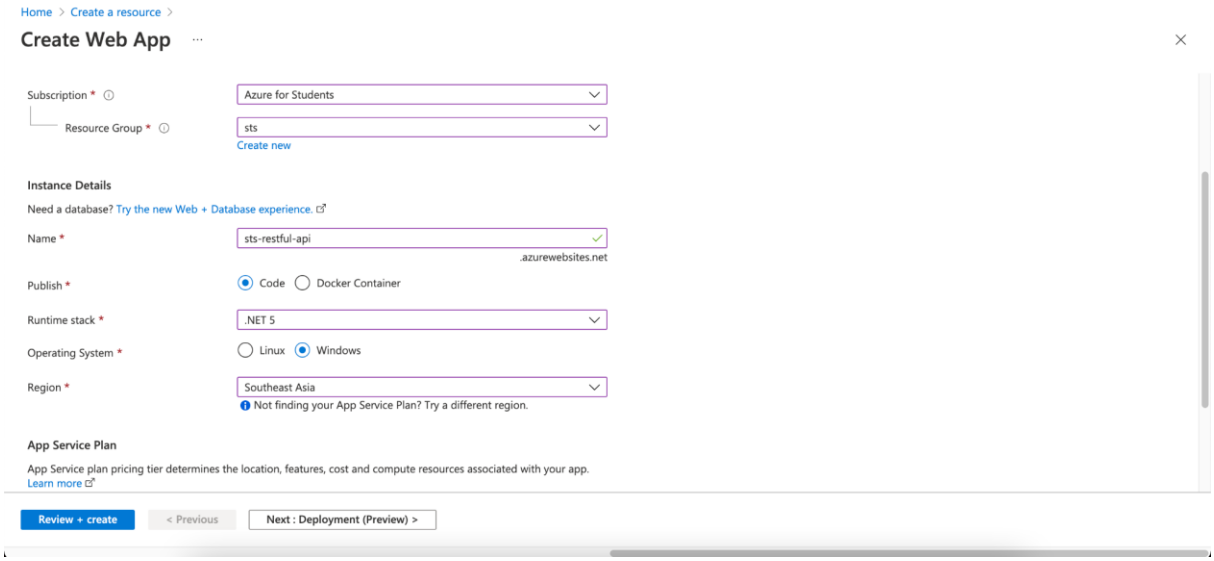
Step 2: Check again the information for confirmation, click “Create” To create your database.



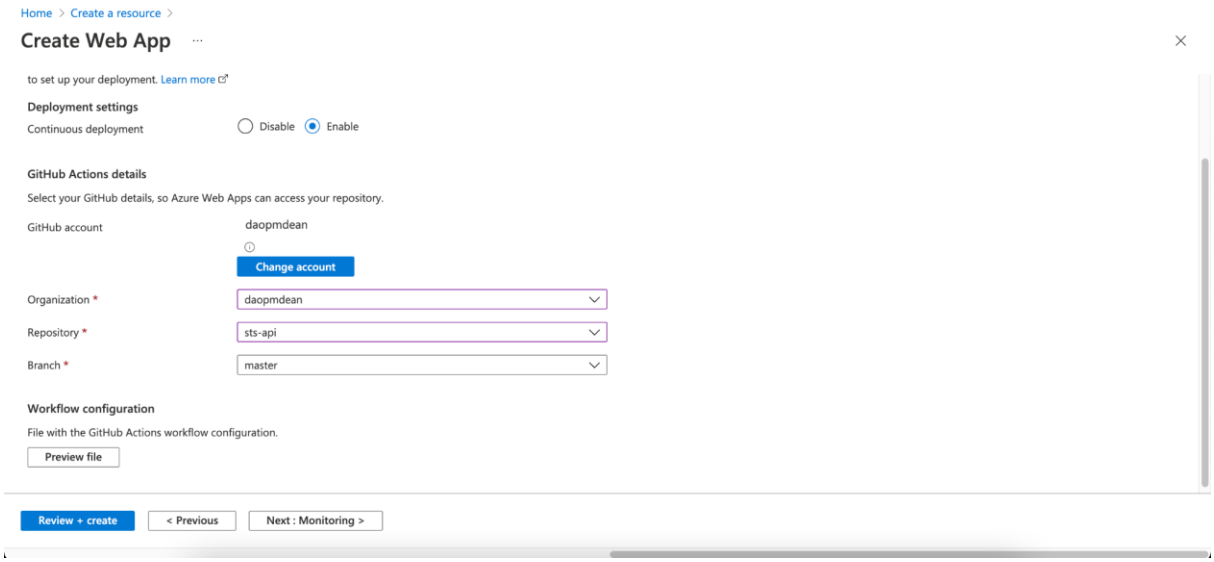
2.3.2 Deploy Server - ASP.Net Core

This setup used Microsoft Azure cloud service for deployment.

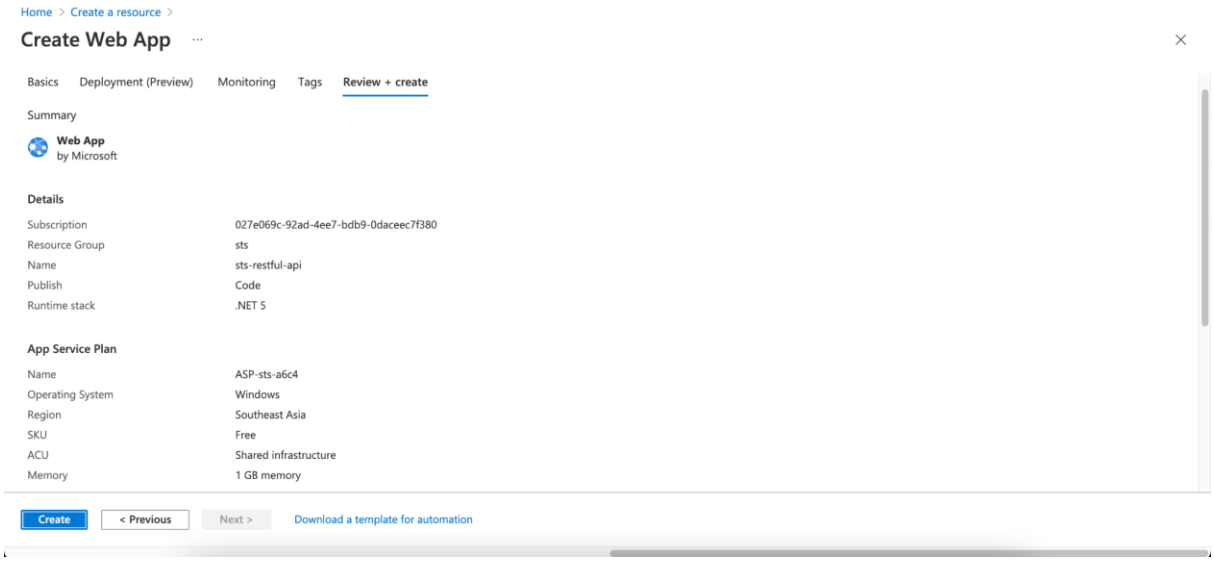
Step 1: Fill in the information for your project deployment. After finishing, click "Next: Deployment" to configure continuous deployment for your project.



Step 2: Enable continuous deployment, authenticated to your github account then choose appropriate repository & brand for your deployment. After finishing, click "Review & Create".



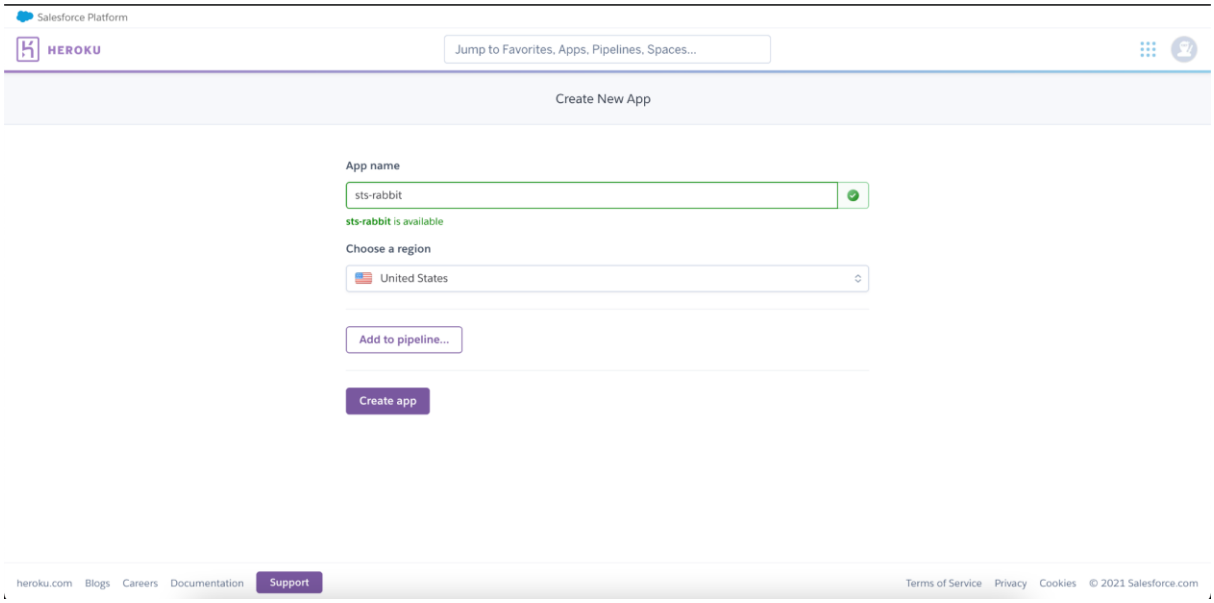
Step 3: review your installation and click “Create”



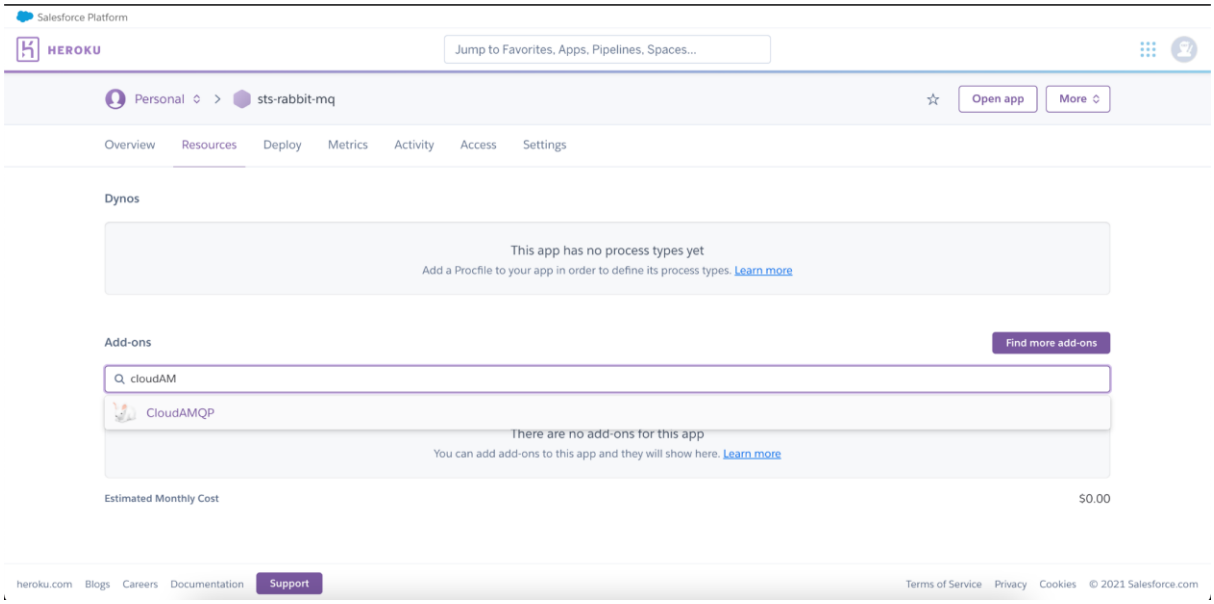
2.3.3 Deploy RabbitMQ - CloudAMQP

This setup used Heroku Add-on “CloudAMQP” for deployment.

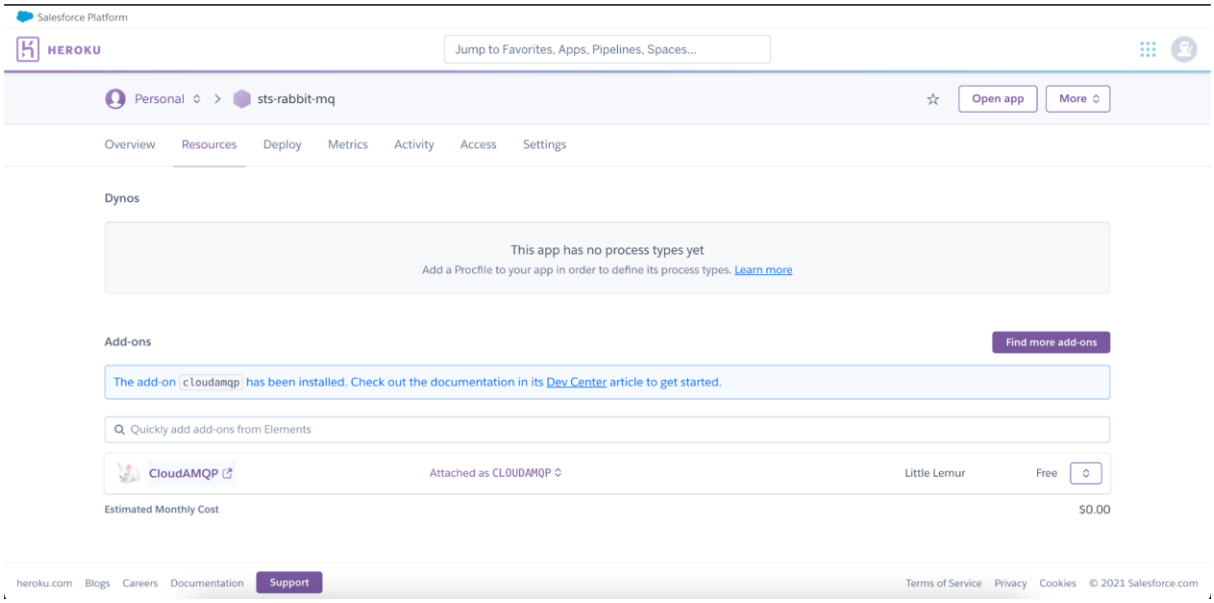
Step 1: Create your heroku project.



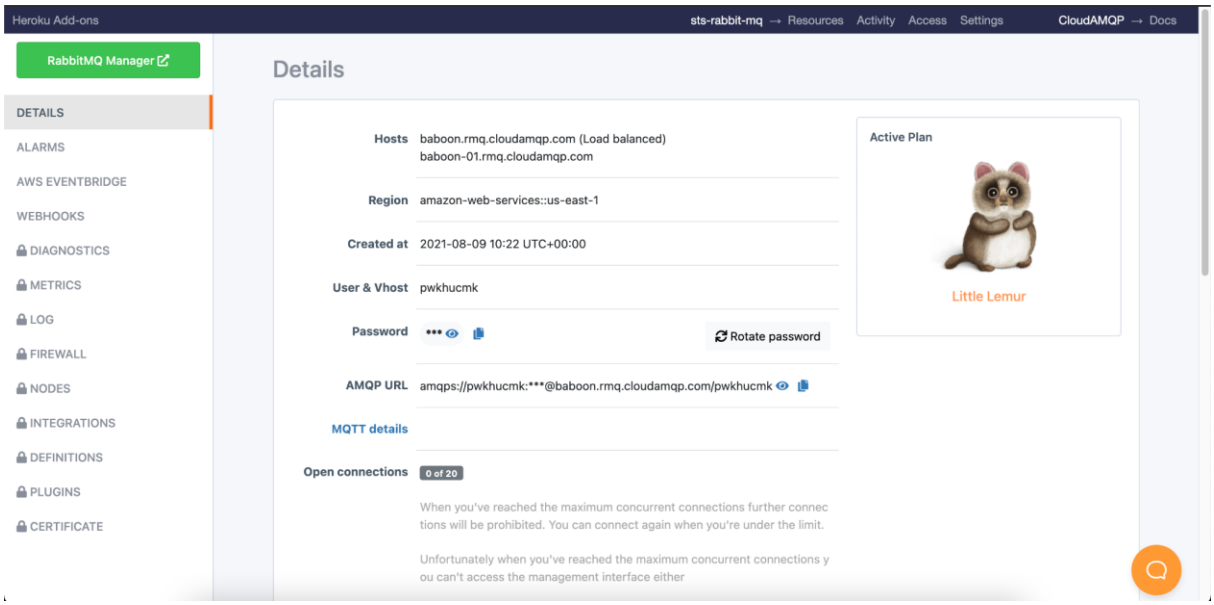
Step 2: Search for CloudAMQP Add-ons in the Resources tab.

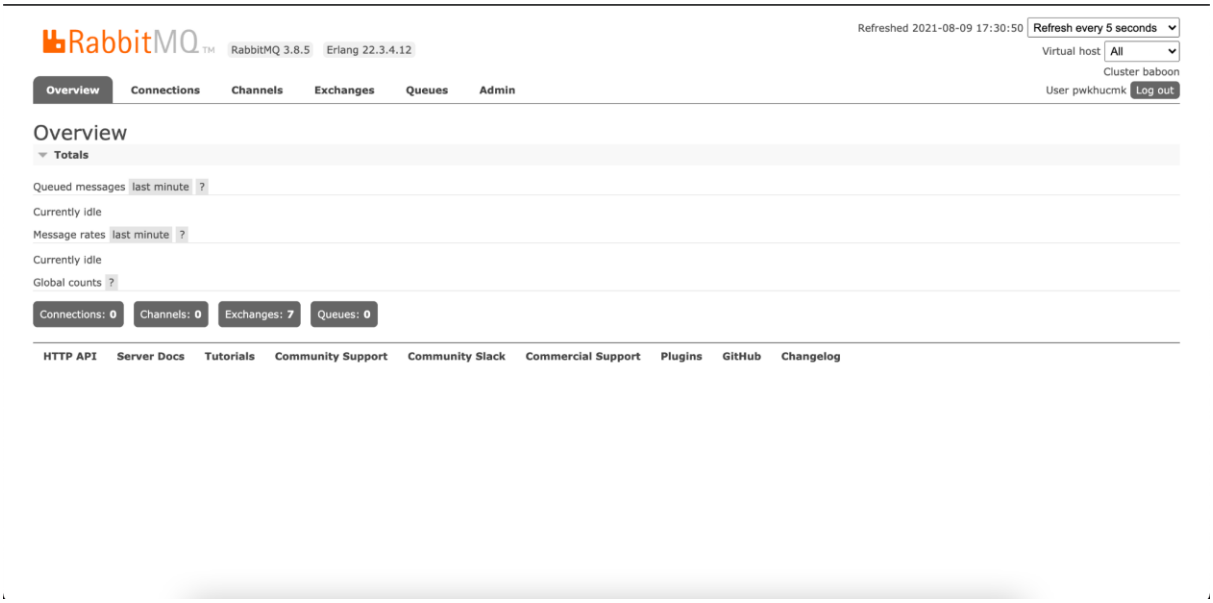


Step 3: Follow heroku instructions to complete your creation. After finishing, you can use your Add-ons service.



Step 4: Now you can use your credentials provided by heroku to use RabbitMQ.

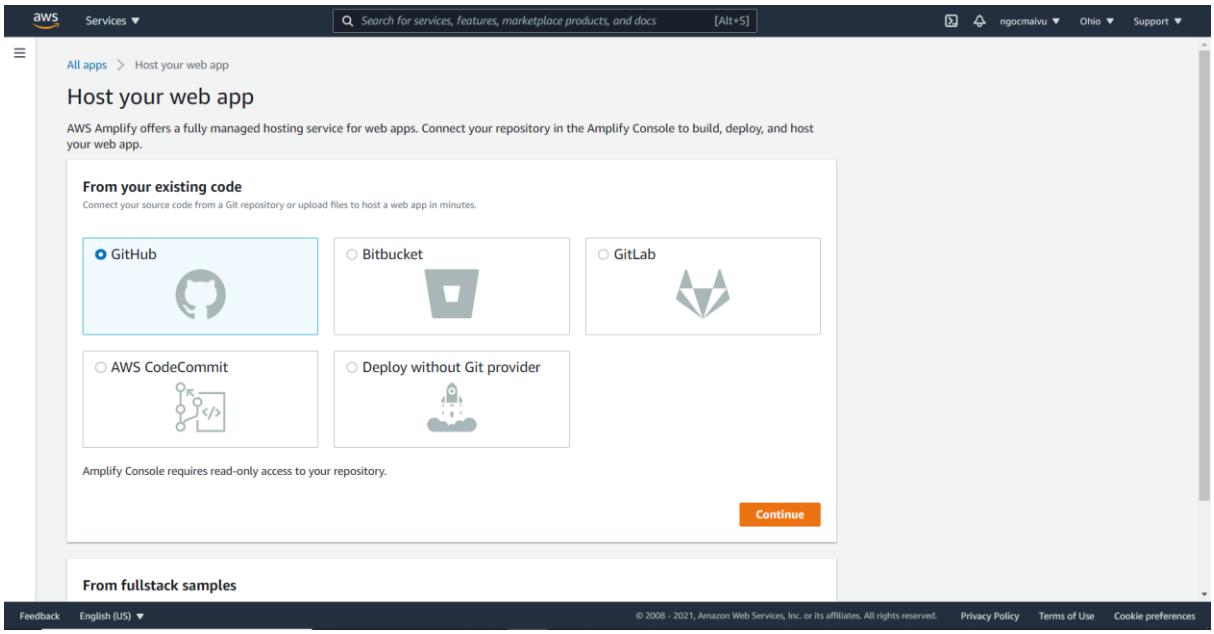




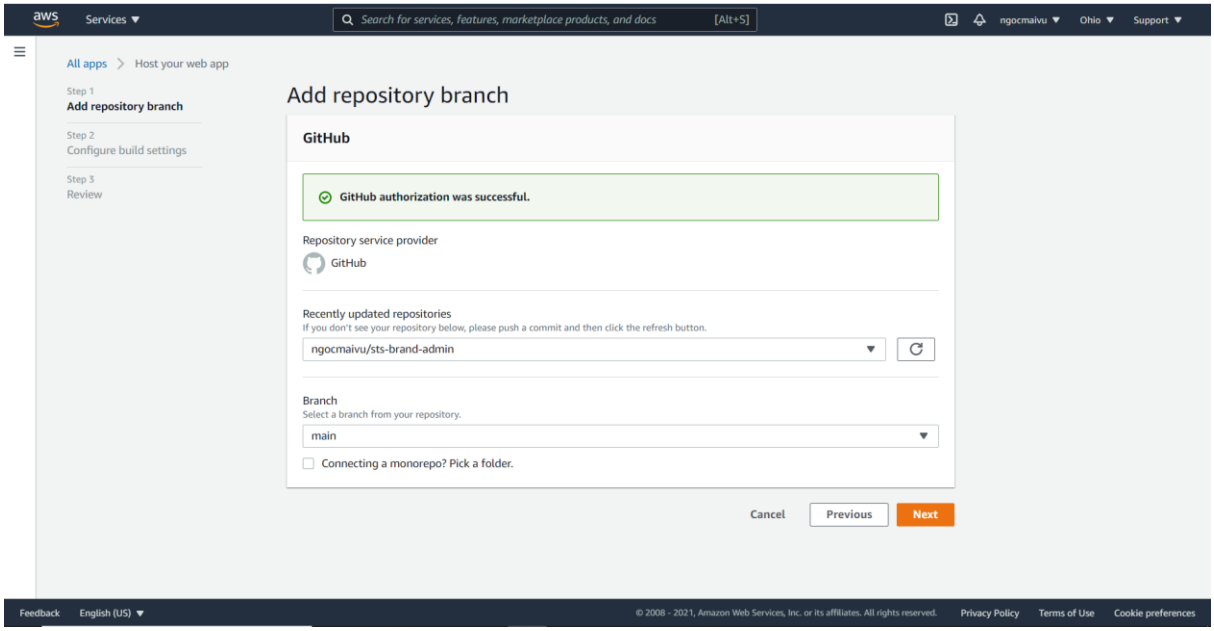
2.3.4 Deploy Web Application in Amazon Web Services (AWS)

This setup used AWS amplify - Global Cloud Service Provider for deployment.

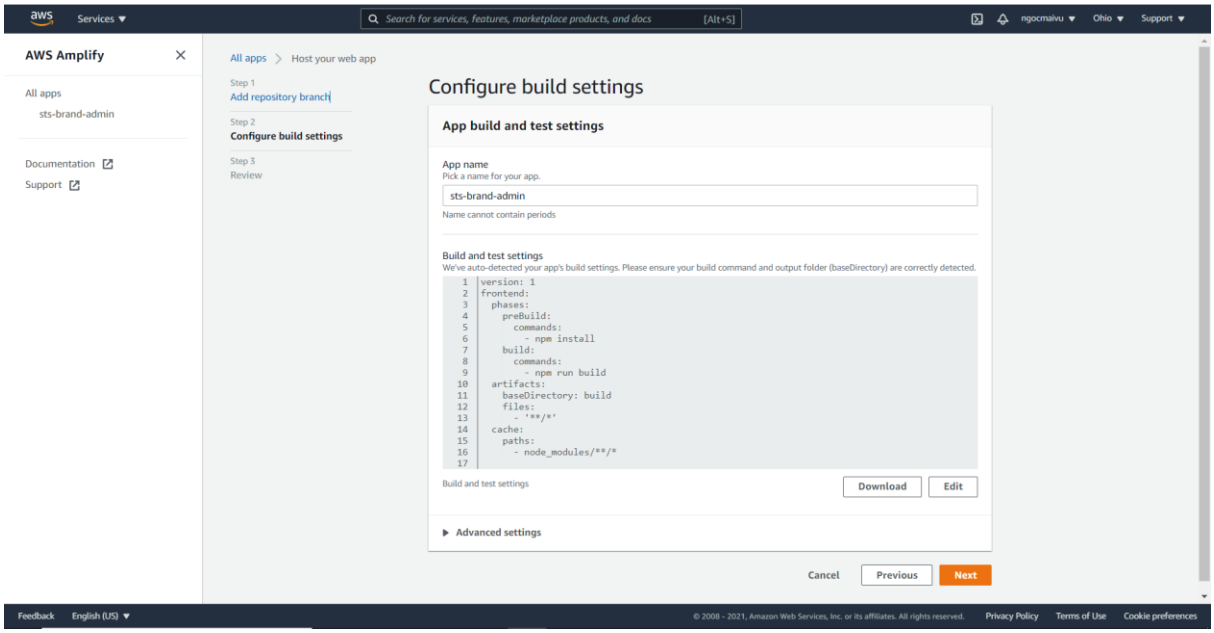
- **Step 1:** Create your project by connecting your source code from a Git repository. Click “Continue”



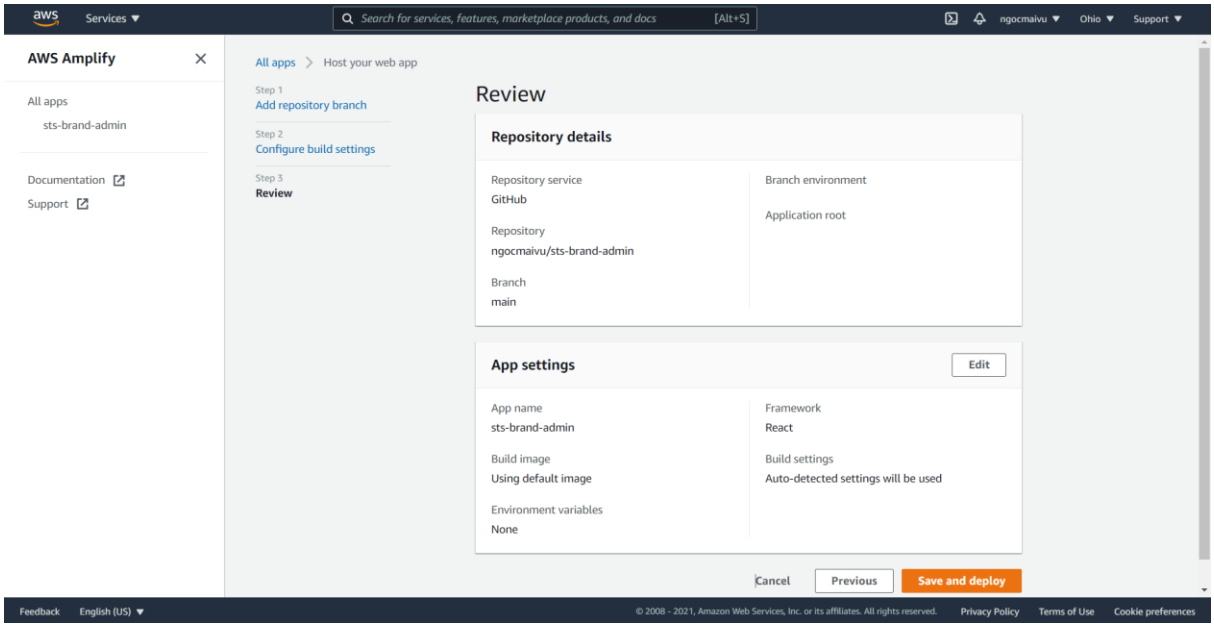
- **Step 2:** Add a repository branch after successful Git account authentication. Choose branch and click “Next”.



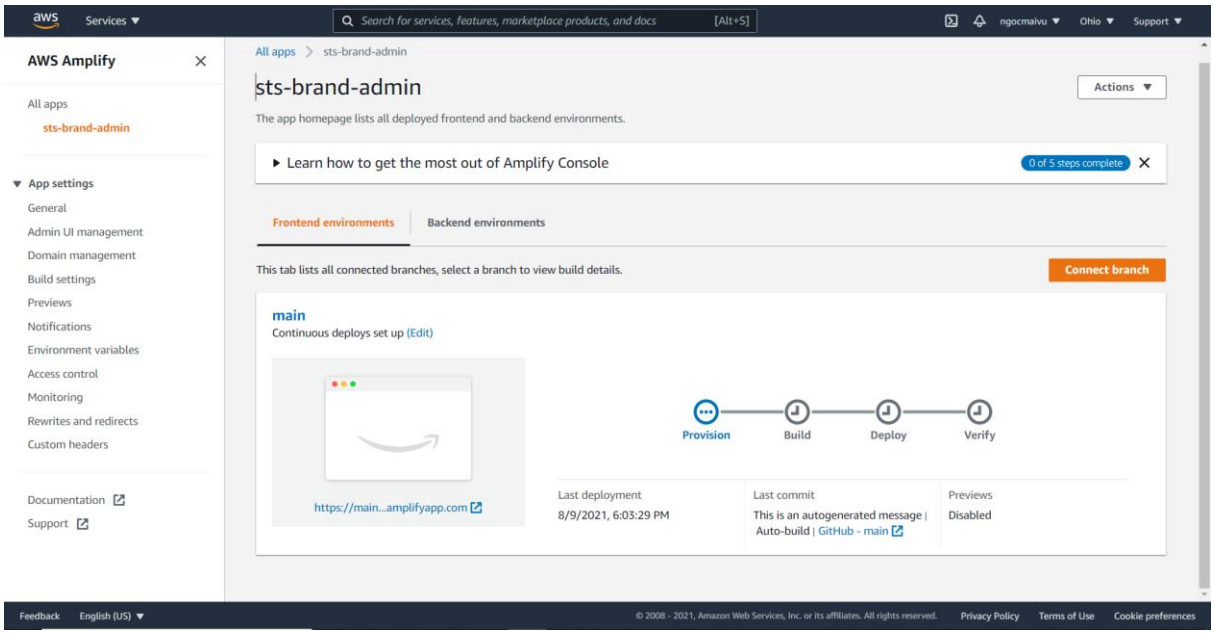
- **Step 3:** Configure build settings and click “Next”.



- **Step 4:** Review and click “Save and deploy”



- **Step 5:** Now we can manage lists of all deployed frontend environments, track the process deploy of web apps. Any commit from Git will be deployed automatically.



2.3.5 Deploy Face Recognize Server (IIS)

This setup used IIS (Internet Information Services) for Python Flask application deployment.

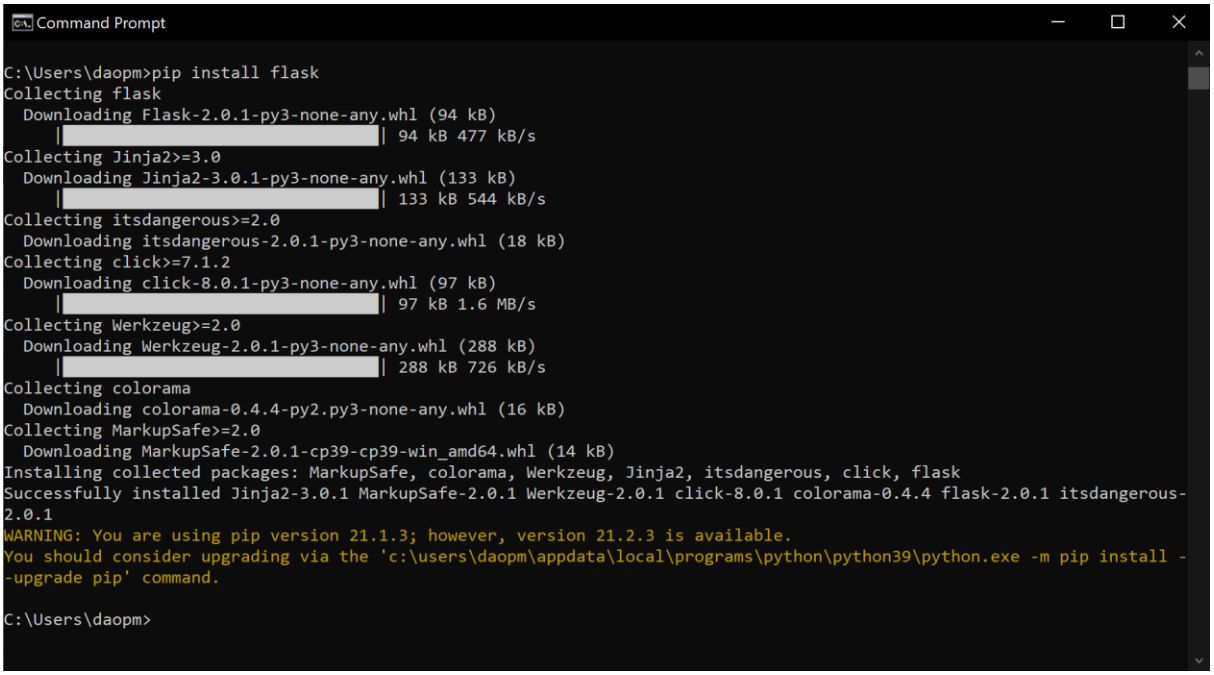
Step 1: Download Python at

https://www.python.org/downloads

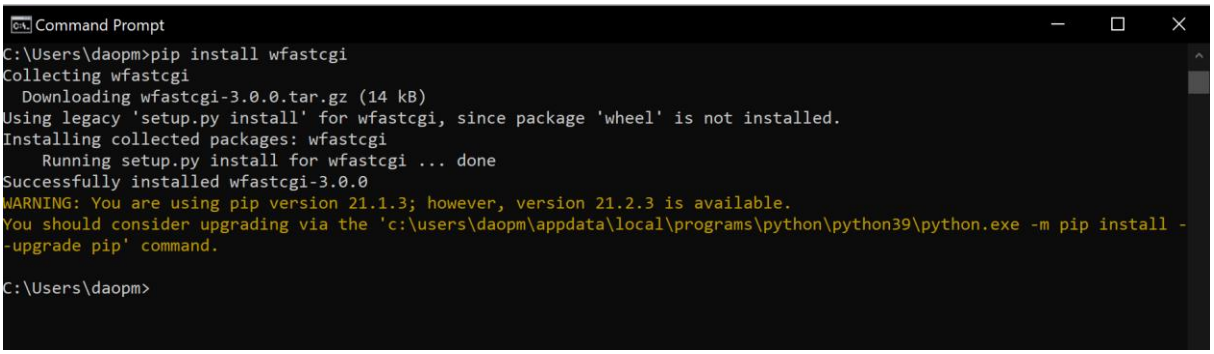
Step 2: Install “flask” and “wfastcgi”

Open CMD and type

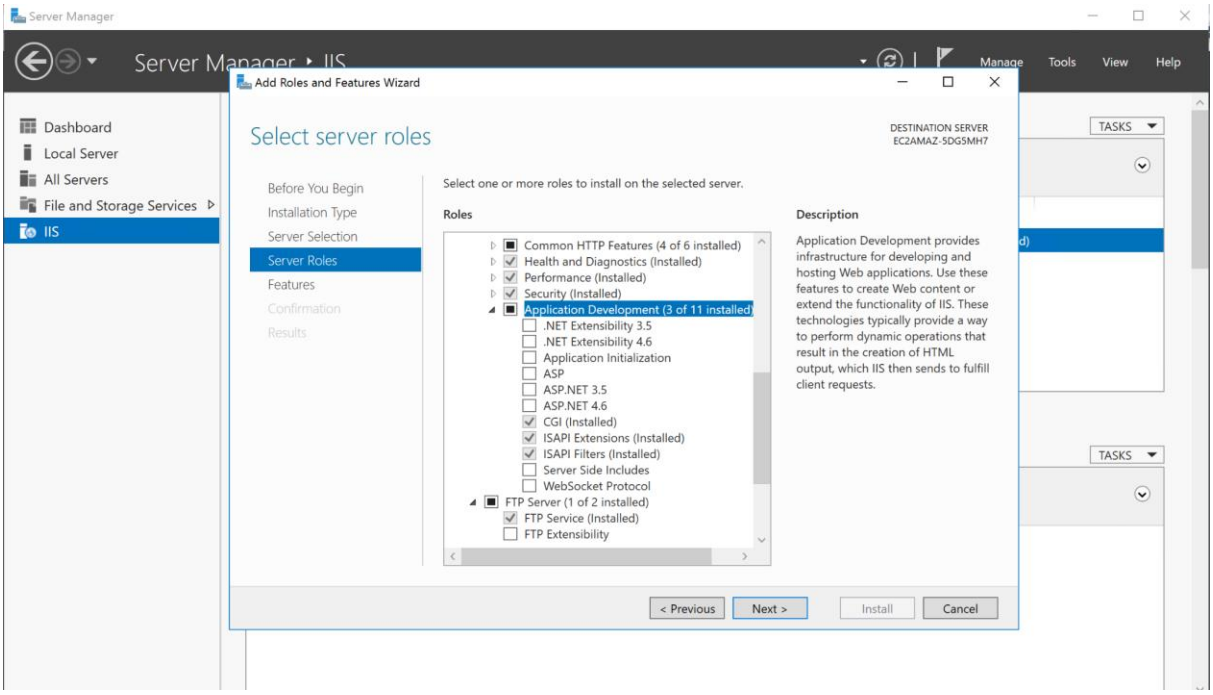
pip install flask



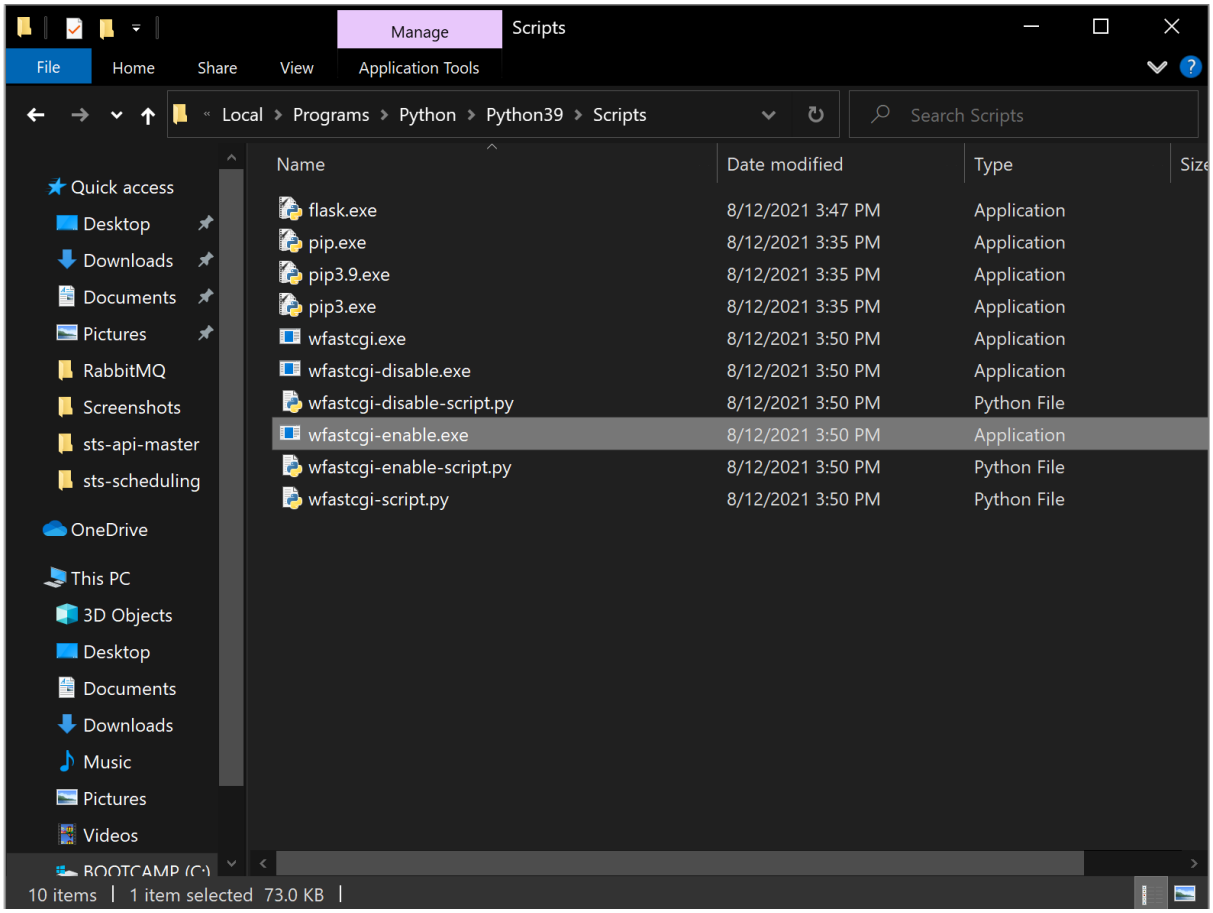
pip install wfastcgi



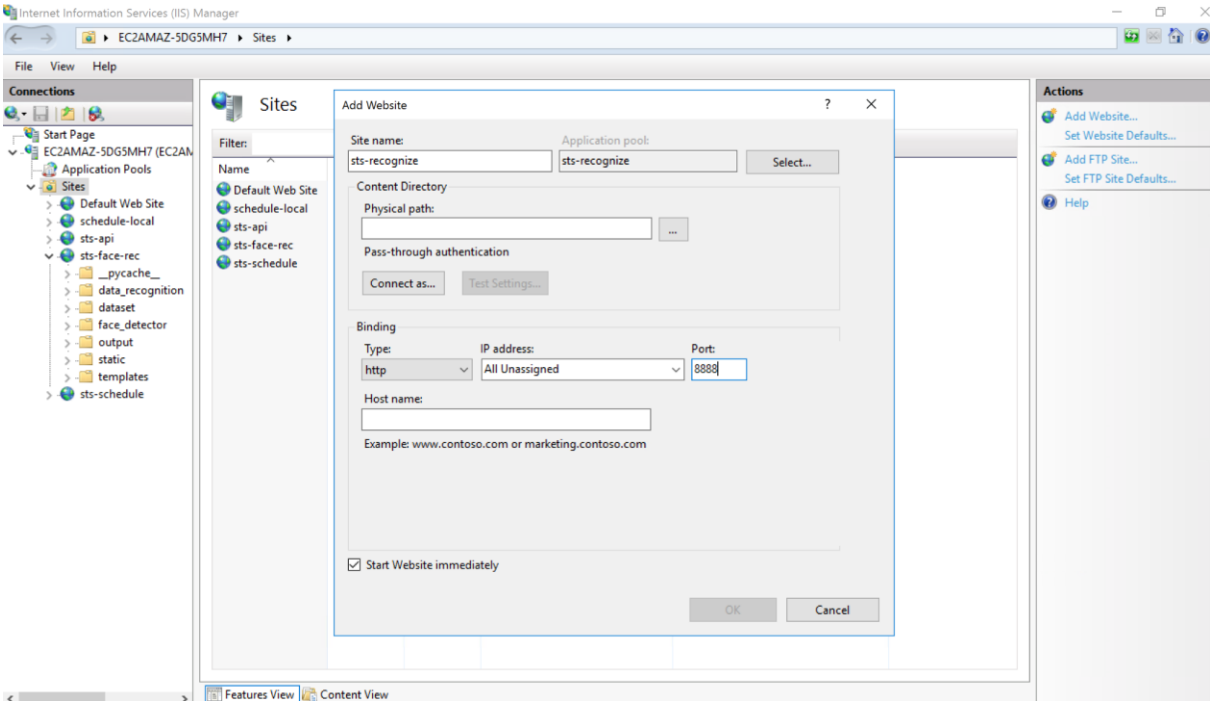
Step 3: Config IIS feature with CGI role



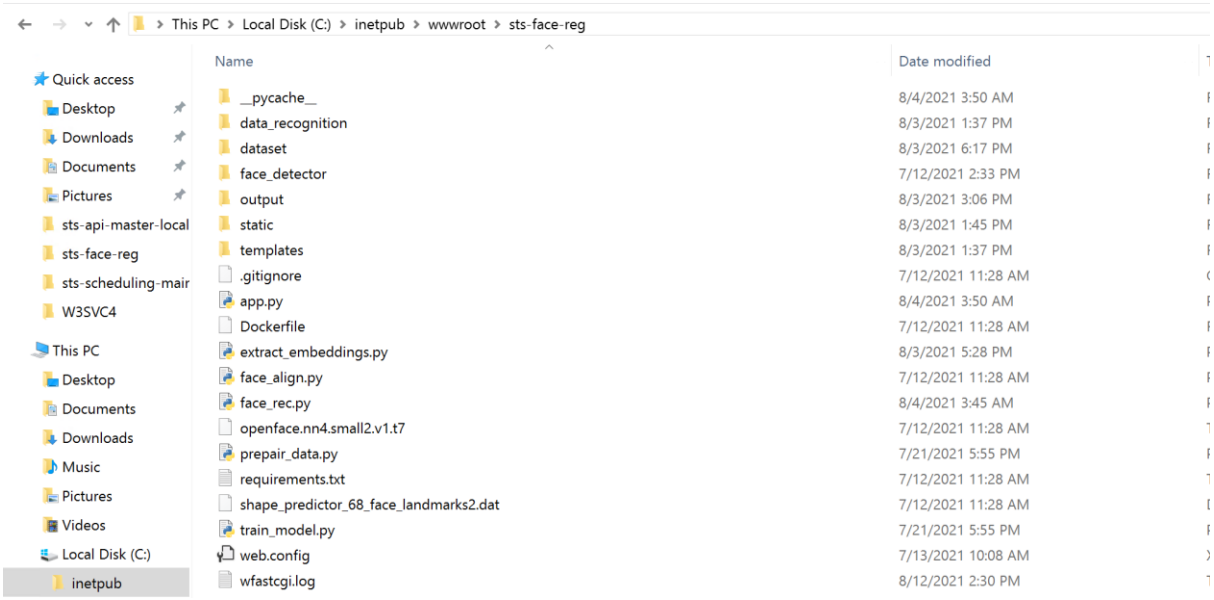
Step 4: Open wfascgi-enable.exe.



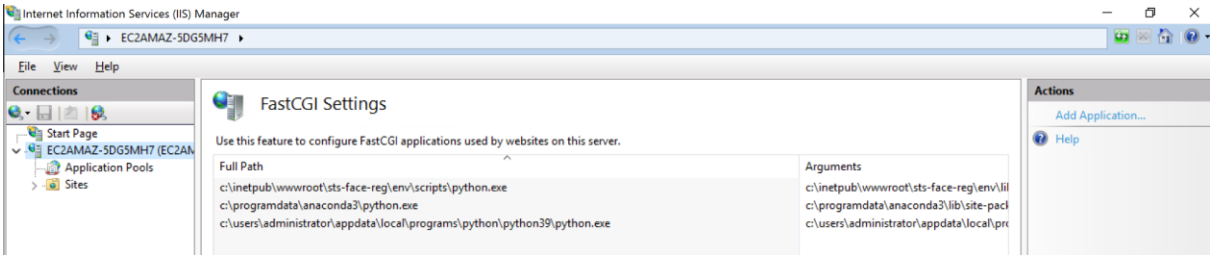
Step 5: Open IIS Manager and Add new website



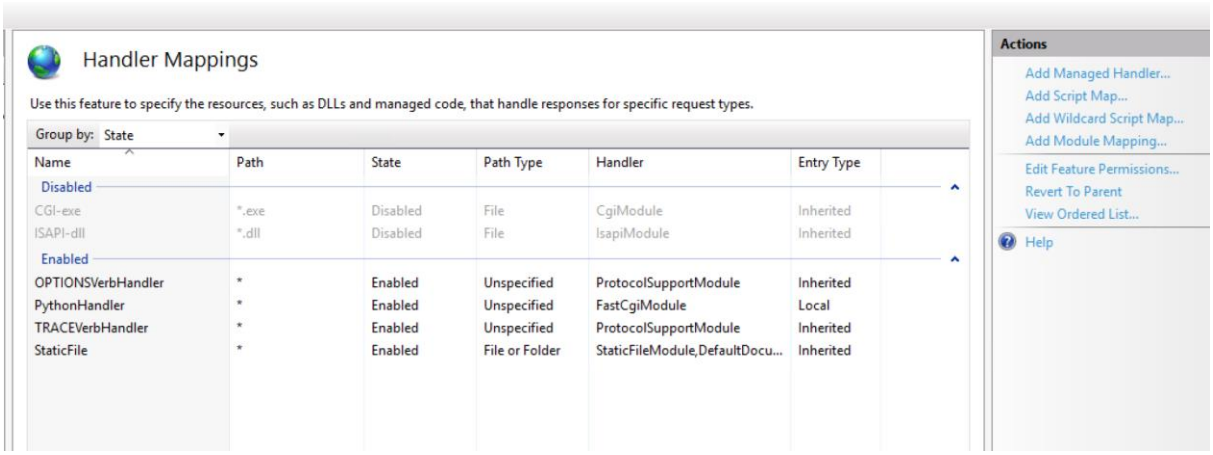
Step 6: Move source code to wwwroot folder

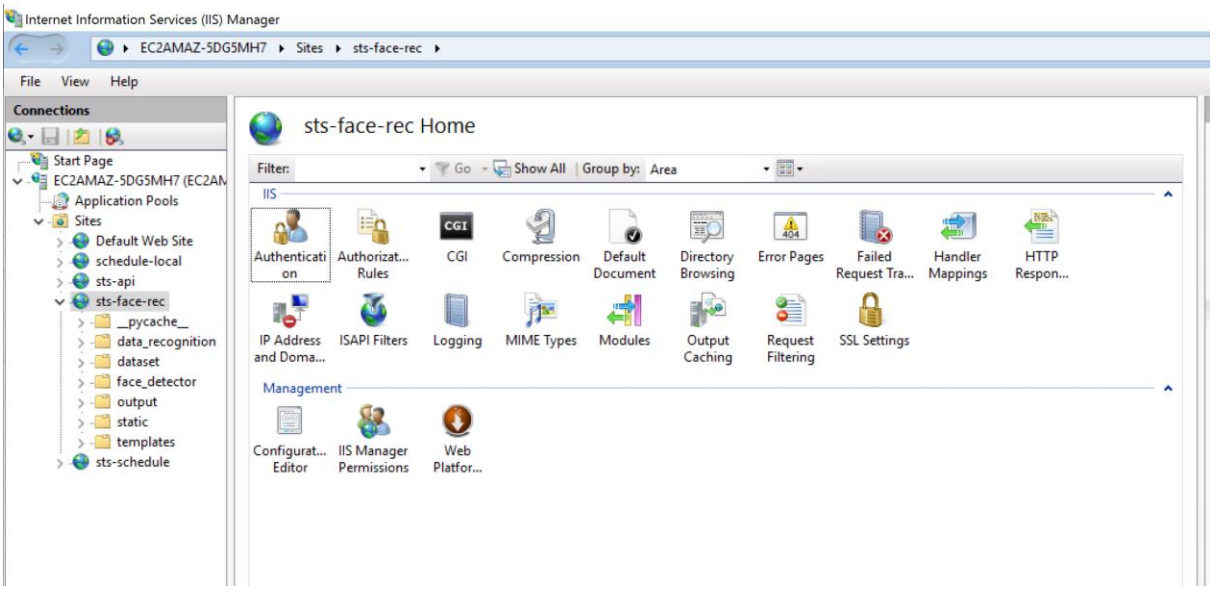


Step 7: Update FastCGI Application



Step 8: Setting handler mapping

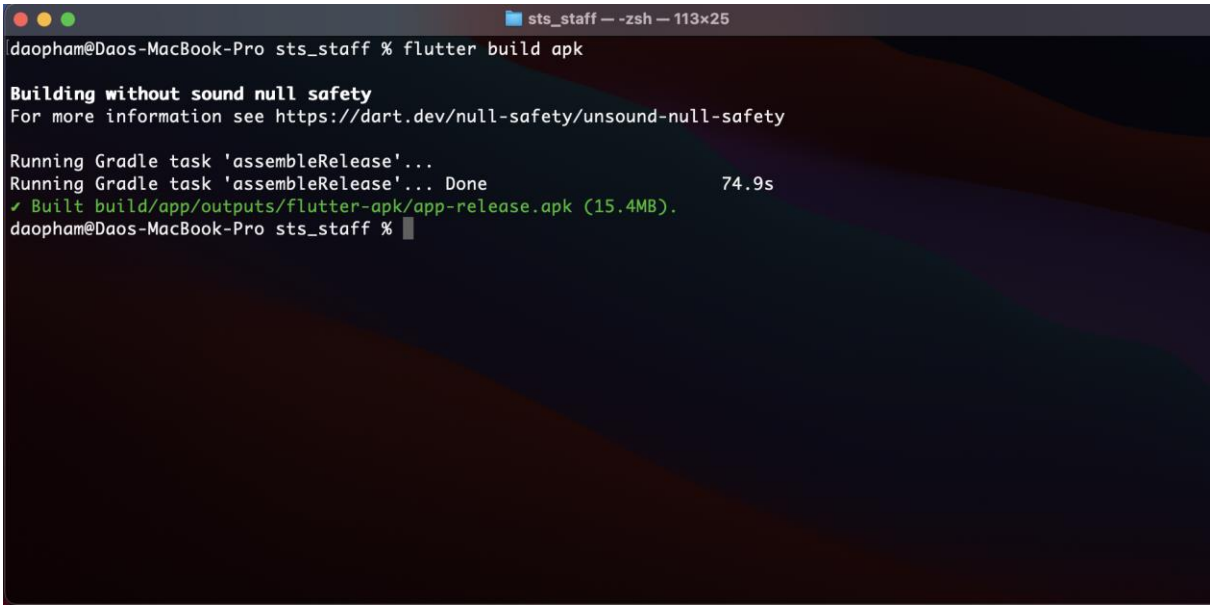




2.3.6 Set up Mobile Application

Step 1: Download flutter at: https://flutter.dev/docs/get-started/install

Step 2: Open folder contains source code, build apk file.



Step 3: Download and install apk file on your android device.

3. User Manual

3.1 System requirements

- Stable internet connection
- Browser: Google Chrome, Firefox, Internet Explorer,...
- Mobile Device: Android 9, iOS 13

3.2 Application Usage

a. Overview

This system use guide include 5 module:

- Admin Module
- Brand Manager Module
- Store Manager Module
- Staff Module
- Store Face Recognize Module

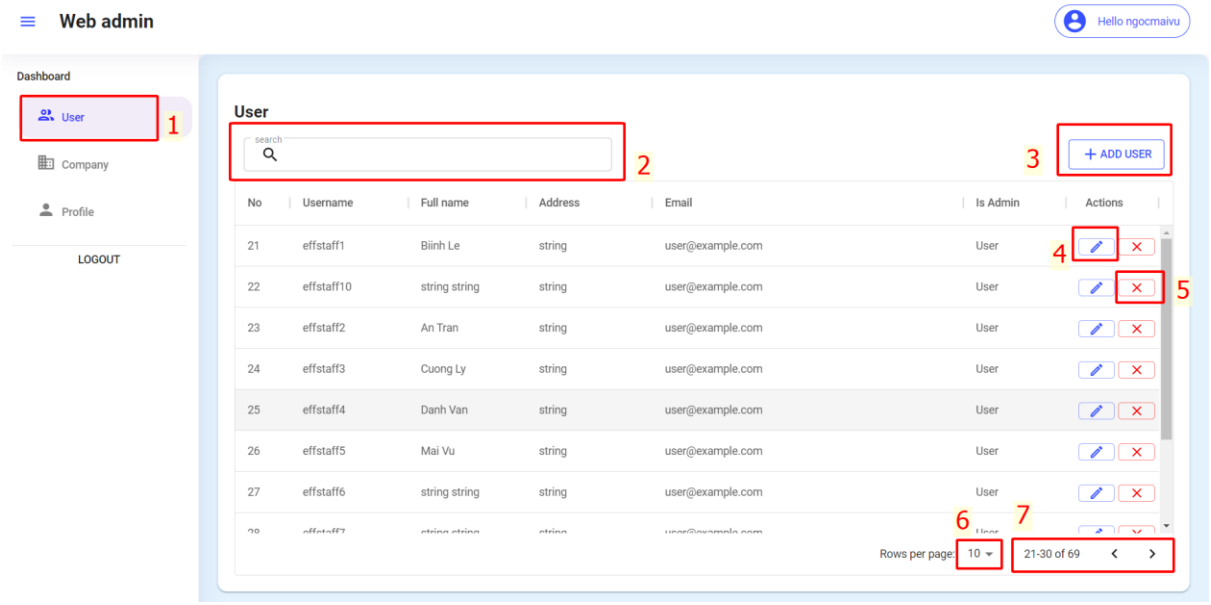
b. Admin Module

b1. Login

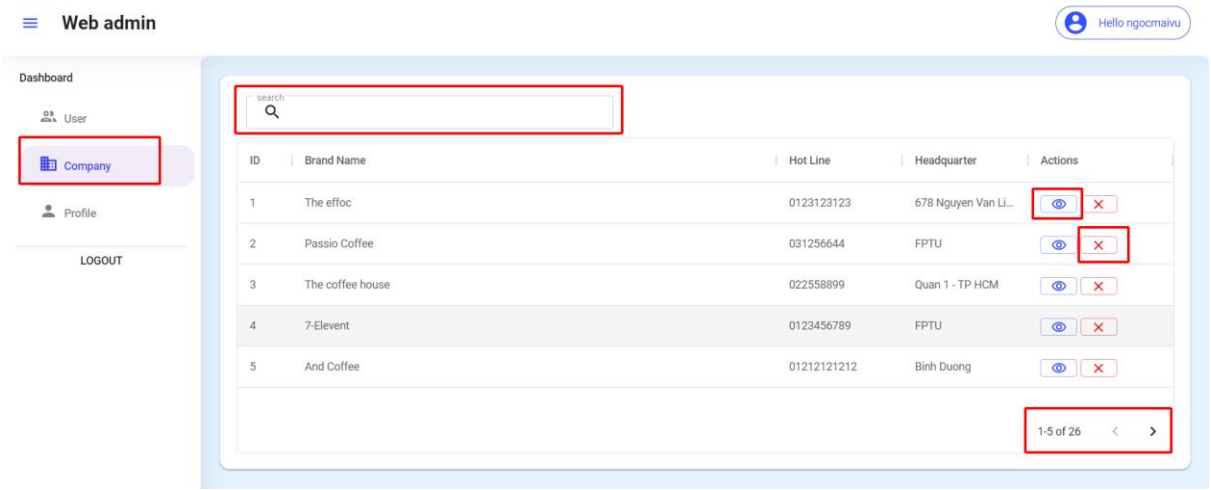
No	Field Name	Description	Read only	Mandatory	Control Type	Data Type
1	Username	Field to input username	Yes	Yes	Text Box	string
2	Password	Field to input password	Yes	Yes	Password	string

No	Function	Description	Validation	Outcome
1	Login	Login to the system	No	

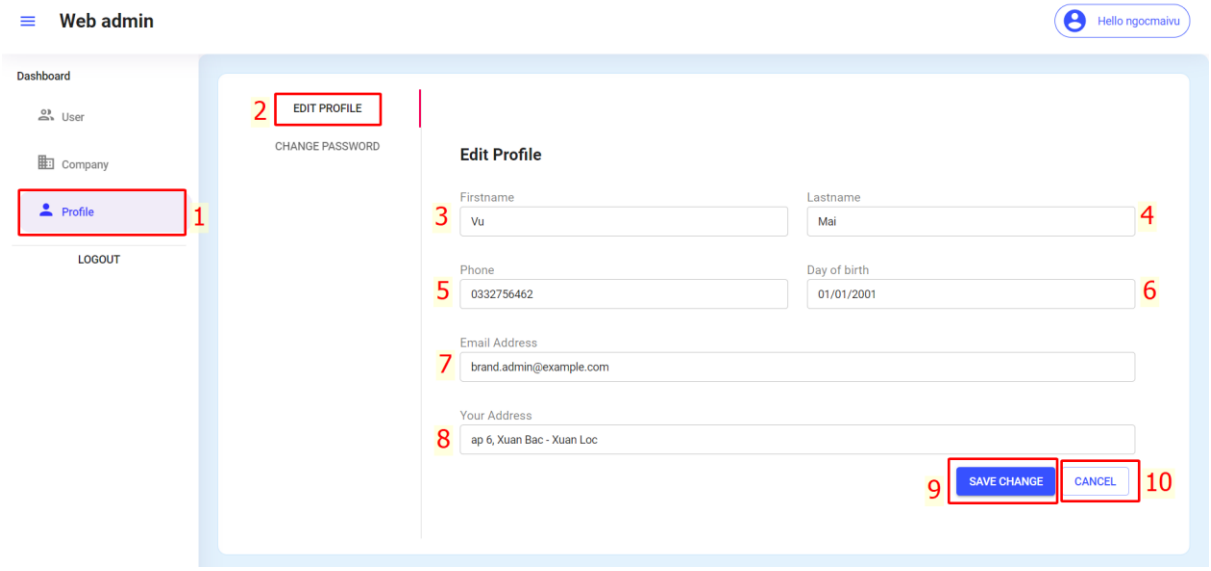
b2. Manage Users



b3. Manage Brands

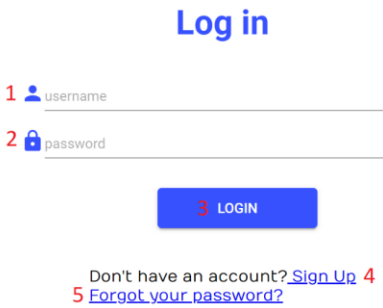


b4.Edit Profile



c. Brand Manager Module

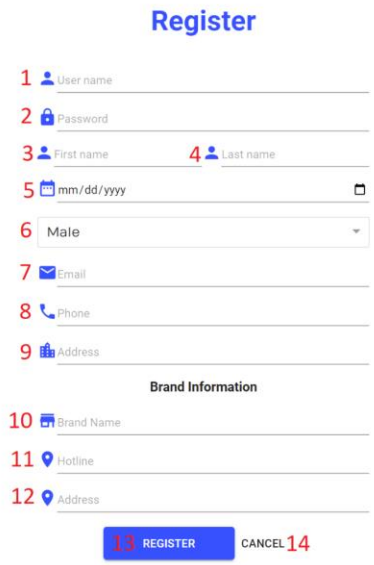
c1. Login



No	Field Name	Description	Read only	Mandatory	Control Type	Data Type
1	Username	Field to input username	Yes	Yes	Text Box	string
2	Password	Field to input password	Yes	Yes	Password	string

No	Function	Description	Validation	Outcome
3	Login	Login to the system	Yes	
4	Sign Up	Sign up a new account	No	
5	Forgot Password	Forgot password	No	

c2. Register

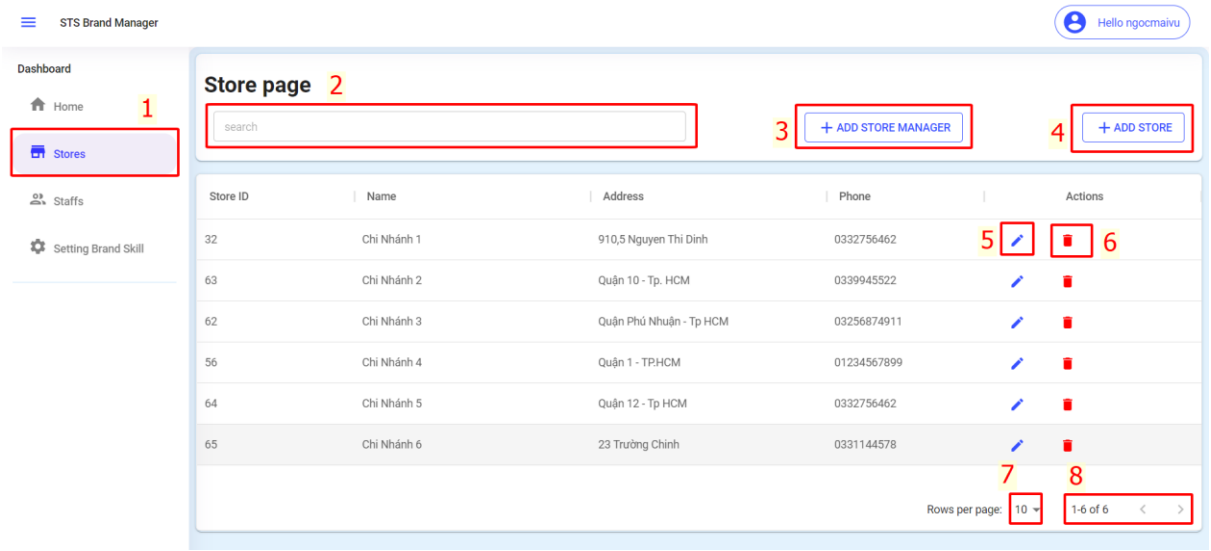


No	Field Name	Description	Read only	Mandatory	Control Type	Data Type
1	Username	Field to input username	No	Yes	Text Box	string
2	Password	Field to input password	No	Yes	Password	string
3	First Name	Field to input first name	No	Yes	Text Box	string
4	Last Name	Field to input last name	No	Yes	Text Box	string
5	Dob	Field to input	No	Yes	Date	Date

		date of birth			Picker	
6	Gender	Field to choose gender	Yes	Yes	Dropbox	Number
7	Email	Field to input email	No	Yes	Text Box	email
8	Phone	Field to input phone	No	Yes	Text box	number
9	Address	Field to input address	No	Yes	Text Box	string
10	Brand Name	Field to input brand name	No	Yes	Text Box	String
11	Brand Hotline	Field to input brand hotline	No	Yes	Text Box	Number
12	Brand Address	Field to input brand address	No	Yes	Text Box	String

No	Function	Description	Validation	Outcome
13	Register	Register User and Brand	Yes	
14	Cancel	Return to home page	No	

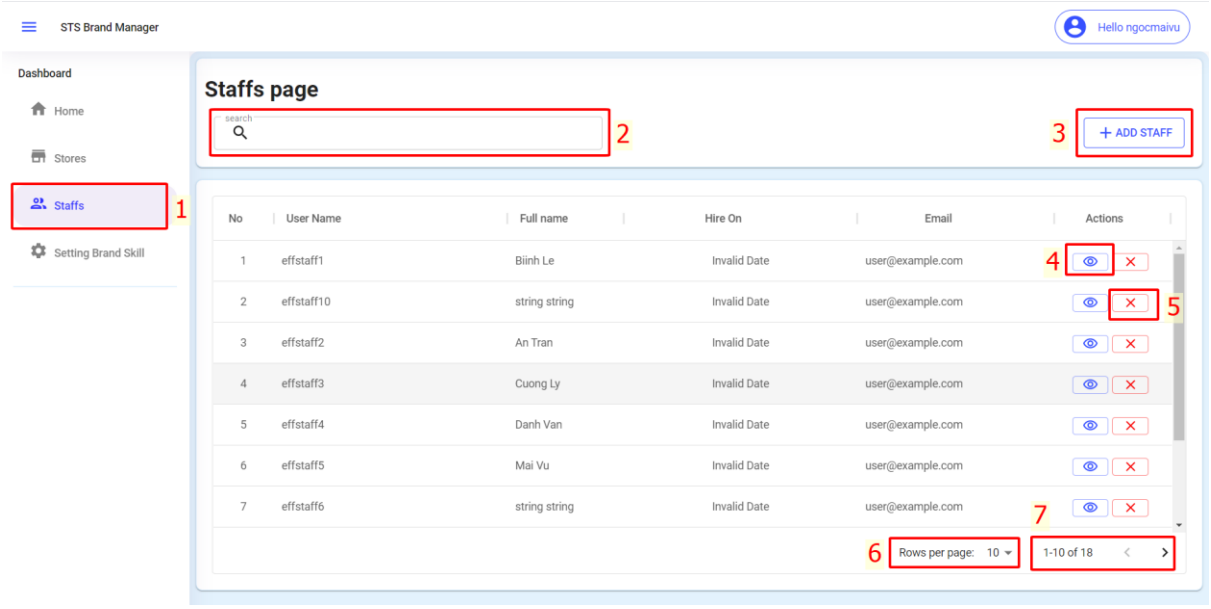
c3. Manage Stores



No	Field Name	Description	Read only	Mandatory	Control Type	Data Type
2	Search	Search by store name.	No	No	Search Box	string

No	Function	Description	Validation	Outcome
3	Add store manager	Click to open add store manager page	No	Add store manager Page
4	Add store	Open add store page	No	Add store Page
5	Edit	Edit store information	No	Edit Store Page
6	Delete	Delete store	No	
7	Row per page	Number of rows is allowed to show in one page.	No	
8	Paging	Pagination for easy viewing.	No	

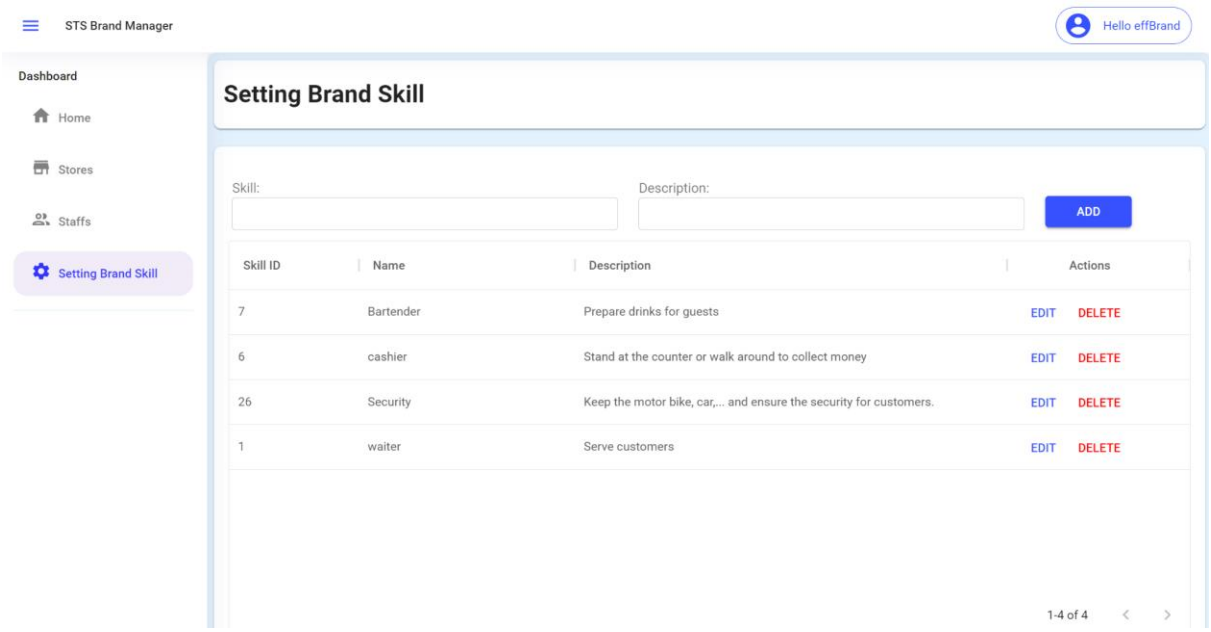
c4. Manage Staff



No	Field Name	Description	Read only	Mandatory	Control Type	Data Type
2	Search	Search by staff name.	No	No	Search Box	string

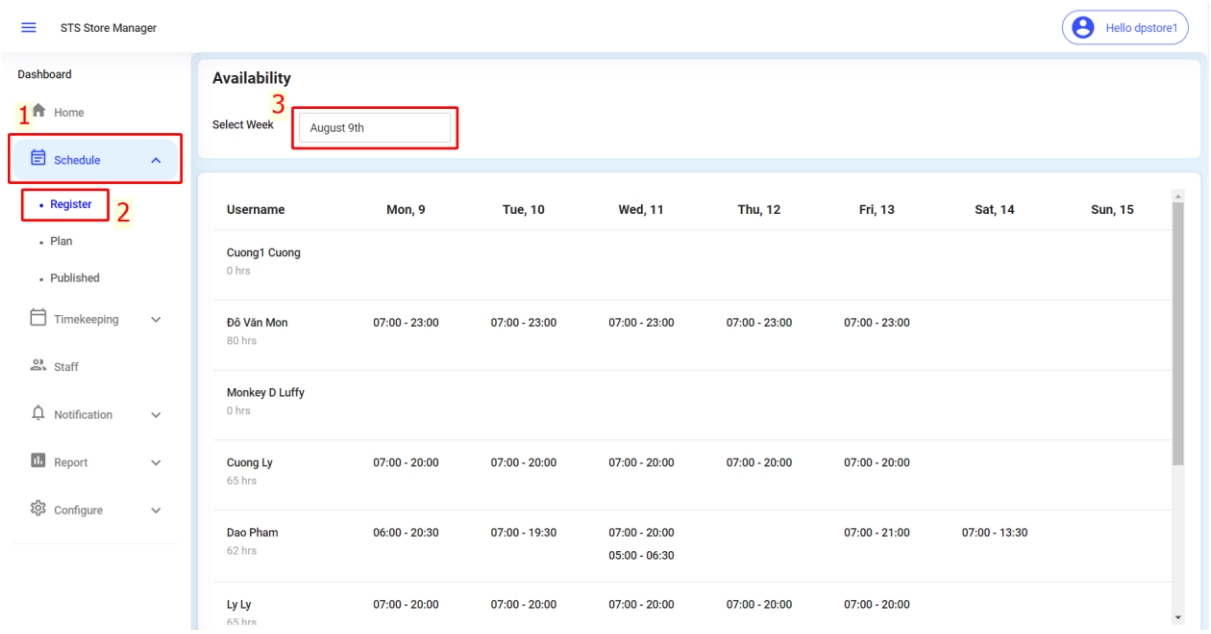
No	Function	Description	Validation	Outcome
3	Add staff	Click to open add store manager page	No	Add StaffPage
4	View	View staff information	No	
5	Delete	Delete staff	No	
6	Row per page	Number of rows is allowed to show in one page.	No	
7	Paging	Pagination for easy viewing.		

c5. Setting Skills



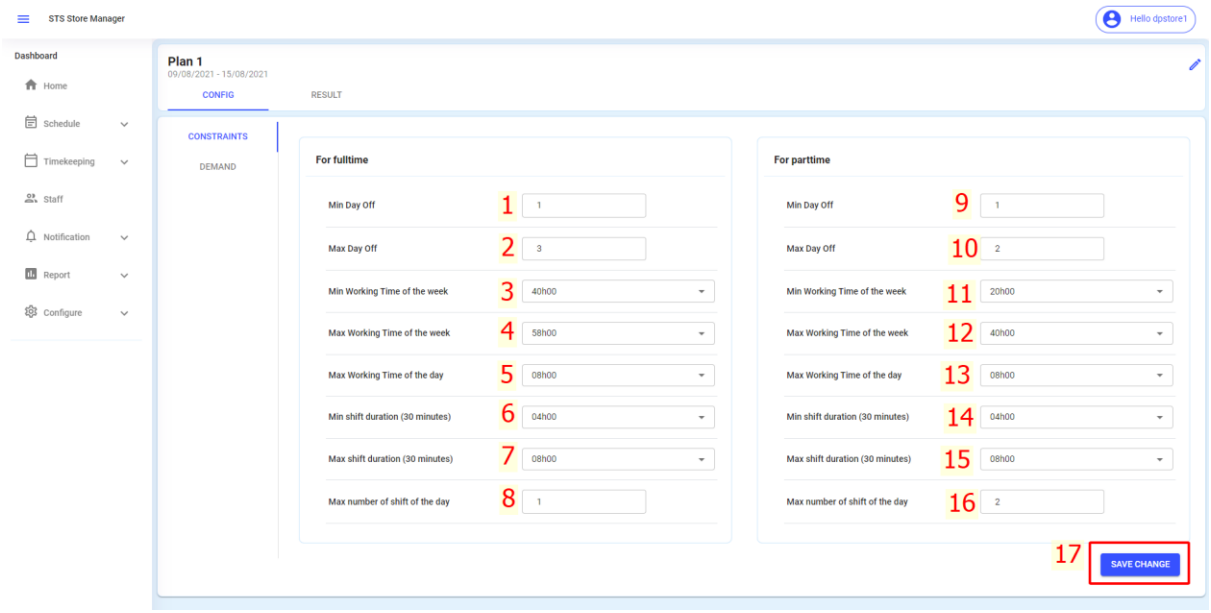
d. Store Manager Module

d1. View Staff Available Time



No	Field Name	Description	Read only	Mandatory	Control Type	Data Type
3	Select week	Choose week to see available time to work of staff	Yes	Yes	Week picker	Date - Week

d2. Config schedule plan - Constraints

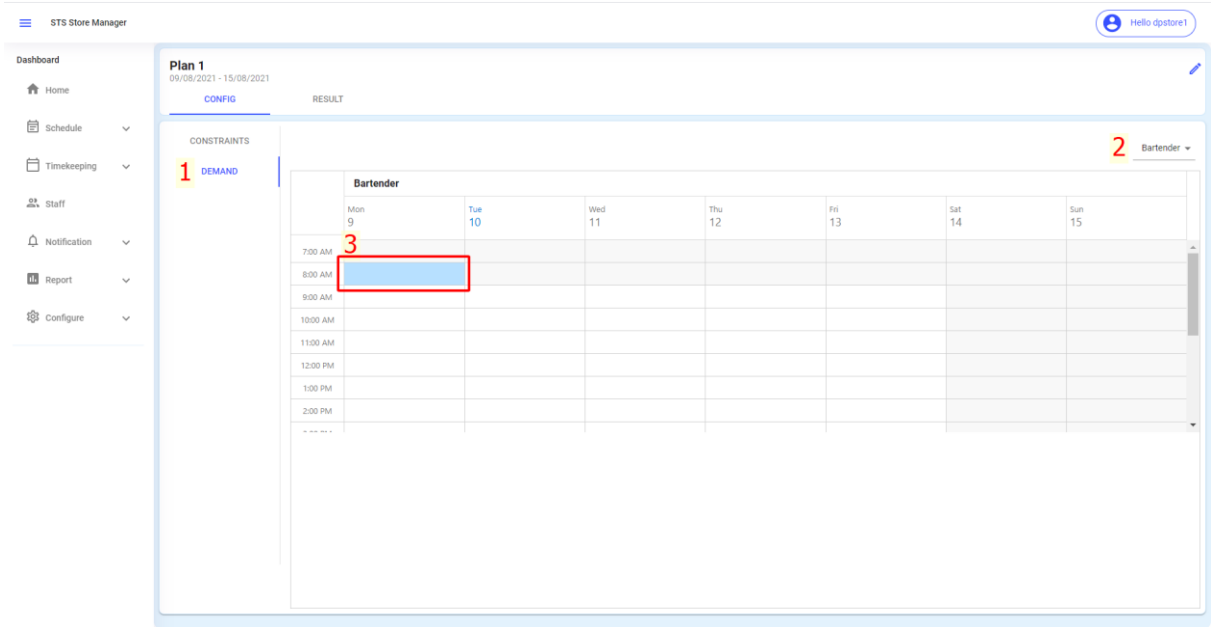


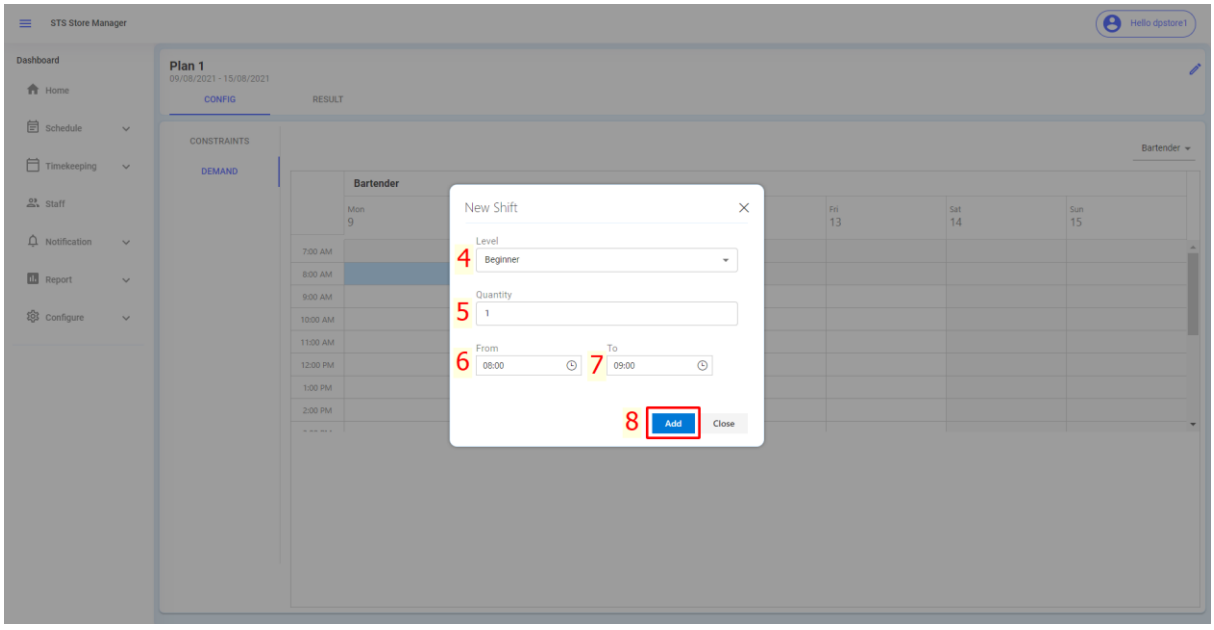
No	Field Name	Description	Read only	Mandatory	Control Type	Data Type
1	Min Day Off	Min Day Off	Yes	No	Text Box Number	Number
2	Max Day Off	Max Day Off	Yes	No	Text Box Number	Number
3	Min Working Time of the week	Min Working Time of the week	Yes	No	Dropdown	Hour
4	Max Working Time of the week	Max Working Time of the week	Yes	No	Dropdown	Hour
5	Max Working Time of the day	Max Working Time of the day	Yes	No	Dropdown	Hour
6	Min shift duration (30 minutes)	Min shift duration (30 minutes)	Yes	No	Dropdown	Hour
7	Max shift duration (30 minutes)	Max shift duration (30 minutes)	Yes	No	Dropdown	Hour
8	Max number of shift of	Max number of shift of the day	Yes	No	Text Box Number	Number

	the day					

No	Function	Description	Validation	Outcome
17	Save	Save new edit constraints.	No	

d3. Config schedule plan - Demands

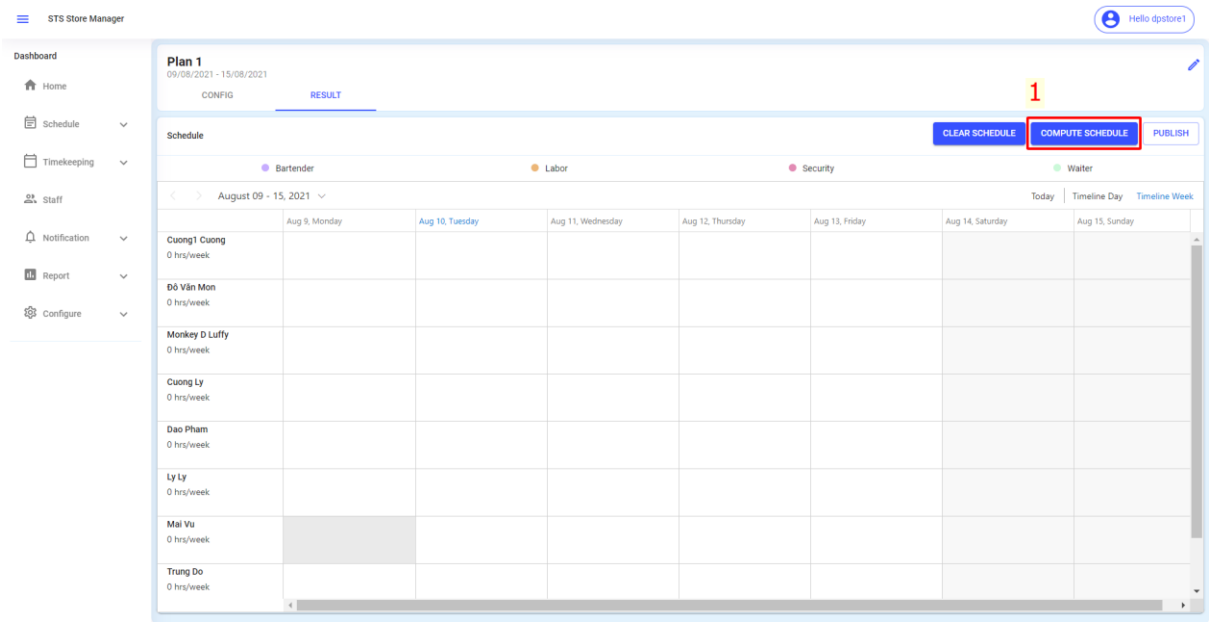




No	Field Name	Description	Read only	Mandatory	Control Type	Data Type
2	Select skill	Select staff skill	No	No	Dropdown	String
3	Shift	Click to create a new shift.	Yes	Yes		
4	Level	Level of staff skill	Yes	Yes	Dropdown	
5	Quantity	Quantity staff are needed in this shift.	No	Yes	Text Box	Number
6	From	Work hour From	Yes	Yes	Time	Time
7	To	Work hour To	Yes	Yes	Time	Time

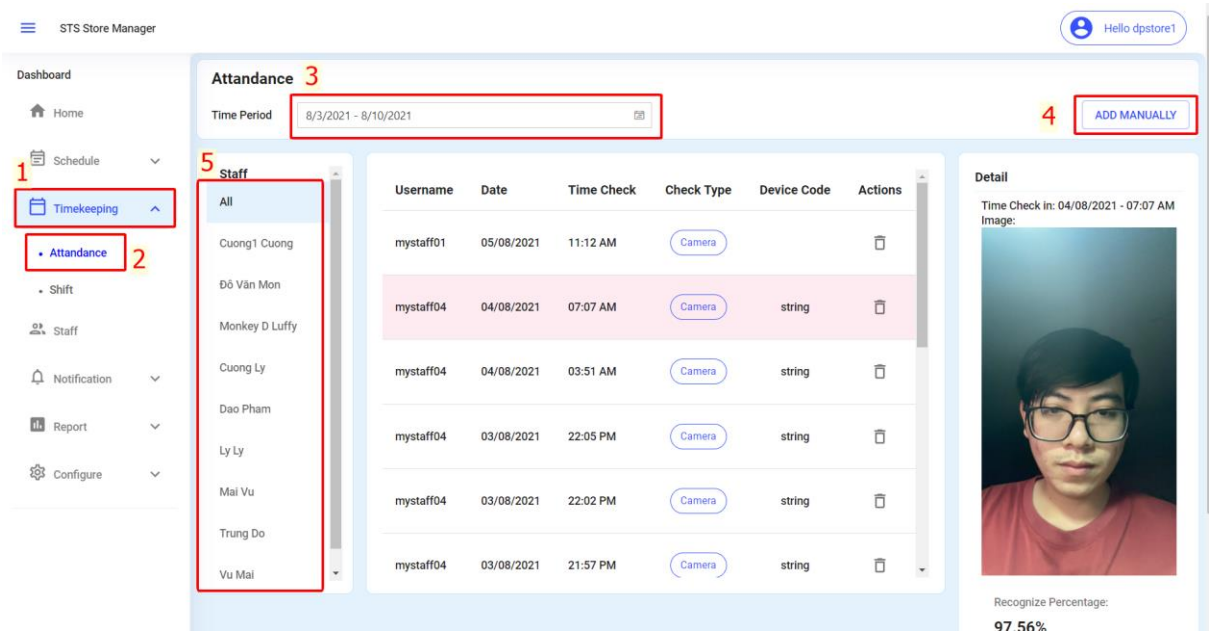
No	Function	Description	Validation	Outcome
8	Add	Add new shift with request personnel for that shift	No	

d4. Compute schedule



No	Function	Description	Validation	Outcome
1	Compute Schedule	Compute schedule based on config and available time to work of staff.	No	Schedule is computed and appears on the screen. If alogithm can't compute, the popup "Unable to schedule".

d5. View Attendance

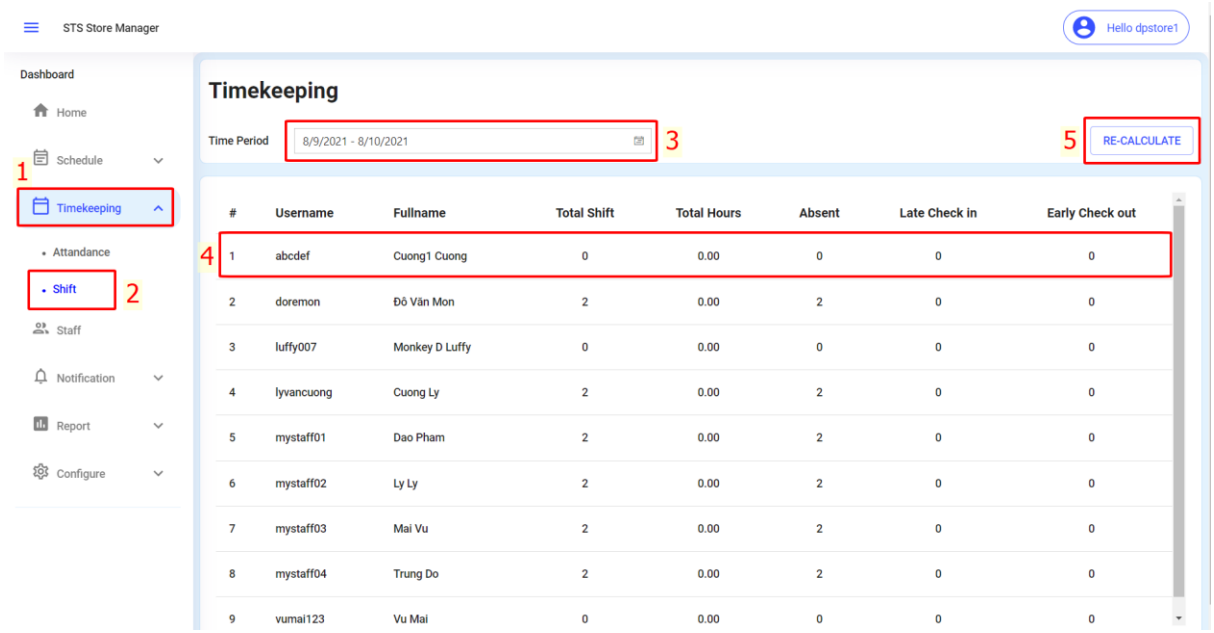


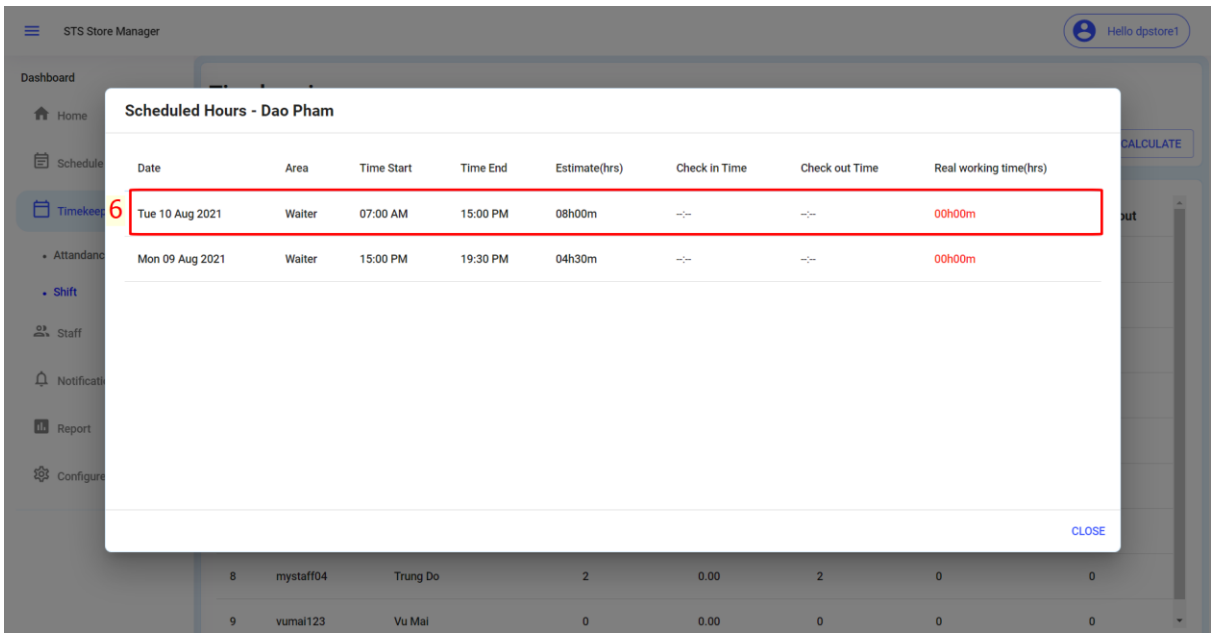
No	Field Name	Description	Read only	Mandatory	Control Type	Data Type
3	Time period	Choose a time period to view attendance at this time.	Yes	No	Date time picker	Date

No	Function	Description	Validation	Outcome
4	Add manually	Add manual attendance in case the timekeeper is	No	Add manually popup

		broken or wrong,...		
5	Choose staff	Select a staff to view his or her attendance for the selected time period.	No	Detail of staff's attendance at right screen.

d6. View shift

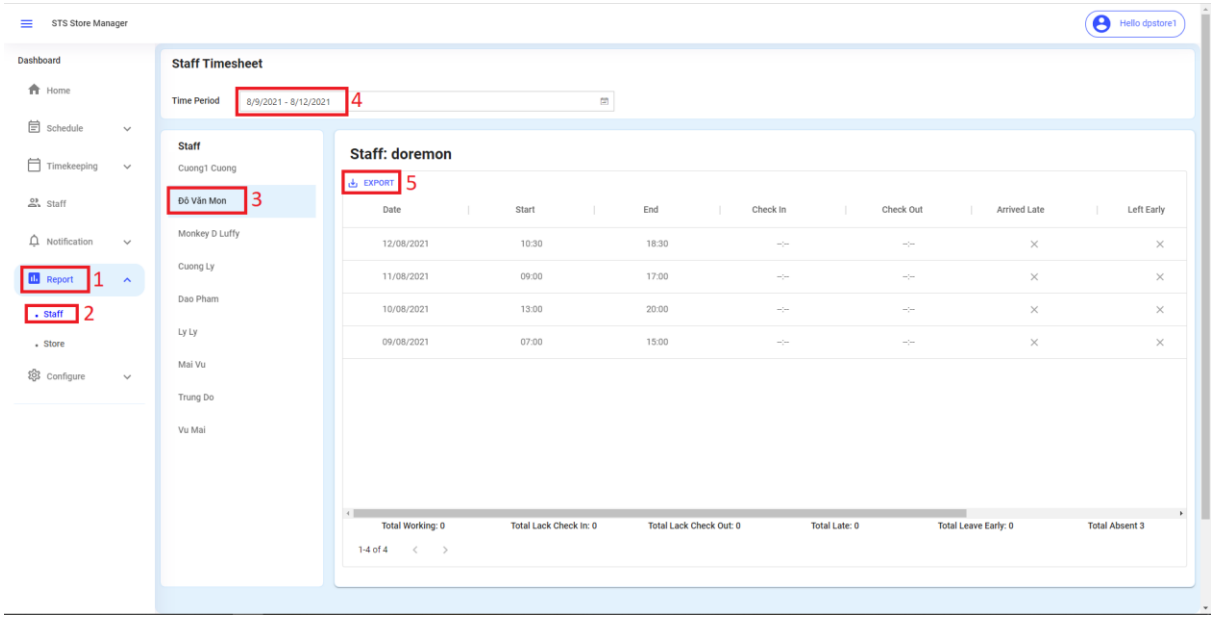




No	Field Name	Description	Read only	Mandatory	Control Type	Data Type
3	Time period	Choose a time period to view timekeeping at this time.	Yes	No	Date time picker	Date

No	Function	Description	Validation	Outcome
4	Timekeeping detail	Timekeeping detail of each staff	No	
5	Recalculate	Recalculate real working time, total shift, total hour,..	No	New timekeeping detail

d7. View report of staff

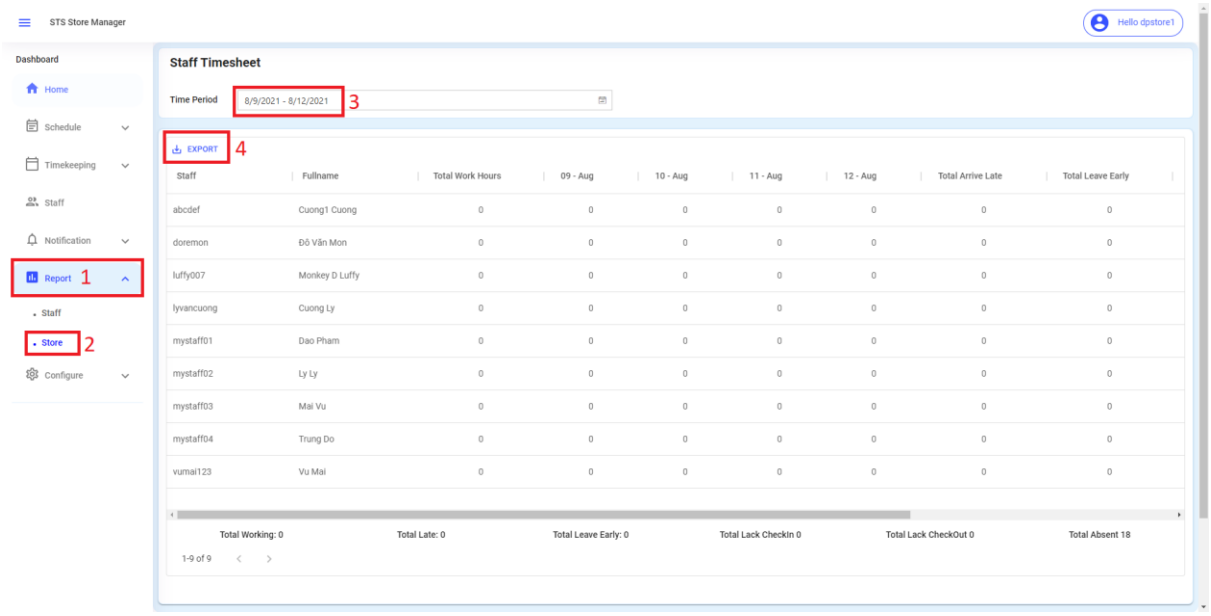


No	Field Name	Description	Read only	Mandatory	Control Type	Data Type
4	Time period	Choose a time period to view the report at this time	Yes	No	Date time picker	Date

		range.				

No	Function	Description	Validation	Outcome
3	Choose Staff	Choose staff to show report	No	
5	Export	Export report to csv	No	csv file

d8. View report of store

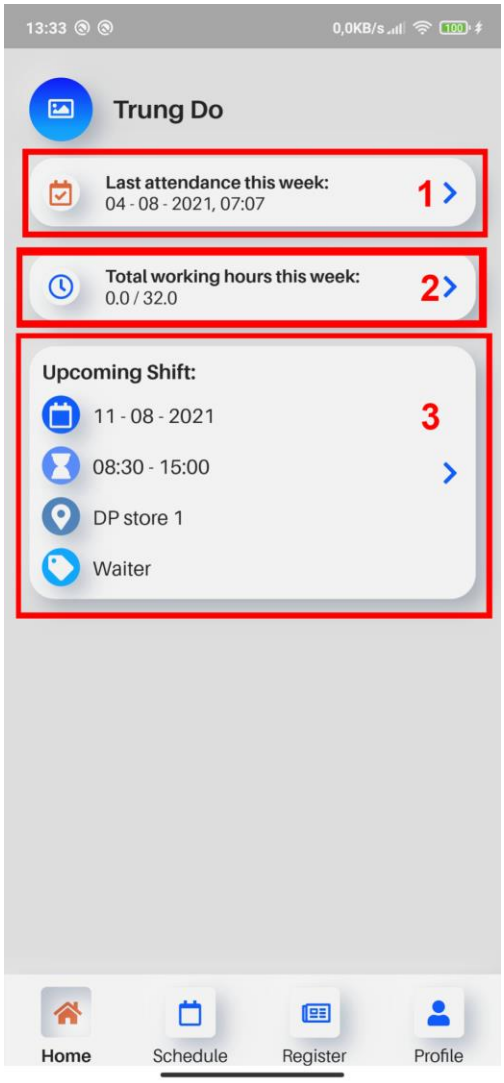


No	Field Name	Description	Read only	Mandatory	Control Type	Data Type
3	Time period	Choose a time period to view the report at this time range.	Yes	No	Date time picker	Date

No	Function	Description	Validation	Outcome
4	Export	Export report to csv	No	csv file

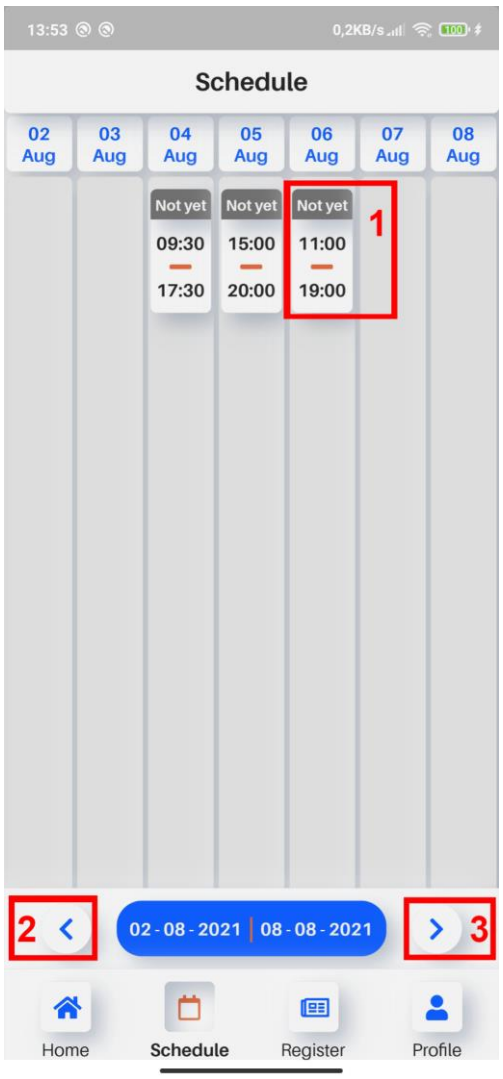
e. Staff Module

e1. Home Screen



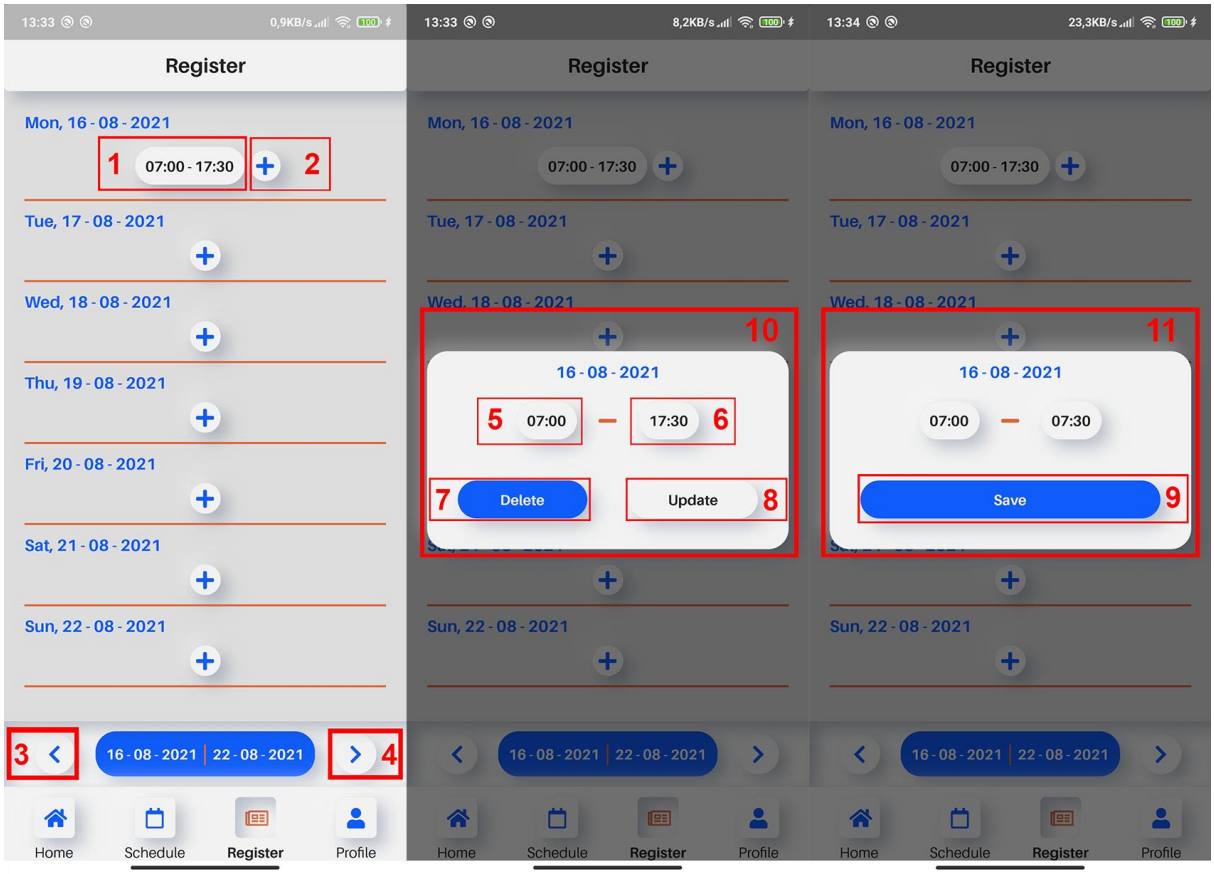
No	Function	Description	Validation	Outcome
1	View Attendance report	View attendance report of current week	No	Go to Attendance report screen
2	View Work report	View work report of current week	No	Go to Work report screen
3	View detail of upcoming shift	View detail of upcoming shift	No	Go to Shift detail screen

e2. Schedule Screen



No	Function	Description	Validation	Outcome
1	View Shift detail	View detail of selected shift	No	Go to Shift detail screen
2	View schedule of previous week	View schedule of previous week	No	Display schedule of previous week
3	View schedule of next week	View schedule of next week	No	Display schedule of next week

e3. Register Screen



No	Field Name	Description	Read only	Mandatory	Control Type	Data Type
5	Time start	Pick time start for available time	No	Yes	Time picker	DateTime
6	Time End	Pick time end for available time	No	Yes	Time picker	DateTime
10	Edit available time dialog	This dialog is used to update or delete available time	No	Yes	Dialog box	DateTime Range
11	Create available time dialog	This dialog is used to create available time	No	Yes	Dialog box	DateTime Range

No	Function	Description	Validation	Outcome
1	Open edit	Open dialog to	Yes	Display edit

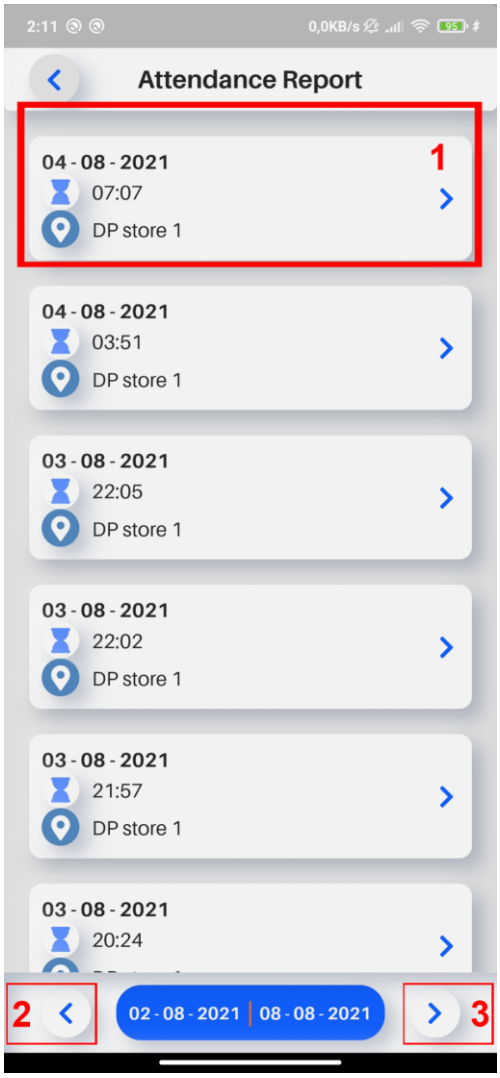
	available time dialog	edit available time		available time dialog
2	Open create available time dialog	Open dialog to create available time	Yes	Display create available time dialog
3	View available time of previous week	View available time of previous week	No	Display available time of previous week
4	View available time of next week	View available time of next week	No	Display available time of next week
7	Delete available time	Delete selected available time	No	Close edit available time dialog and reload list of available time
8	Update available time	Update selected available time	Yes	Close edit available time dialog and reload list of available time
9	Create available time	Create available time	Yes	Close create available time dialog and reload list of available time

e4. Shift Detail Screen

No	Field Name	Description	Read only	Mandatory	Control Type	Data Type
1	Working location	Location where staff work	Yes	Yes	Text field	String
2	Working position	Position of staff	Yes	Yes	Text field	String
3	Working date	working date of shift	Yes	Yes	Text field	String
4	Working time	working time of shift	Yes	Yes	Text field	String

No	Function	Description	Validation	Outcome
5	Request absence	Make request to absent selected shift	Yes	Display confirm dialog
6	Request swap shift	Make request to swap selected shift	Yes	Display confirm dialog

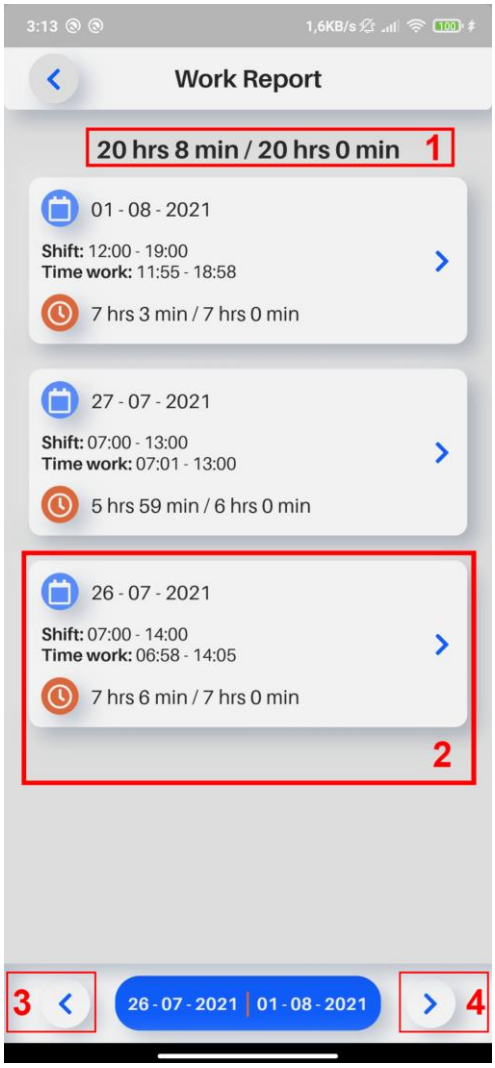
e5. Attendance Report Screen



No	Function	Description	Validation	Outcome
1	View attendance detail	View detailed information of selected	No	Go to Attendance detail screen

		attendance		
2	View previous week	View attendance report of previous week	No	Display attendance report of previous week
3	View next week	View attendance report of next week	No	Display attendance report of next week

e6. Work Report Screen



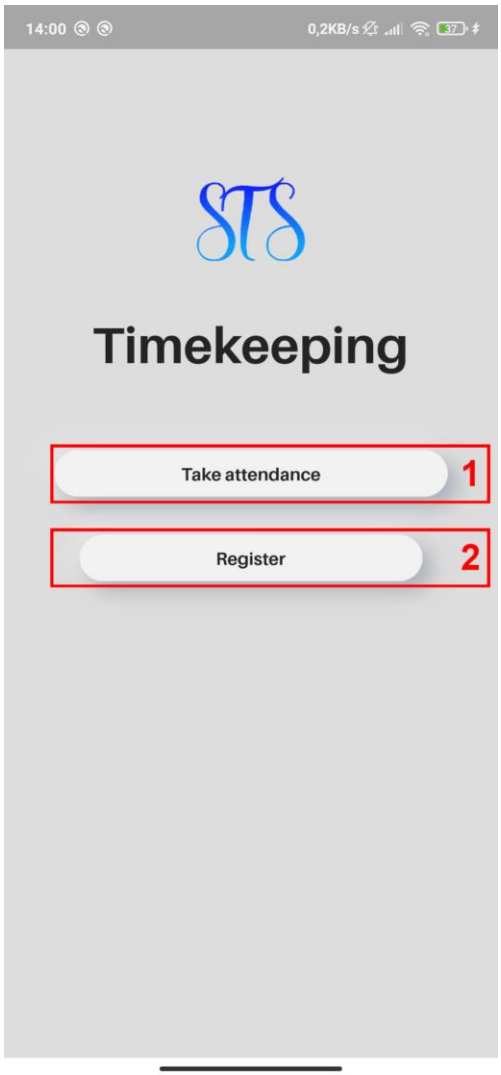
No	Field Name	Description	Read only	Mandatory	Control Type	Data Type

1	Total working hours of week	Total working hours of selected week	Yes	Yes	Text field	String

No	Function	Description	Validation	Outcome
2	View work detai	View detailed information of selected work	No	Go to work detail screen
3	View previous week	View work report of previous week	No	Display work report of previous week
4	View next week	View work report of next week	No	Display work report of next week

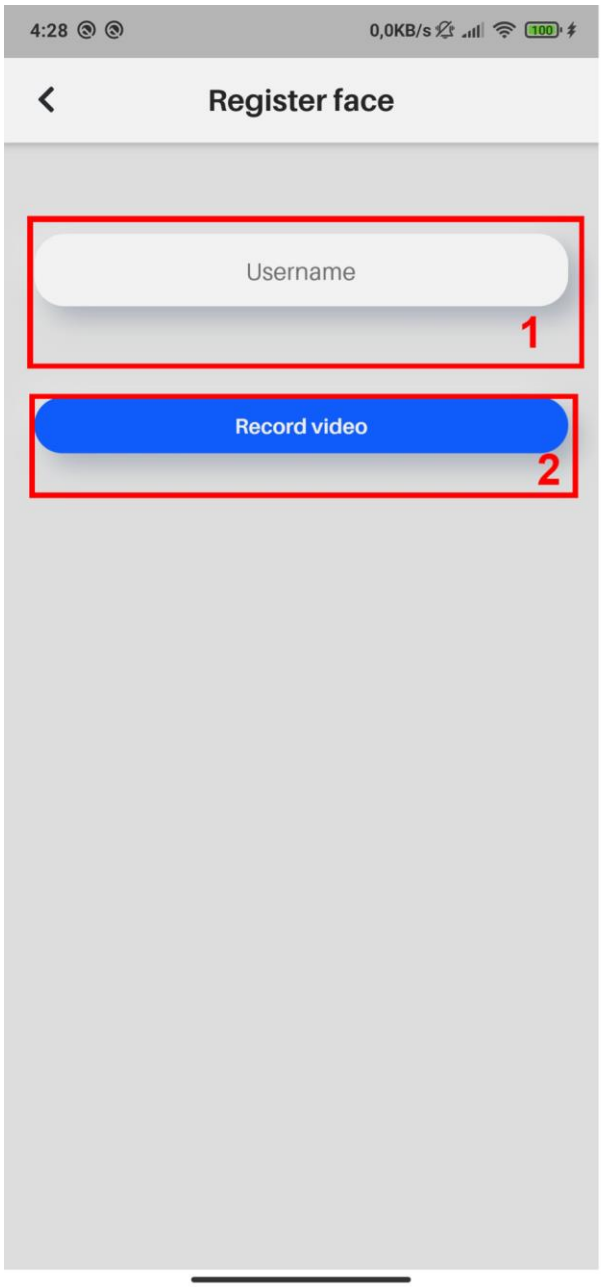
f. Store Timekeeping Module

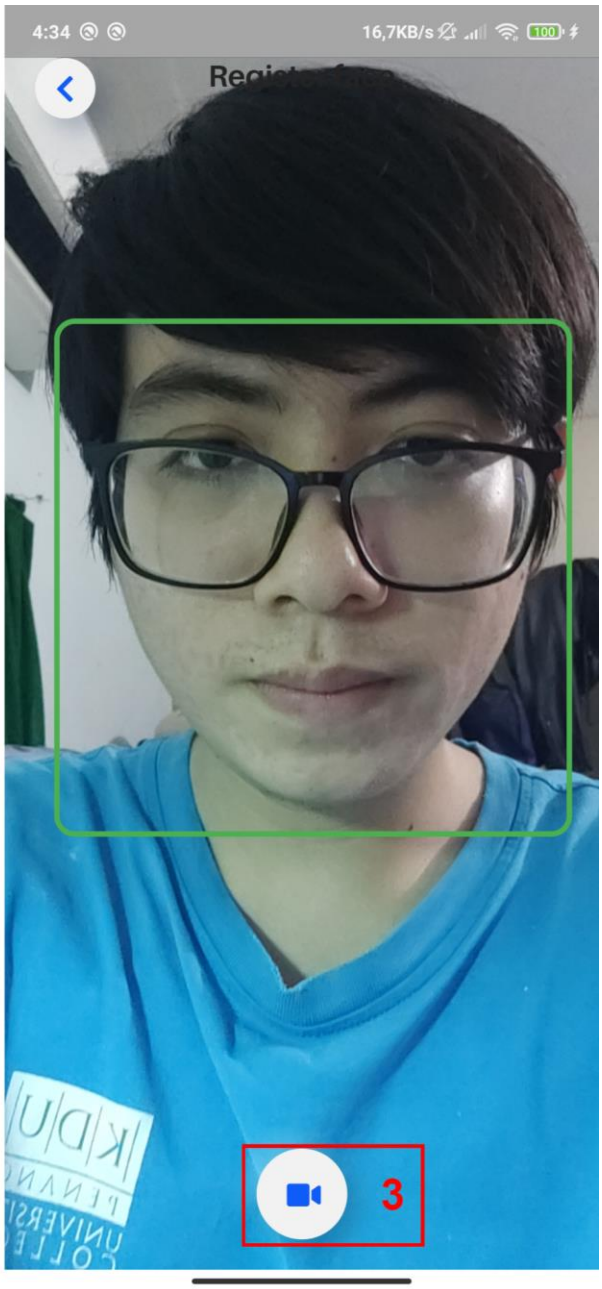
f1. Home Screen



No	Function	Description	Validation	Outcome
2	Take attendance	Staff takes attendance	No	Go to Take attendance screen
3	Register face	Register face for recognition	No	Go to Register face screen

f2. Register Face identity

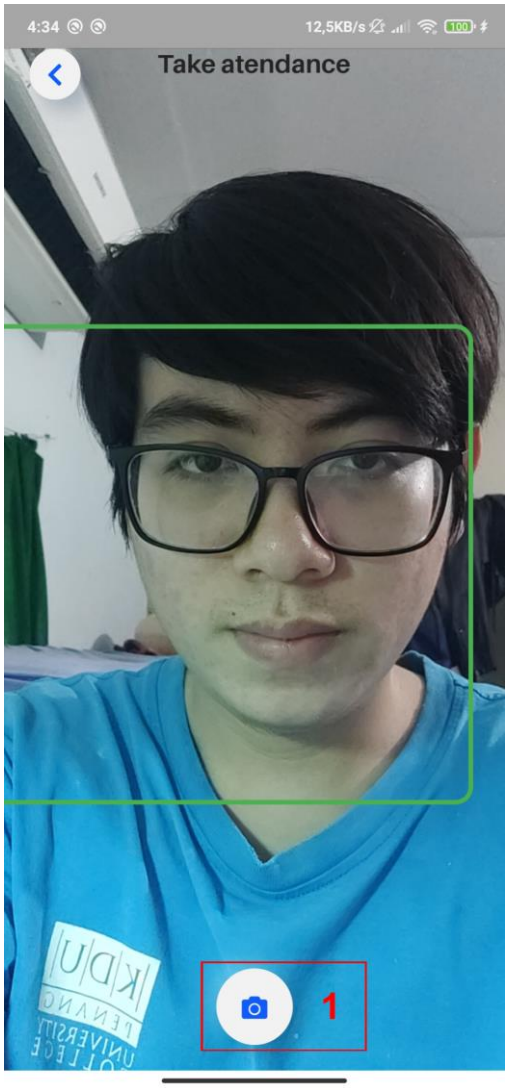




No	Field Name	Description	Read only	Mandatory	Control Type	Data Type
1	Username	Staff ‘s username	No	Yes	Text field	String

No	Function	Description	Validation	Outcome
2	Record video	Record video of staff	No	Go to record video screen
3	Record video and save registration	Start to record video and save registration	No	Display result dialog

f3. Take Attendance



No	Function	Description	Validation	Outcome
1	Take attendance	Staff take attendance	Yes	Display result dialog

VII. Appendix

1. UML
https://www.uml-diagrams.org

2. Scrum Framework
https://www.scrum.org

3. Firebase
https://firebase.google.com

3. RabbitMQ
https://www.rabbitmq.com

4. Or-Tools
https://developers.google.com/optimization/introduction/overview

5. ReactJs
https://reactjs.org

6. Material UI
https://material-ui.com

7. Flutter
https://flutter.dev

8. Flask
https://flask.palletsprojects.com/en/2.0.x

9. Asp.Net Core 5.0
https://docs.microsoft.com/en-us/aspnet/core/?view=aspnetcore-5.0

10. Microsoft Azure
https://azure.microsoft.com/en-us

11. Heroku
https://www.heroku.com

12. AWS Amplify
https://aws.amazon.com/amplify/hosting

13. Face Recognition
https://pypi.org/project/face-recognition

14. Dlib
http://dlib.net