|  | **MINISTRY OF EDUCATION AND TRAINING** |
| --- | --- |

| **FPT UNIVERSITY** |
| --- |
| Capstone Project Document |
| Furniture repair and maintenance service |

| **GFA22SE01** | |
| --- | --- |
| **Group Members** | Nguyễn Tấn Sang - Team leader - SE130221  Nguyễn Tấn Sang - Team member - SE140026  Trương Hoàng Huy - Team member - SE140172  Trần Hoàng Nam - Team member - SE130257 |
| **Supervisor** | Trần Thanh Nguyên |
| **Ext Supervisor** | N/A |
| **Capstone Project code** | FA22SE58 |

- Ho Chi Minh, December/2022 -

**Table of Contents**

[***Acknowledgement***](#_heading=h.30j0zll) ***7***

[***Definition and Acronyms***](#_heading=h.1fob9te) ***7***

[***I. Project Introduction***](#_heading=h.3znysh7) ***8***

[**1. Overview**](#_heading=h.2et92p0) **8**

[1.1 Project Information](#_heading=h.tyjcwt) 8

[1.2 Project Team](#_heading=h.8c7r6bfr6bvg) 8

[**a. Supervisor**](#_heading=h.1t3h5sf) **8**

[b. Team member](#_heading=h.mlzqb3rotl5e) 8

[2. Product Background](#_heading=h.mvk535ritcrl) 8

[**3. Existing Systems**](#_heading=h.4d34og8) **8**

[3.1 CÔNG TY TNHH VĨNH SANG](#_heading=h.erokjssjkobp) 8

[3.2 Tan Tam](#_heading=h.qrkew7dcetrd) 9

[**4. Business Opportunity**](#_heading=h.2s8eyo1) **9**

[**5. Software Product Vision**](#_heading=h.17dp8vu) **9**

[**6. Project Scope & Limitations**](#_heading=h.3rdcrjn) **9**

[***II. Project Management Plan***](#_heading=h.26in1rg) ***10***

[**1. Overview**](#_heading=h.lnxbz9) **10**

[1.1 Scope & Estimation](#_heading=h.35nkun2) 10

[1.2 Project Objectives](#_heading=h.yjkymwkx7ih8) 13

[● Timelines: 90%](#_heading=h.vt0hksnw9dek) 13

[1.3 Project Risks](#_heading=h.44sinio) 13

[**2. Management Approach**](#_heading=h.2jxsxqh) **14**

[2.1 Project Process](#_heading=h.z337ya) 14

[2.2 Quality Management](#_heading=h.3j2qqm3) 15

[2.3 Training Plan](#_heading=h.parpp42uk7ld) 15

[**3. Project Deliverables**](#_heading=h.1y810tw) **16**

[**4. Responsibility Assignments**](#_heading=h.4i7ojhp) **16**

[5. Project Communications](#_heading=h.w7kr4aq4ughx) 17

[**6. Configuration Management**](#_heading=h.2xcytpi) **18**

[6.1 Document Management](#_heading=h.1ci93xb) 18

[Google Drive is used to manage project’s documents.](#_heading=h.p8549toacwyj) 18

[6.2 Source Code Management](#_heading=h.3whwml4) 18

[6.3 Tools & Infrastructures](#_heading=h.2bn6wsx) 18

[***III. Software Requirement Specification***](#_heading=h.qsh70q) ***19***

[1. Product Overview](#_heading=h.3as4poj) 19

[2. User Requirements](#_heading=h.kblghywpjhc2) 20

[2.1 Actors](#_heading=h.huwv3krjlghy) 20

[2.2 User Case](#_heading=h.zf05w7m6ieay) 20

[**3. Functional Requirements**](#_heading=h.49x2ik5) **23**

[3.1 System Functional Overview](#_heading=h.2p2csry) 23

[3.1.1 Screen flow](#_heading=h.e4wqh7yyp5md) 23

[3.1.2 Screen Description](#_heading=h.g39xpulgink5) 24

[3.1.3 Screen Authorization](#_heading=h.nstv8ol9x1y) 28

[3.1.4 Entity Relationship Diagram](#_heading=h.xy9mxbiknzaq) 30

[3.2 Order](#_heading=h.xi51fs5kes8n) 30

[3.2.1 Order Screen](#_heading=h.sdkgndhbv936) 30

[3.2.2 Order detail screen](#_heading=h.qcr19xz34hv4) 31

[3.2.2 Assign staff](#_heading=h.9t8cvvhqgvcg) 31

[3.2.3 Assign staff](#_heading=h.fl66g5cmu1fg) 32

[3.3 Customer](#_heading=h.ywk7yp57aa5) 32

[3.3.1 Customer screen](#_heading=h.tpi71fj8e1c1) 32

[3.3.2 Customer detail screen](#_heading=h.ggz4jyr3xa80) 33

[3.3.3 Order history screen](#_heading=h.c71kd3b769ik) 33

[3.4 Staff](#_heading=h.hzs334romfur) 34

[3.4.1 Staff screen](#_heading=h.xgpn6a7pg83o) 34

[3.4.1 Staff day off screen](#_heading=h.rslv4eh4igrf) 34

[3.5 Login screen](#_heading=h.94xas7jvooda) 35

[3.6 Logout screen](#_heading=h.21k0ejd7pav7) 35

[**4. Non-Functional Requirements**](#_heading=h.147n2zr) **36**

[4.1 External Interfaces](#_heading=h.3o7alnk) 36

[UI-1: The user interfaces must be simple, friendly.](#_heading=h.23ckvvd) 36

[UI-2: Icons used must be easy to recognize.](#_heading=h.xz9tykynjzxo) 36

[UI-3: Language used in the web and mobile application is Vietnamese.](#_heading=h.crx9ozggz454) 36

[**4.2.1 Usability**](#_heading=h.ihv636) **36**

[US-1: The mobile application for the customer should be simple and easy to use.](#_heading=h.haxlz5p3o28h) 36

[US-1: The mobile application for the staff should be simple and easy to use.](#_heading=h.5suznb7dlo09) 36

[US-2: The web manager for the manager should be easy to use and have an eye-catching user interface.](#_heading=h.m232k8hcv86m) 36

[US-3: The web application for the manager should require no more than one day of training to be used.](#_heading=h.w03w0l5vdyfq) 36

[US-4: Staff are not required to have technical knowledge before using.](#_heading=h.zc04s0ri7vas) 36

[4.2.2 Reliability](#_heading=h.sw98m4lq1l2w) 36

[5. Requirement Appendix](#_heading=h.9685ag2rr0yn) 36

[5.1 Business Rules](#_heading=h.32hioqz) 36

[5.2 Common Requirements](#_heading=h.9r9oxcremcc6) 39

[5.3 Application Messages List](#_heading=h.m4yxagyoiob7) 39

[5.4 Other Requirements…](#_heading=h.jm7dmwo1lz9) 39

[***IV. Software Design Description***](#_heading=h.1hmsyys) ***40***

[**1. System Design**](#_heading=h.41mghml) **40**

[1.1 System Architecture](#_heading=h.2grqrue) 40

[1.2 Package Diagram](#_heading=h.vx1227) 41

[2. Database Design](#_heading=h.3fwokq0) 42

[2.1. Table Role](#_heading=h.2iwr22oeufwz) 42

[2.2. Table Account](#_heading=h.ysbvbbo2hgal) 42

[2.3. Table Manager](#_heading=h.9e13pww7u2py) 42

[2.4. Table Customer](#_heading=h.w0fnt9iameof) 43

[2.5. Table Employee](#_heading=h.kujtgzbxa7es) 43

[2.6. Table Customer Address](#_heading=h.tiu80nu7pct8) 43

[2.7. Table Assign](#_heading=h.uvdczp8tykxt) 43

[2.8. Table Order](#_heading=h.oag8jy403cuo) 43

[2.9. Table Working Status](#_heading=h.l97k8aslx53d) 44

[2.10. Table Order Service](#_heading=h.vj2v9058us0c) 44

[2.11. Table Order Image](#_heading=h.kna98y8tjob2) 44

[2.12. Table Specialty](#_heading=h.32xj99asi8mg) 44

[2.13. Table Service](#_heading=h.b2jixyl28za) 44

[2.14. Table Service Detail](#_heading=h.sb037bxhtvgn) 45

[2.15. Table Category](#_heading=h.fqlvyeasins2) 45

[2.16. Table Employee Day Off](#_heading=h.7aozq626m7nz) 45

[3. Detailed Design](#_heading=h.lo68ipylo91z) 46

[3.1.1 Class Diagram](#_heading=h.tver3mw3n8ng) 46

[3.1.2. Account](#_heading=h.iv8649a0epie) 46

[3.1.3. Customer](#_heading=h.vuvl04om0aaq) 47

[3.1.4. Order](#_heading=h.sij01pe829zj) 47

[3.1.5. Order Service](#_heading=h.rbrtkfo5io9b) 48

[3.1.6. Working Status](#_heading=h.nmp77k9hbtab) 48

[3.1.7. Service](#_heading=h.kmywsaoc494u) 49

[3.1.8. Category](#_heading=h.nwx9wa82mkb0) 49

[3.1.9. Assign](#_heading=h.22swfpj3snzf) 50

[3.1.10. Manager](#_heading=h.phqqeotxwgq4) 50

[3.1.11. Employee](#_heading=h.7ufawoe08v4) 51

[3.1.12. Order](#_heading=h.5ukb4cp8vjot) 51

[3.2 Sequence diagram](#_heading=h.u91j22tgi9b0) 52

[**V. Software Testing Documentation**](#_heading=h.yb4b21q5iiu6) **56**

[1. Scope of Testing](#_heading=h.19c6y18) 56

[2. Test Strategy](#_heading=h.3tbugp1) 56

[**2.1 Testing Types**](#_heading=h.28h4qwu) **56**

[2.2 Test Levels](#_heading=h.nmf14n) 57

[2.3 Supporting Tools](#_heading=h.37m2jsg) 57

[3. Test Plan](#_heading=h.kcd32xo7y3eq) 57

[3.1 Human Resources](#_heading=h.n569i2fuls1o) 57

[3.2 Test Environment](#_heading=h.o6sj963vcjuv) 57

[3.3 Test Milestones](#_heading=h.1mrcu09) 57

[**4. Test Cases**](#_heading=h.46r0co2) **58**

[5. Test Reports](#_heading=h.2dzlwk7ah474) 58

[5.1 Unit Test](#_heading=h.97tz6vrx133z) 58

[5.2 Test Documents](#_heading=h.xw04fzc21gkr) 59

[***VI. Release Package & User Guides***](#_heading=h.111kx3o) ***59***

[1. Deliverable Package](#_heading=h.3l18frh) 59

[1.1 Source codes & documents](#_heading=h.1euxkys) 59

[1.2 Known Issues, Limitations & Restrictions](#_heading=h.t55o27) 59

[2. Installation Guides](#_heading=h.irzpyd4jy4sv) 60

[2.1 System Requirements](#_heading=h.4k668n3) 60

[a. For PC](#_heading=h.4c9qzlm) 60

[b. For Android](#_heading=h.16kbk18) 60

[2.2 Installation Instruction](#_heading=h.2zbgiuw) 60

[a. Mobile](#_heading=h.1jzhg09) 60

[b. Front End](#_heading=h.2j4f8vv) 60

[c. Back End](#_heading=h.y9pj3o) 61

[3. User Manual](#_heading=h.1egqt2p) 61

[3.1 Overview](#_heading=h.3ygebqi) 61

[3.2 Assign Staff](#_heading=h.2dlolyb) 61

[3.3 Approved order](#_heading=h.yfogdppq67uf) 63

[3.4 Staff send day off request](#_heading=h.da9jjczhcsbt) 64

[3.5 Staff fulfil order](#_heading=h.nb2qk3z5otmx) 66

[3.6 Customer create a booking](#_heading=h.msh2ed68g79f) 68

# Acknowledgement

The team would like to express the individuals and groups who have always accompanied and helped our team to complete the product.

We would like to express our supervisor, Mr. Tran Thanh Nguyen. He is the venerable teacher in FPT University. Without his instruction and guidance through the project development process, we thought this project would never have turned out as it did.

We would like to say thanks to FPT University, teachers and friends who have always dedicated and helped us to achieve the best result.

# Definition and Acronyms

| **Acronym** | **Definition** |
| --- | --- |
| PWM | Psychology website |
| BA | Business Analysis |
| 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: Furniture repair and maintenance service
* Project code: FMS
* Group name: Furniture services team
* Software type: Website & Android Application

### 1.2 Project Team

#### a. Supervisor

| **Full Name** | **Email** | **Phone Number** | **Title** |
| --- | --- | --- | --- |
| Trần Thanh Nguyên | nguyentt15@fe.edu.vn | 0967863063 | Lecturer |

*Table 2. Supervisor*

#### b. Team member

| **Full Name** | **Email** | **Mobile** | **Role** |
| --- | --- | --- | --- |
| Nguyễn Tấn Sang | sangntse130221@fpt.edu.vn | 0938995710 | Leader |
| Nguyễn Tấn Sang | sangntse140026@fpt.edu.vn | 0973457541 | Member |
| Trương Hoàng Huy | huythse140172@fpt.edu.vn | 0931415154 | Member |
| Trần Hoàng Nam | namthse130257@fpt.edu.vn | 0396258901 | Member |

*Table 3. Team members*

## 2. Product Background

Today, with the hectic pace of life in big cities, equipping a spacious house with interior equipment to ensure sufficient use is what most families are interested in. However, in the process of using these devices, if they are degraded or damaged and need to be maintained and repaired, it is a difficult problem for every family. How to find a reputable unit, skilled experts in this field and make sure to keep the equipment with high aesthetics. Furthermore, people are at high risk of getting unreasonable prices for materials, replacement parts and labour cost for their furniture service.

## 3. Existing Systems

### 3.1 CÔNG TY TNHH VĨNH SANG

Vinh Sang is a company that provides furniture repair services with its own website. The company provides a full range of services related to furniture repair. This company allows customers to book appointments.

Link website: <http://suanoithat.com/>

Main features: book appointment, pricing, news, online chat

System actors: admin, guest, customer

Pros: Provide a good interface that is easy to create sympathy for users. The components related to furniture repair services such as service quotes, appointment booking are easy to access and easy to use. Appointment booking is done quickly, no login required. When completing the booking procedure, there will be an SMS confirming the booking is successful.

Cons: There is no notification that the time slot has been set, which can lead to multiple customers booking the same time slot. The pricing action directs the user to the service price articles or the product page. This system doesn’t allow customers to make purchases and make payments through the website.

### 3.2 Tan Tam

Tan Tam Center is a system that provides home appliance and furniture repair services with its own website and mobile application. The system provides repair services at home.

Link website:  [https://www.dichvutantam.com/](https://www.dichvutantam.com/bang-gia)

Mobile application:  Thợ Tận Tâm

Main features: book appointment, pricing, online chat, news

System actors: admin, staff, guest, customer

Pros: Provide home repair services. Eye-catching and easy-to-use interface. Users are not required to log in to book an appointment.

Cons: Offer a very short range of furniture services. Can not prevent customers from getting high prices for material from staff.

## 4. Business Opportunity

This offers the opportunity to build a system where a customer can track the status, manage the details of invoices, as well as track the number of orders, services, staff, and manager responsibility in one receipt. Furthermore, customers can view the exact price of the receipt after the staff conducts a survey. This system can protect the customer's benefit about the trustworthy price when using repairing or maintenance service. Prevent staff with bad intentions. Provide the possibility of cost savings through discount vouchers.

## 5. Software Product Vision

The Furniture Repair and Maintenance Service System was born to support and serve to meet the diverse requirements of users. Focusing on giving reasonable service prices and quality-assured materials. Unlike the current telephone and manual service ordering processes, customers who use the Furniture Repair and Maintenance Service System will not have to worry about price reasonableness of their order and could easily be provided with after-sales service.

## 6. Project Scope & Limitations

Project scope:

* Support company manage their staff, services and booking with web app
* Allow staff assign their day-off schedule
* Customer can use mobile app for booking services from company

Limitations:

* The system does not support multiple languages
* The system hasn’t supported online payment now. We will develop it in the future
* The system hasn’t supported multiple brands of company. We will develop it in future
* The system hasn’t supported map for customer and staff for searching
* The system hasn’t supported OTP verify

# II. Project Management Plan

## 1. Overview

### 1.1 Scope & Estimation

| **#** | **WBS Item** | **Complexity** | **Est. Effort**  **(man-days)** |
| --- | --- | --- | --- |
| ***1*** | ***Initiation*** |  | ***10*** |
| 1.1 | Prepare Report 1 – Project Introduction | Complex | 10 |
| 1.2 | Deliver Report 1 | Simple | 0 |
| ***2*** | ***Planning & Requirement*** |  | ***45*** |
| 2.1 | Prepare Report 2 – Project Management Plan | Complex | 10 |
| 2.2 | Deliver Report 2 | Simple | 0 |
| 2.3 | Requirement analysing | Complex | 10 |
| 2.4 | Design User Interfaces (UI) prototypes | Complex | 15 |
| 2.5 | Write SRS (Software Requirements Specification) | Complex | 10 |
| 2.6 | Deliver Report 3 | Simple | 0 |
| ***3*** | ***Software Designing*** |  | ***35*** |
| 3.1 | Write SDD (Software Design Document) | Complex | 10 |
| 3.2 | Deliver Report 4 – Software Design Document | Simple | 0 |
| 3.3 | Prepare Test Report | Complex | 10 |
| 3.4 | Deliver Report 5 – Software Testing Document | Medium | 5 |
| 3.5 | Deliver Report 6 – User Guides | Medium | 5 |
| 3.6 | Deliver Report 7 – Final Project Report | Simple | 5 |
| ***4*** | ***Database*** |  | ***31*** |
| 4.1 | Design ERD | Complex | 10 |
| 4.2 | Implement database | Complex | 10 |
| 4.3 | Authorization and Authentication | Medium | 5 |
| 4.3.1 | Login for all role | Medium | 5 |
| 4.3.2 | Logout | Simple | 1 |
| ***5*** | ***Implement*** |  |  |
| ***5.1*** | ***Customer application*** |  | ***46*** |
| 5.1.1 | Login | Medium | 5 |
| 5.1.2 | Register | Medium | 6 |
| 5.1.3 | Add new address | Medium | 5 |
| 5.1.4 | Update address | Simple | 3 |
| 5.1.5 | View home page | Simple | 3 |
| 5.1.5 | View history | Medium | 5 |
| 5.1.6 | View profile screen | Simple | 3 |
| 5.1.7 | View date and time screen | Simple | 3 |
| 5.1.8 | Create booking | Medium | 5 |
| 5.1.9 | Change password | Simple | 2 |
| 5.1.10 | View payment methods | Simple | 2 |
| 5.1.11 | Choose address | Simple | 1 |
| 5.1.12 | View list category | Simple | 3 |
| 5.1.13 | Delete the address | Simple | 2 |
| ***5.2*** | ***Staff Application*** |  | ***19*** |
| 5.2.1 | Login | Simple | 1 |
| 5.2.2 | Logout | Simple | 1 |
| 5.2.3 | Change password | Medium | 2 |
| 5.2.4 | View booking list | Simple | 1 |
| 5.2.5 | View history | Simple | 1 |
| 5.2.6 | View profile screen | Simple | 1 |
| 5.2.7 | Send day-off request | Medium | 2 |
| 5.2.8 | View day-off request status | Simple | 1 |
| 5.2.9 | Send detailed report | Simple | 1 |
| 5.2.10 | View booking details | Simple | 1 |
| 5.2.11 | Add services into a booking | Medium | 2 |
| 5.2.12 | Delete services of a booking | Complex | 5 |
| ***5.3*** | ***Manager Web Application*** |  | ***52*** |
| 5.3.1 | Login | Simple | 3 |
| 5.3.2 | View Order page | Simple | 3 |
| 5.3.3 | View List Order | Simple | 1 |
| 5.3.4 | View Order Status | Simple | 3 |
| 5.3.5 | Assign Work | Complex | 7 |
| 5.3.6 | View Staff Page | Simple | 3 |
| 5.3.7 | View List Staff | Simple | 3 |
| 5.3.8 | View Staff DayOff | Simple | 3 |
| 5.3.9 | Approve Staff DayOff | Medium | 5 |
| 5.3.10 | Deny Staff DayOff | Medium | 5 |
| 5.3.11 | View Customer Page | Simple | 3 |
| 5.3.12 | View List Customer | Simple | 3 |
| 5.3.13 | View Customer Detail | Medium | 5 |
| 5.3.14 | View Customer Order history | Medium | 5 |
| ***Total Estimated Effort (man-days)*** | | | ***240*** |

### 1.2 Project Objectives

### Timelines: 90%

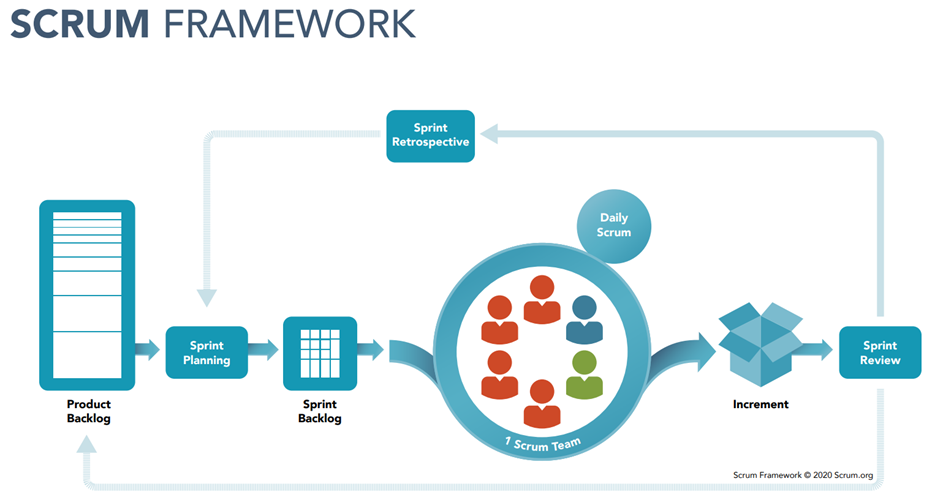
* Allocated Effort: 4(member) \* 4(months) \* 20(days/month) = 320 man-days.
* Estimated total cost: N/A

### 1.3 Project Risks

| **#** | **Risk Description** | **Impact** | **Possibility** | **Response Plans** |
| --- | --- | --- | --- | --- |
| 1 | Team member drop out | Critical | Low | Reduce scope. |
| 2 | Server issue | Medium | Low | Change the other server. |
| 3 | Requirements change, requirements unexpected changes | Critical | Medium | Make some solution to resolve them. |
| 4 | Technology risk | High | Medium | Research or some help from a mentor. |

## 2. Management Approach

### 2.1 Project Process



The project is developed using the Scrum model which is an agile framework for developing, delivering and maintaining products. Our team chooses this model for the following reasons:

● Because Scrum is suitable for small and medium-sized projects, it fits our team which consists of 4 members well.

● For this project, each sprint spans 1 week and the duration of our product backlog is 12 sprints because we need to deliver the product quickly and the customer wants to review the product every week.

● Flexible to requirement changes.

**References**: *https://www.scrum.org/resources/what-is-scrum*

### 2.2 Quality Management

To improve project quality, team will follow these approaches:

● One member will be responsible for writing API Testing on Postman.

● Making and following the coding rules, coding convention.

● Code review before code is merged, deployed.

● Holding daily meetings to follow project’s tasks and support each other.

### 2.3 Training Plan

| Training Area | Participants | When, Duration | Waiver Criteria |
| --- | --- | --- | --- |
| Git, Github | All member | Week 1, 2 days | Mandatory |
| Trello | All member | Week 2, 1 days | Mandatory |
| Figma Tool | All member | Week 2, 1 days | Mandatory |
| StarUml, Draw IO | All member | Week 2, 2 days | Mandatory |
| C# | Nguyễn Tấn Sang | Week 2-3, 5 days | Mandatory |
| ReactJS | Trần Hoàng Nam | Week 2-3, 5 days | Mandatory |
| Flutter | Nguyễn Tấn Sang, Trương Hoàng Huy | Week 2-3, 7 days | Mandatory |

## 3. Project Deliverables

| **#** | **Deliverable** | **Due Date** | **Notes** |
| --- | --- | --- | --- |
| 1 | Project Plan document | 05/09/2022 | Overall project plan |
| 2 | SRS documents | 26/09/2022 | Software requirement specification |
| 3 | Design documents | 10/10/2022 | Architecture Design, Detailed design, Database, UI design |
| 4 | Software Testing Document | 24/10/2022 | Testing Reports |
| 5 | Guides documents | 07/11/2022 | Architecture Design, Detailed design, Database, UI design |
| 6 | Customer application, Staff application | 21/11/2022 | Code, System test case |
| 7 | Web manager application | 28/11/2022 | Code, System test case |
| 8 | Web admin application | 05/12/2022 | Code, System test case |

## 4. Responsibility Assignments

| **Responsibility** | **SangNTSE130221** | **SangNTSE140026** | **Huythse140172** | **NamTHSE130257** |
| --- | --- | --- | --- | --- |
| Project Planning & Tracking | D | D | R | R |
| Prepare Project Introduction Document | D | D | D | D |
| Prepare SRS Document (Overview Part) | D | D | S | S |
| Prepare SRS Document (User Requirements) | D | R | S | S |

## 5. Project Communications

## 

| **Communication Item** | **Who/ Target** | **Purpose** | **When, Frequency** | **Type, Tool, Method(s)** |
| --- | --- | --- | --- | --- |
| Daily Meeting | Team Member | · Report task progress  · Plan tasks for the day  · Discuss and solve problems | Every working day | Google Meet |
| Sprint planning | Team Member | · Define what can be delivered in the sprint  · Determine how that work will be achieved | The fine day of a new sprint: Monday morning | Google Meet, Offline |
| Sprint review | Supervisor and Team Member | · Demo function completed in sprint  · Describe the work they’ve done | At the end of the sprint: Saturday afternoon | Offline |

## 6. Configuration Management

### 6.1 Document Management

### Google Drive is used to manage project’s documents.

### 6.2 Source Code Management

GitHub is used to manage project’s source code.

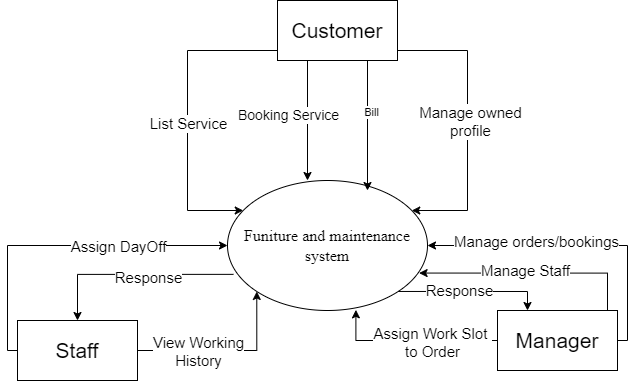
### 6.3 Tools & Infrastructures

| **Category** | **Tools / Infrastructure** |
| --- | --- |
| **Technology** | ReactJS (Front-end), C# (Back-end), Flutter (Mobile). |
| **Database** | SQL Server |
| **IDEs/Editors** | Visual Studio Code, Android studio, Visual Studio. |
| **Diagramming** | StarUML, DrawIO |
| **Documentation** | Ms Office, Google Docs/Sheets/Slides |
| **Version Control** | GitHub (Source Codes), Google Drive (Documents) |
| **Deployment server** | Smarter net |
| **Project management** | Trello |

# III. Software Requirement Specification

## 1. Product Overview

The product aims to connect Furniture repair and maintenance service with users. We do not advocate directly competing with the chain stores that have developed but mainly targeting small stores with the customer segment more suitable for our system.



*Figure 7. Product overview*

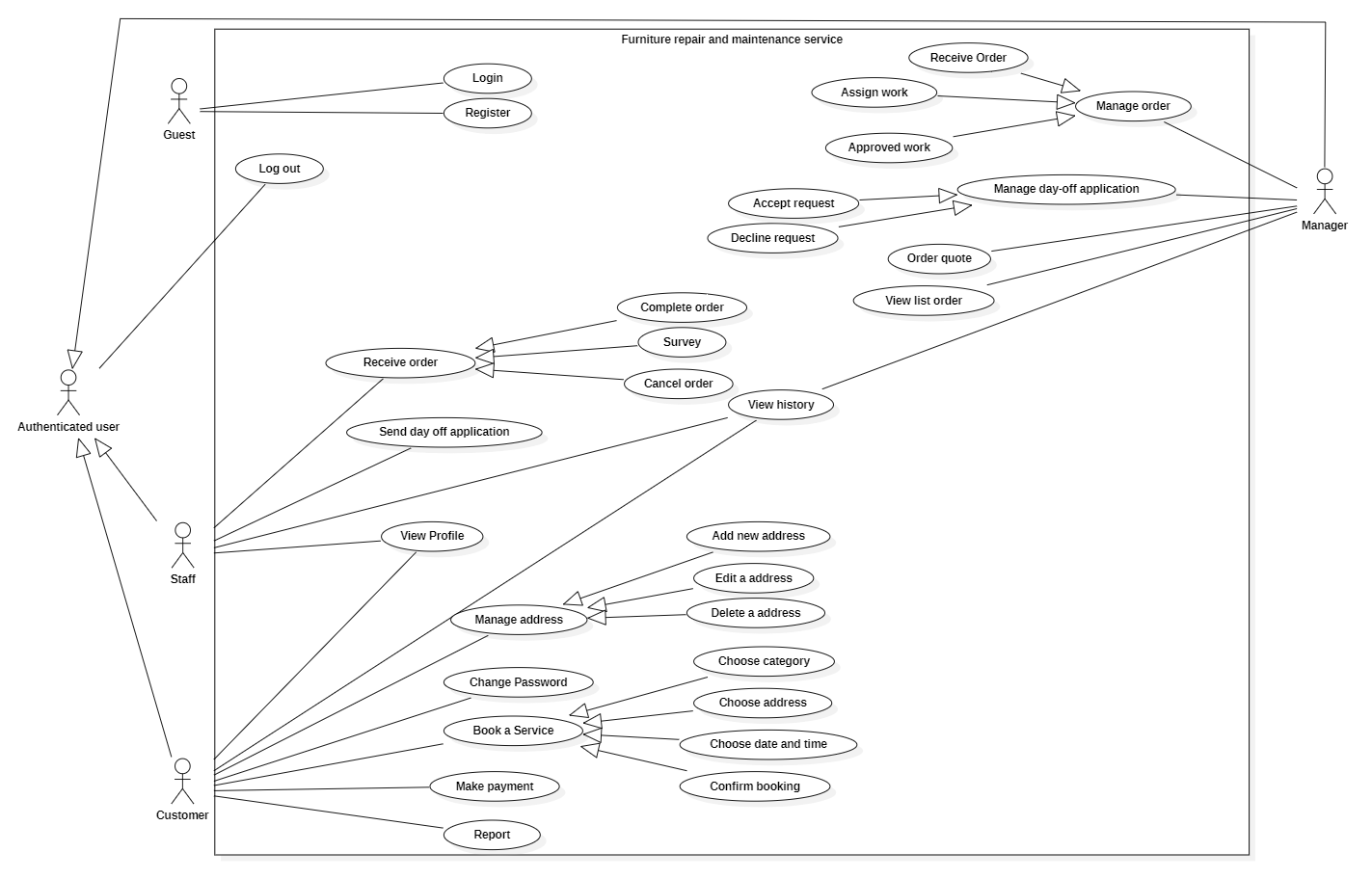
## 2. User Requirements

### 2.1 Actors

| **#** | **Actor** | **Description** |
| --- | --- | --- |
| 1 | Guest | Users use FMS to log in or register. |
| 2 | Customer | The guest has been authenticated successfully. |
| 3 | Staff | receive assigned booking and also send detailed reports to manager |
| 4 | Manager | Manage the whole system. |

### 2.2 User Case

**2.2.1 Diagram(s)**



**2.2.2 Descriptions**

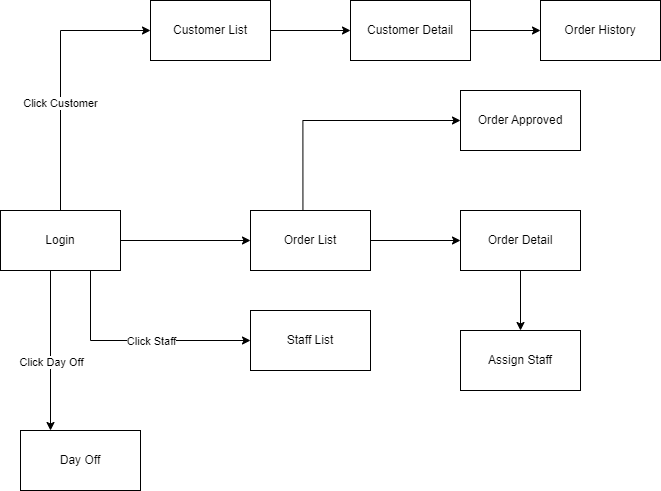
| **#** | **User Case** | **Actors** | **Use Case Description** |
| --- | --- | --- | --- |
| 01 | Login | Manager,,  Customer,  Staff. | Users will login in to the system. At this time, the system will assign permissions to each role. |
| 02 | Register | Guest | Guests can create a new account. |
| 03 | Logout | Manager,,  Customer,  Staff. | Users logout of the system. |
| 04 | View profile | Manager,,  Customer,  Staff. | Users view detailed information of their personal profile. |
| 05 | Book a service | Customer | Customers make a service booking after viewing the service. |
| 06 | View service | Customer | View all service details. |
| 07 | Make payment | Customer | Make payment for service used directly on smart mobile through QR code payment method of Momo. |
| 08 | View list booking | Manager,  Customer,  Staff. | View all booking information. |
| 09 | Cancel booking from customer’s request. | Manager,  Staff. | Make cancel booking when requested by the customer. |
| 10 | View list service booked | Manager,  Customer,  Staff. | View list service requests of customers. |
| 11 | Add new address | Customer | For some reason customers want to have a new address, the customer is allowed to add a new address. |
| 12 | Edit a address | Customer | For some reason customers want to edit an address, the customer is allowed to edit it. |
| 13 | Delete a address | Customer | For some reason customers don’t want to use the address, so customers can delete it. |
| 14 | Change password | Customer | Customers can change passwords. |
| 15 | Choose category | Customer | A list of categories for the customer to choose to create a booking. |
| 16 | Choose address | Customer | A list of address of customer to choose for creating a booking |
| 17 | Choose date and time | Customer | Customers can freely choose the date and time for creating a booking |
| 18 | Select completed service | Staff | Confirmation of completion when the service has been completed for the customer. |
| 19 | View account | Manager | View list account information. |
| 20 | Assign work | Manager | Manager assigns staff into booking according to the staff's profession. |
| 21 | Receive order | Staff | Staff receive a new assigned order from the manager. |
| 22 | Send day off request | Staff | When it is necessary to take a day off, staff can send requests to the manager. |
| 23 | Report | Customer | The Customer can send a report ticket if having an issue happen. |
| 24 | Approve day off application | Manager | Manager approves after considering the staff’ request. |
| 25 | Make field survey | Staff | Staff edit details of order after a field survey at customer location. |
| 26 | Decline day off application | Manager | Manager declines after considering the staff’ day off request. |
| 27 | Confirm day off application | Manager | Manager confirm after considering the staff’ day off request. |

## 3. Functional Requirements

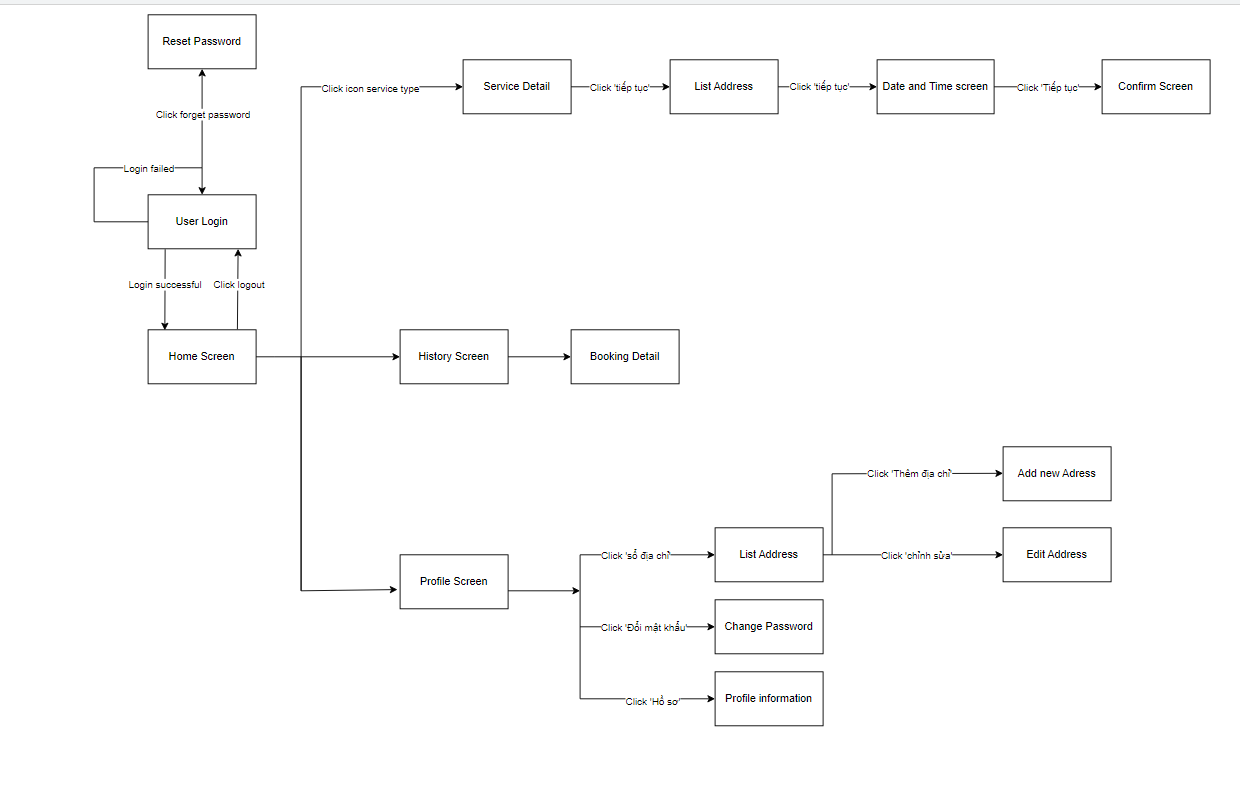
### 3.1 System Functional Overview

#### 3.1.1 Screen flow

* Manager role



* Customer role

****

* Staff role

#### 

#### 3.1.2 Screen Description

* Manager role

| **#** | **Feature** | **Screen** | **Description** |
| --- | --- | --- | --- |
| 1 | Order list | Order screen | The screen will get list order when manager check in |
| 2 | Display order by status | Order screen | The screen will get list order by status when manager select status |
| 3 | Search order | Order screen | The screen will show order search by name when manager search |
| 4 | Sort order from old to new | Order screen | The screen will get list order from old to new when manager select |
| 5 | Sort order from new to old | Order screen | The screen will get list order from new to old when manager select |
| 6 | View order detail | Order detail screen | The screen will get detail information of order when manager click view detail button |
| 7 | Assign staff | Assign staff screen | The screen will get list staff to assign |
| 8 | Confirm assign staff | Assign staff screen | After choosing staff manager will able to assign staff to order |
| 9 | Approved order | Approved order screen | Approved the order for the customer |
| 10 | Staff list | Staff screen | The screen will get staff list when manager check in |
| 11 | Customer list | Customer screen | The screen will get customer list when manager check in |
| 12 | Customer detail | Customer detail screen | The screen will show detail information of customer |
| 13 | Order history | Order history screen | The screen will show order detail when manager select order in customer detail |
| 14 | Day off | Day off screen | The screen will show list of staff day off |
| 15 | Confirm day off | Day off screen | Make day off confirm at the request of the staff |
| 16 | Cancel day off | Day off screen | Make day off cancel at the request of the staff |

* Staff role

| **#** | **Feature** | **Screen** | **Description** |
| --- | --- | --- | --- |
| 1 | Login | Login Screen | Allow users to login with username and password. |
| 2 | View profile | Profile Screen | View profile information. |
| 3 | View account information | Profile Information | View account detailed information. |
| 4 | Reset password | Reset Password | Allow users to reset with a new password. |
| 5 | View awaiting orders | Awaiting Orders | The screen will get a list order of scheduled orders received from manager. |
| 6 | View order detail | Order detail screen | The screen will get detailed information of order when staff click on the order ticket. |
| 7 | View on-going orders | On-going Orders | The screen will get a list order that is currently in progress. |
| 8 | Edit order details | Order Editing Screen | Allow staff to edit, add, delete services accordingly to real situations as well as take pictures. |
| 9 | View manager proposed order details | Manager’ Order Details | The screen will get detailed information of the order sent by the manager when staff click on the order ticket. |
| 10 | Add service to order | Add Services | The screen will get a list of services and allow staff to add into order. |
| 11 | Day off | Day-off Screen | The screen will show a list of staff days off. |
| 12 | Send day off request | Day-off Form Screen | Allow staff to fill in a day off request form and send it to the manager. |
| 13 | View day off request status | Request Details | The screen will show detailed information of sent quests and manager status on it. |
| 14 | View order history | Order History | The screen will show a list of orders in the history of staff. |
| 15 | View completed history | Completed Orders | The screen will show a list of completed orders of staff. |
| 16 | View cancelled history | Cancelled Orders | The screen will show a list of cancelled orders of staff. |
| 17 | View notifications | Notification Screen | The screen will show a list of received notifications. |

* Customer role

| **#** | **Feature** | **Screen** | **Description** |
| --- | --- | --- | --- |
| 1 | Login | Login Screen | Allow users to login with username and password. |
| 2 | View profile | Profile Screen | View profile information. |
| 3 | View account information | Profile Information | View account detailed information. |
| 4 | Reset password | Reset Password | Allow users to reset with a new password. |
| 5 | Change password | Change Password Screen | Allow user to change to the new password. |
| 6 | Create Booking | Service Detail | Allow user create a booking |
| 7 | Choose address | List Address | The screen will show a list of addresses, and the user chooses one address. |
| 8 | Choose Date and Time | Date and Time screen | Allow user to choose date and time |
| 9 | Confirm Booking | Confirm Booking screen | Allow user to view the sum of the bill and confirm that |
| 10 | View History | View History screen | The screen will show all the booking include processing, done, cancelling order |
| 11 | View Address | List Address Screen | Allow user to view list of address |
| 12 | Add new Address | Add new Address Screen | The screen will show a form for the user to fill in to create a new address. |
| 13 | Edit address | Edit Address Screen | Allow user to Edit an Address |

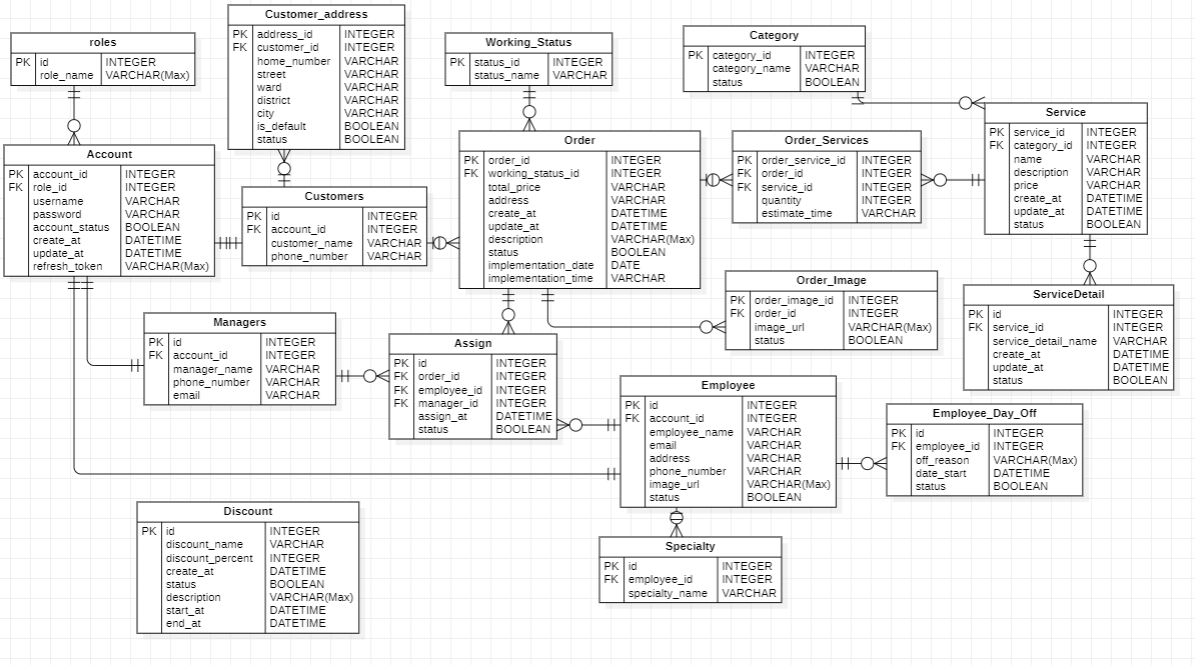
#### 3.1.3 Screen Authorization

| **Screen** | **Staff** | **Customer** | **Manager** |
| --- | --- | --- | --- |
| Order screen | x | x | X |
| Order detail screen | x | x | X |
| Assign staff screen |  |  | X |
| Approved order screen | x |  | X |
| Staff screen | x |  | X |
| Customer screen |  |  | X |
| Customer detail screen |  |  | X |
| Order history screen | x | x | X |
| Day off screen | x |  | X |
| Login screen | x | x | X |
| Reset password screen | x | x | X |
| Staff detail screen | x |  | X |
| Order editing screen | x |  | X |
| Add service screen | x |  |  |
| Day off request detail | x |  | X |
| Day off form screen | x |  |  |
| List Address screen |  | x |  |
| Add new Address screen |  | x |  |
| Change Password screen |  | x |  |

#### 

#### 

#### 3.1.4 Entity Relationship Diagram



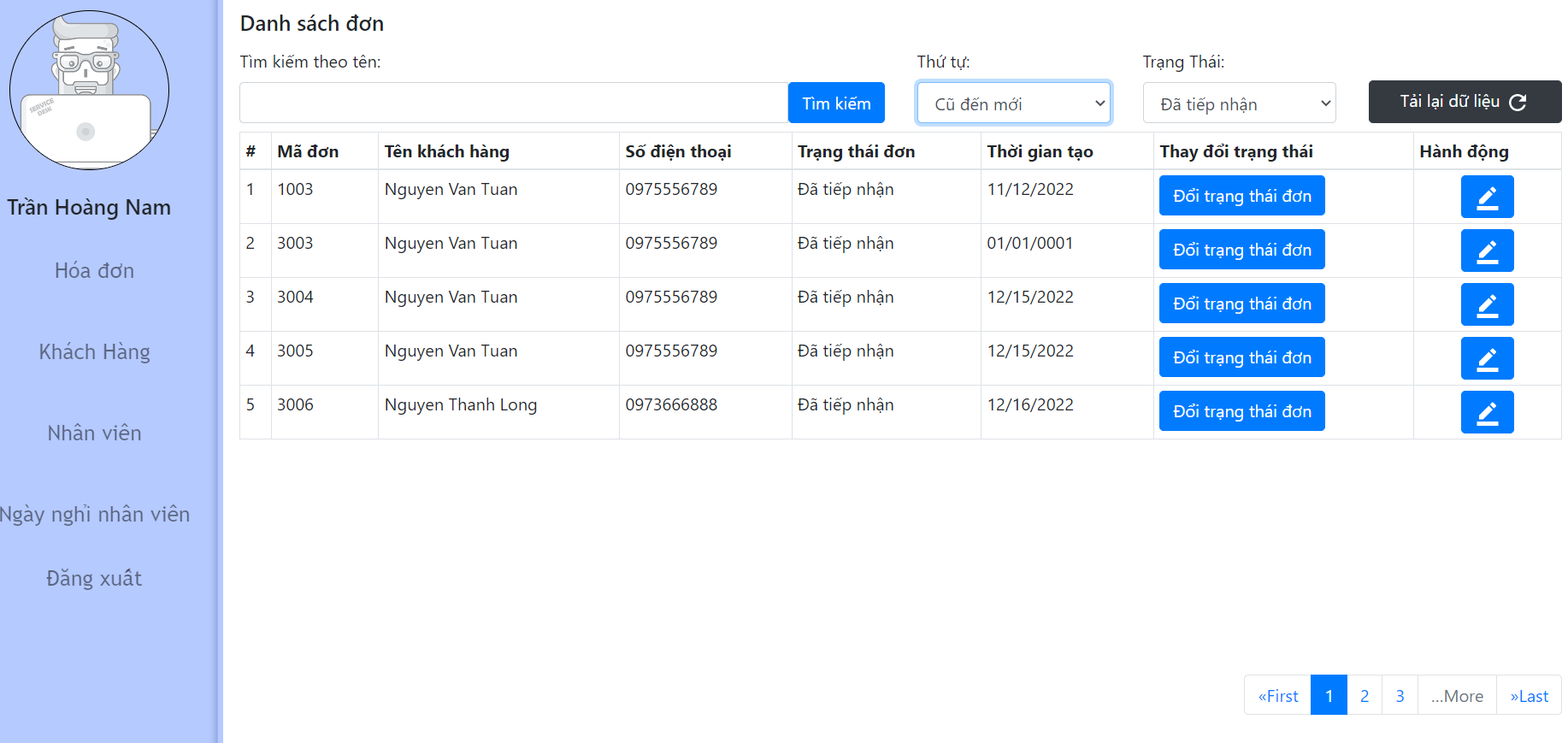
**Entity Description**

| **#** | **Entity** | **Description** |
| --- | --- | --- |
|  |  |  |
|  |  |  |

### 3.2 Order

### 3.2.1 Order Screen

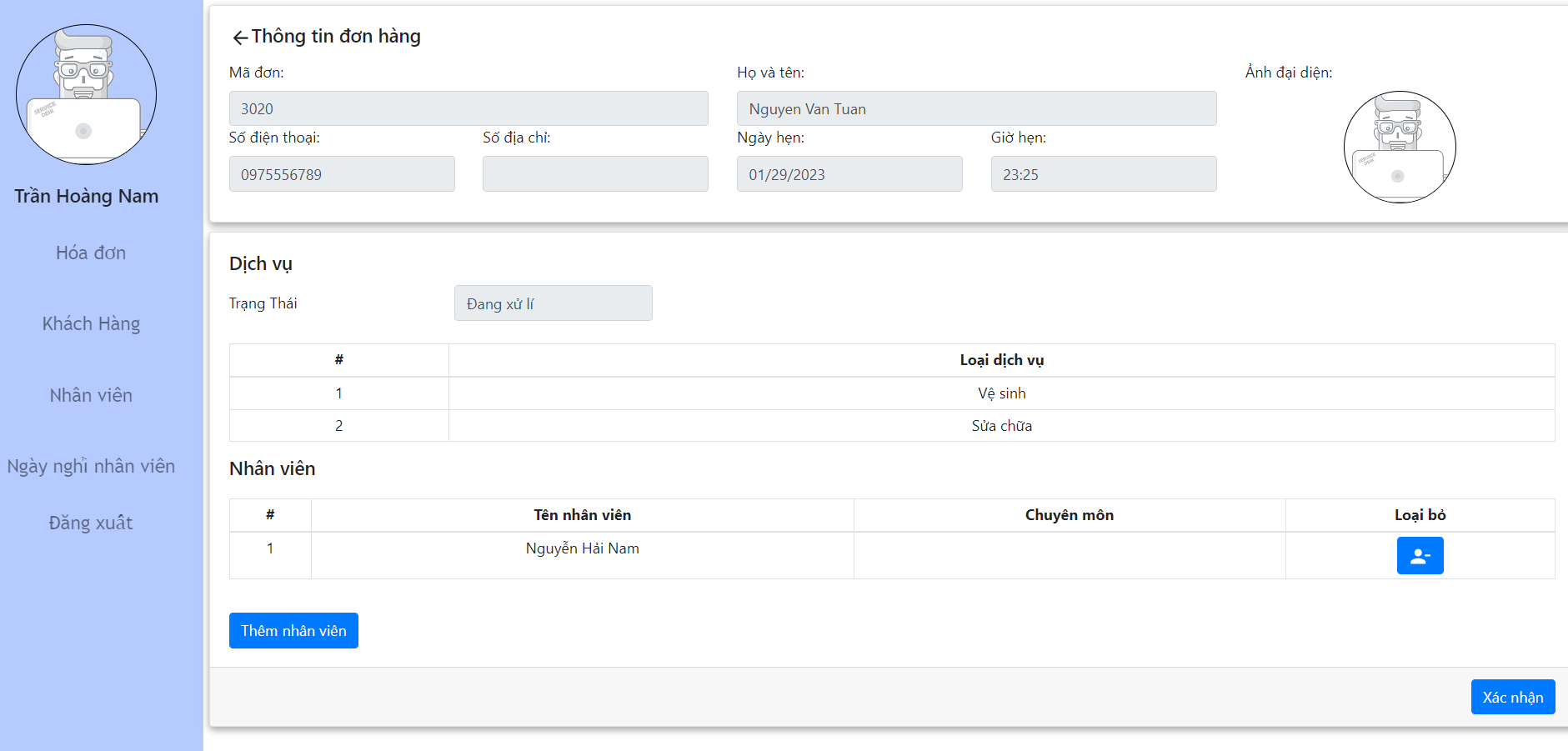
* *Function trigger:* The manager logged in
* *Actor:* manager
* *Function description:* Manager see the list of orders
* *Function detail:* List of order, manager can choose to go to other pages from this screen
* *Business rule:* BR-009, BR-025
* *Screen layout:*



### 

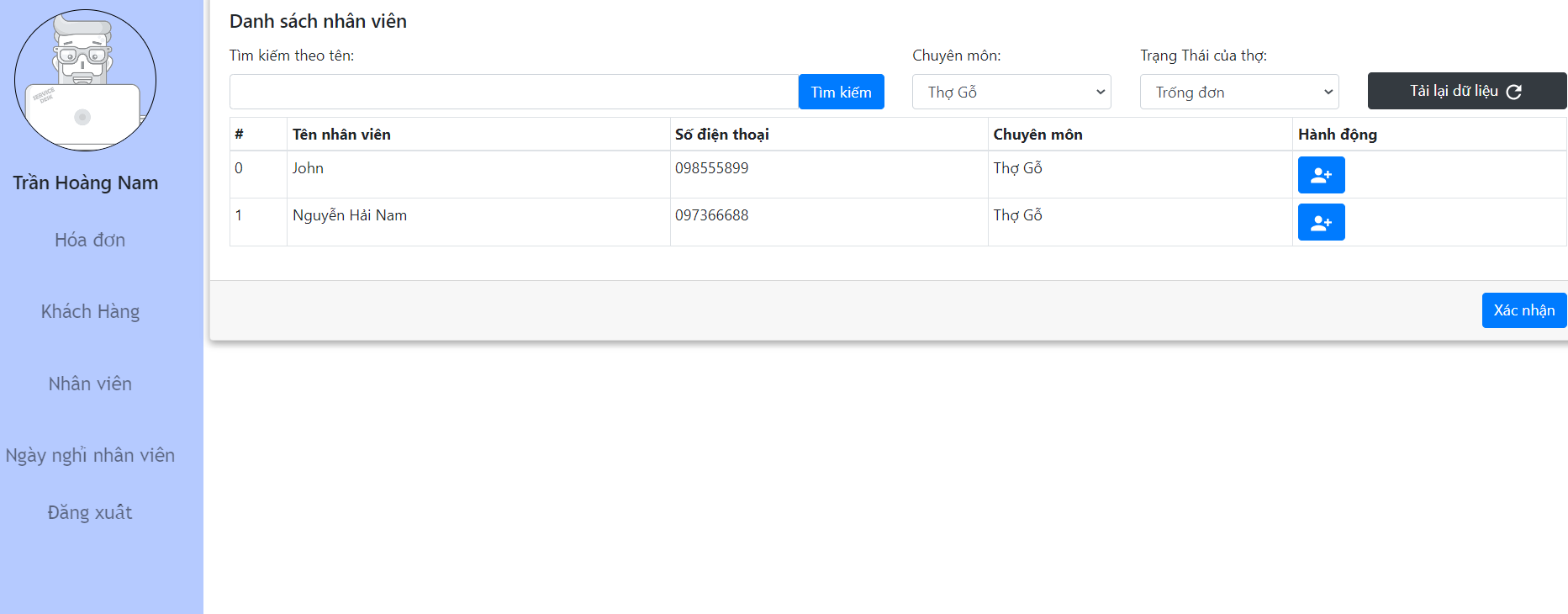
### 3.2.2 Order detail screen

* *Function trigger:* Manager want to see order detail
* *Actor:* manager
* *Function description:* Manager see the detail of the select order
* *Function detail:* Manager can see customer information, service, staff in order and assign staff
* *Business rule:* BR-056, BR-057
* *Screen layout:*

**

### 3.2.2 Assign staff

* *Function trigger:* Manager want to assign staff into order
* *Actor:* manager
* *Function description:* Manager see list of staff to assign to order
* *Function detail:* Manager can see list of staff with specialty, search staff by name
* *Business rule:* N/A
* *Screen layout:*

**

### 3.2.3 Approved order

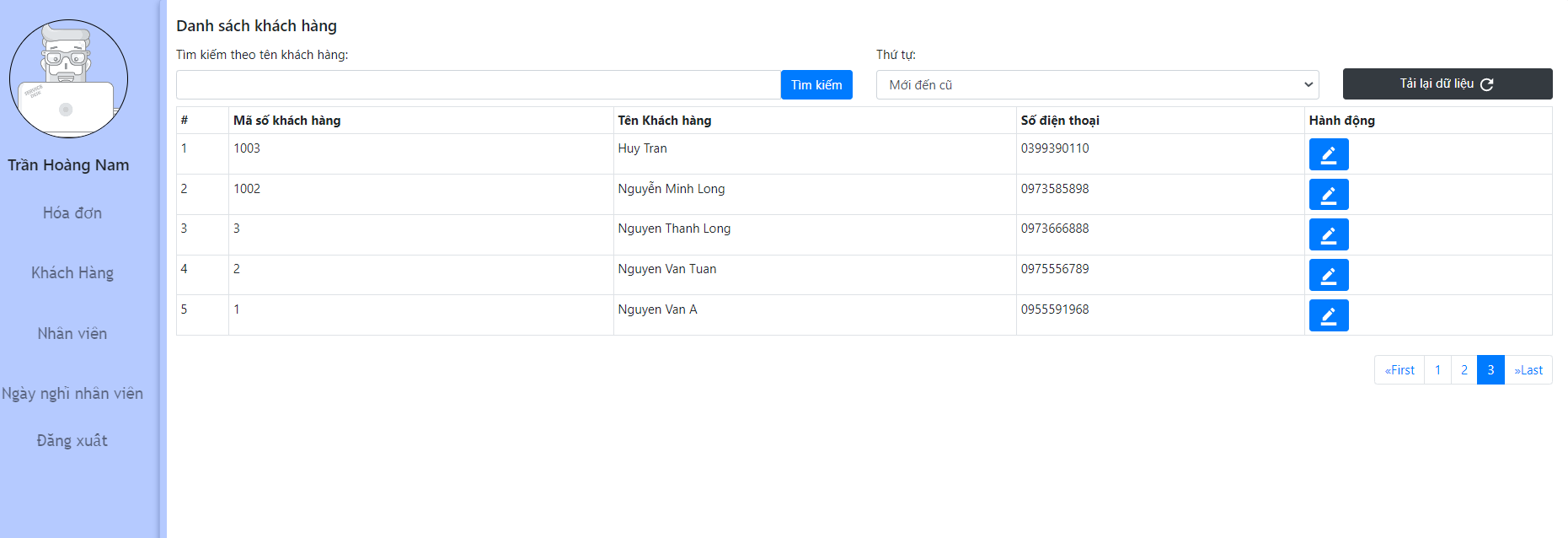
* *Function trigger:* Manager want to approved the order with status “Đang thực hiện”
* *Actor:* manager
* *Function description:* Manager want to approved order
* *Function detail:* Manager can see detail of the order to approve, list of sever, price, staff, image, and confirm approve
* *Business rule:* BR-059
* *Screen layout:*

### 

### 3.3 Customer

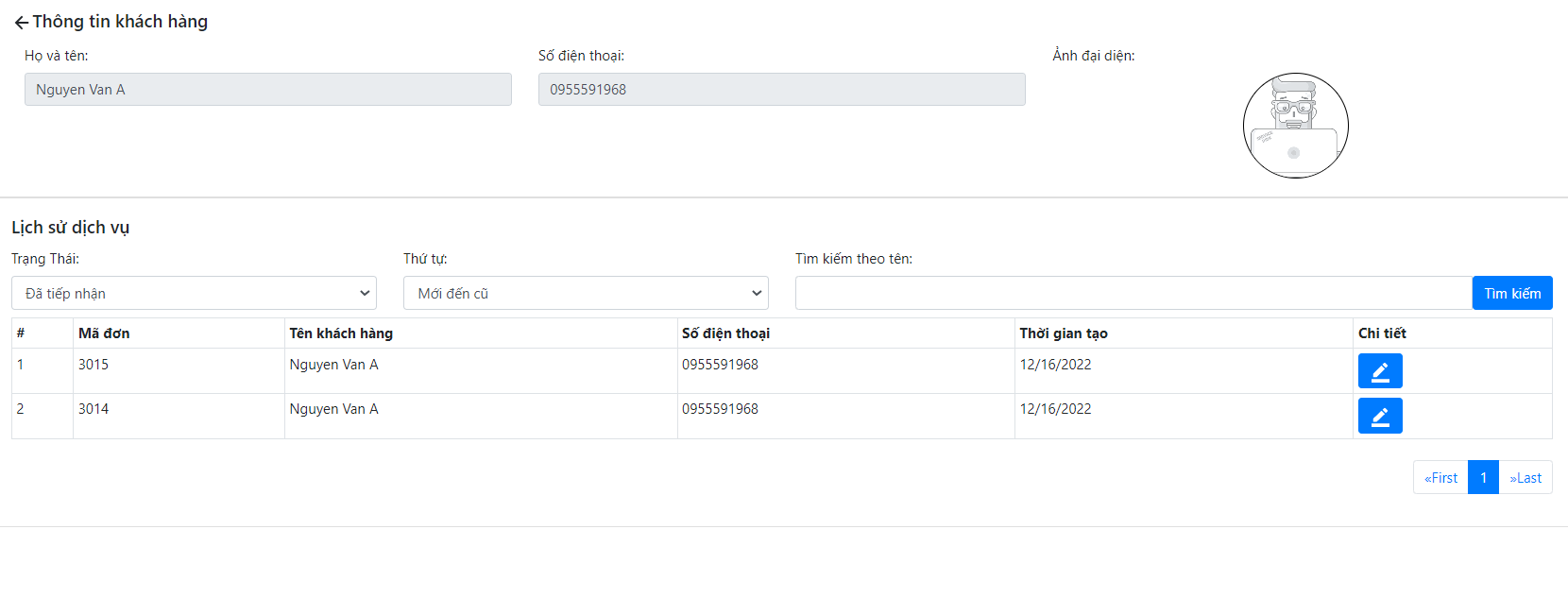
### 3.3.1 Customer screen

* *Function trigger:* Manager want to see list of customer
* *Actor:* manager
* *Function description:* Manager want to see list of customer
* *Function detail:* Manager can see list of customer with name, phone number, sort, search
* *Business rule:* BR-035
* *Screen layout:*

**

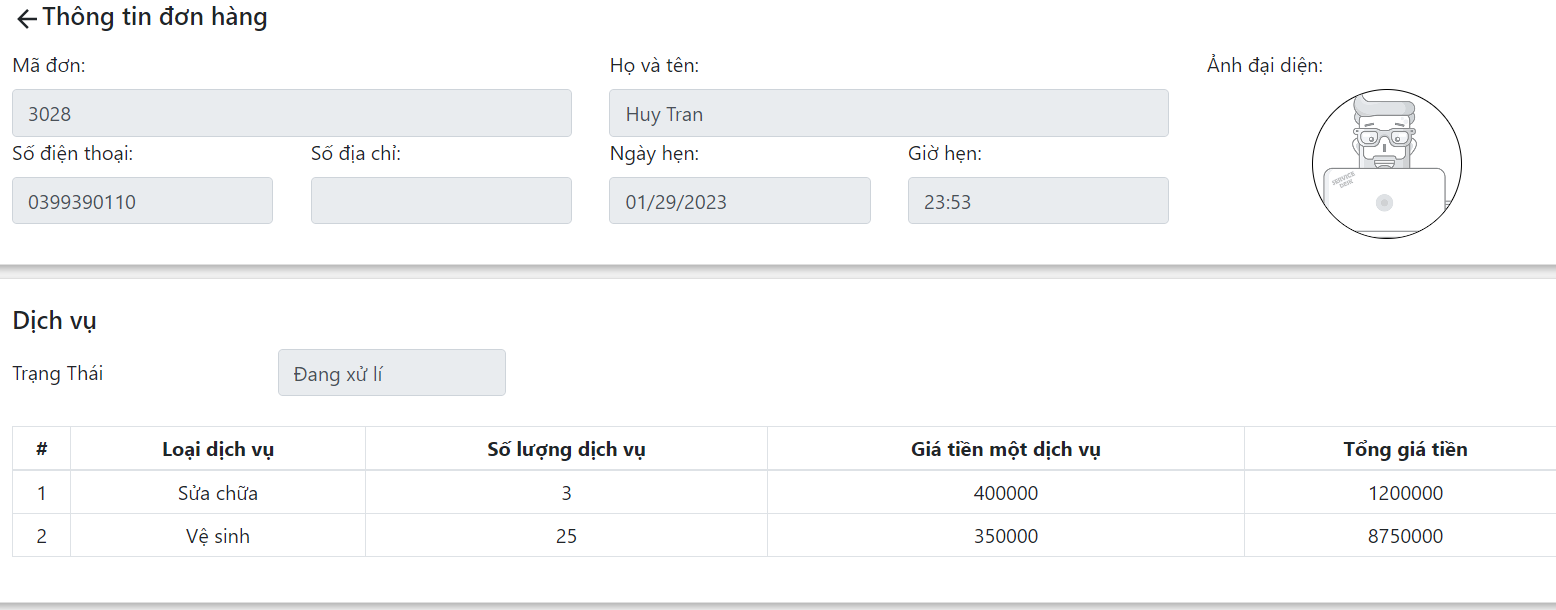
### 3.3.2 Customer detail screen

* *Function trigger:* Manager want to see customer details
* *Actor:* manager
* *Function description:* Manager want to see customer detail of selected customer
* *Function detail:* Manager can see customer phone, customer name, customer order
* *Business rule:* BR-060
* *Screen layout:*

**

### 3.3.3 Order history screen

* *Function trigger:* Manager want to see order detail of customer
* *Actor:* manager
* *Function description:* Manager want to see order detail select in customer screen
* *Function detail:* Manager can see customer phone, customer name, order id, list services, price
* *Business rule:* BR-052
* *Screen layout:*

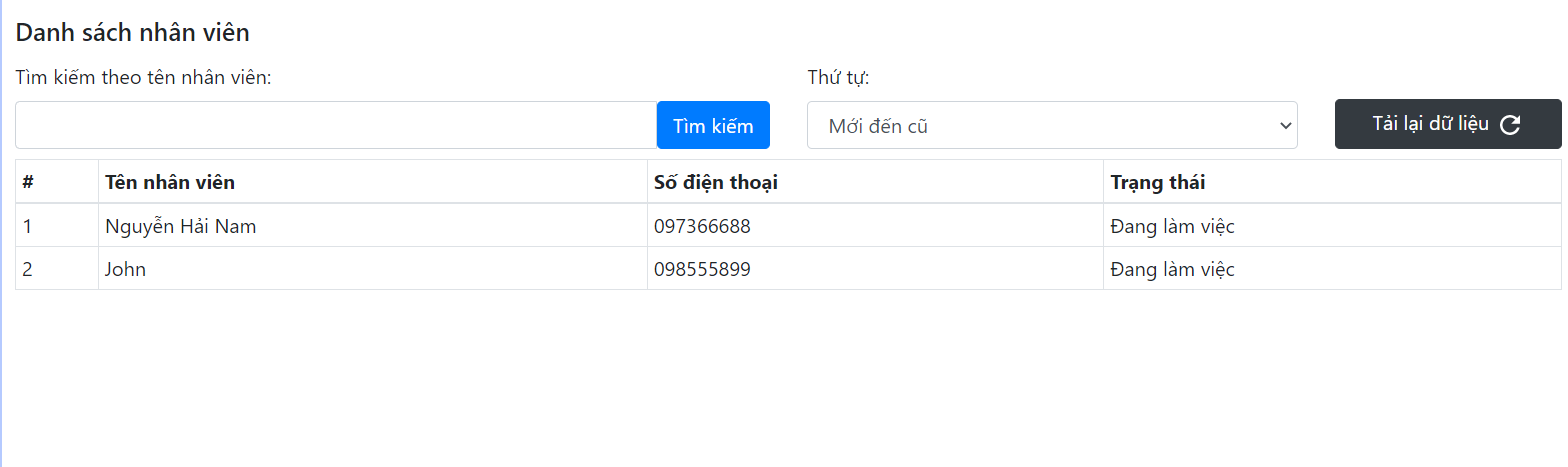
**

### 

### 3.4 Staff

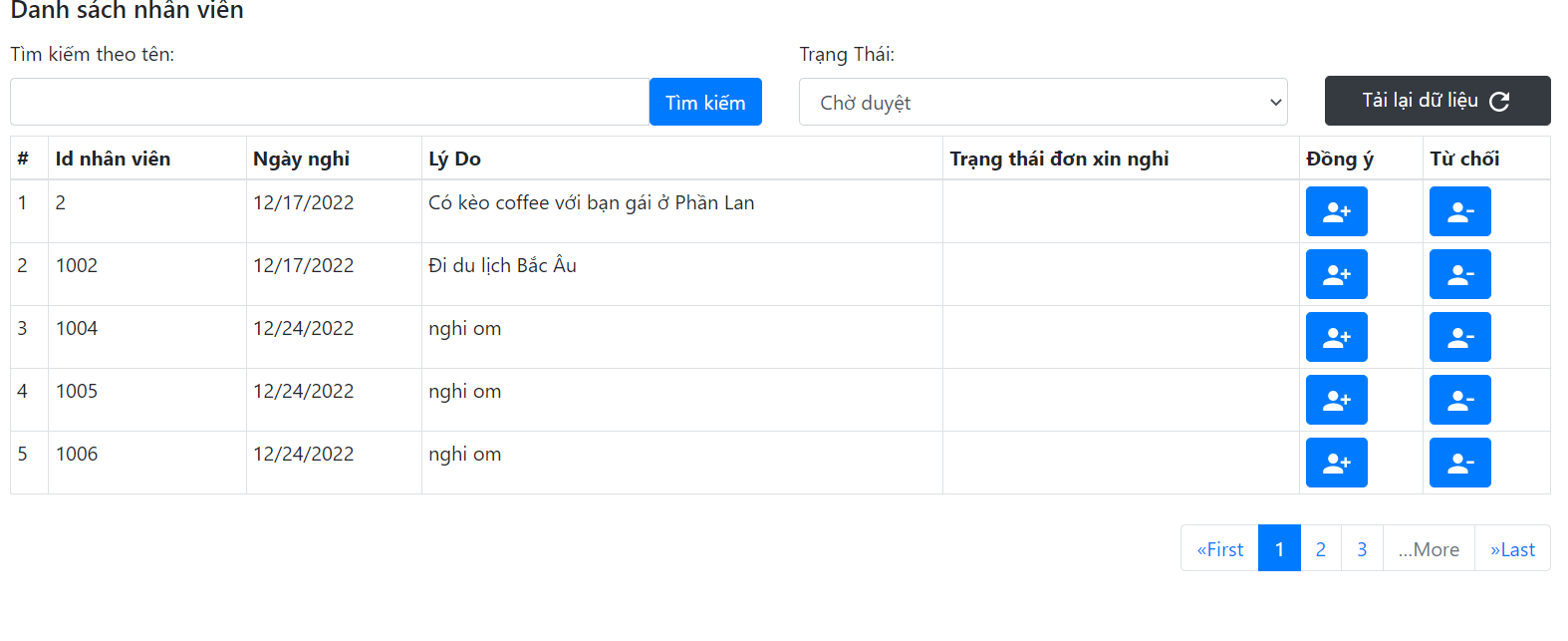
### 3.4.1 Staff screen

* *Function trigger:* Manager want to see list of staff
* *Actor:* manager
* *Function description:* Manager want to see list of staff
* *Function detail:* Manager can see staff name, staff phone and staff working status
* *Business rule:*  BR-036, BR-062
* *Screen layout:*

**

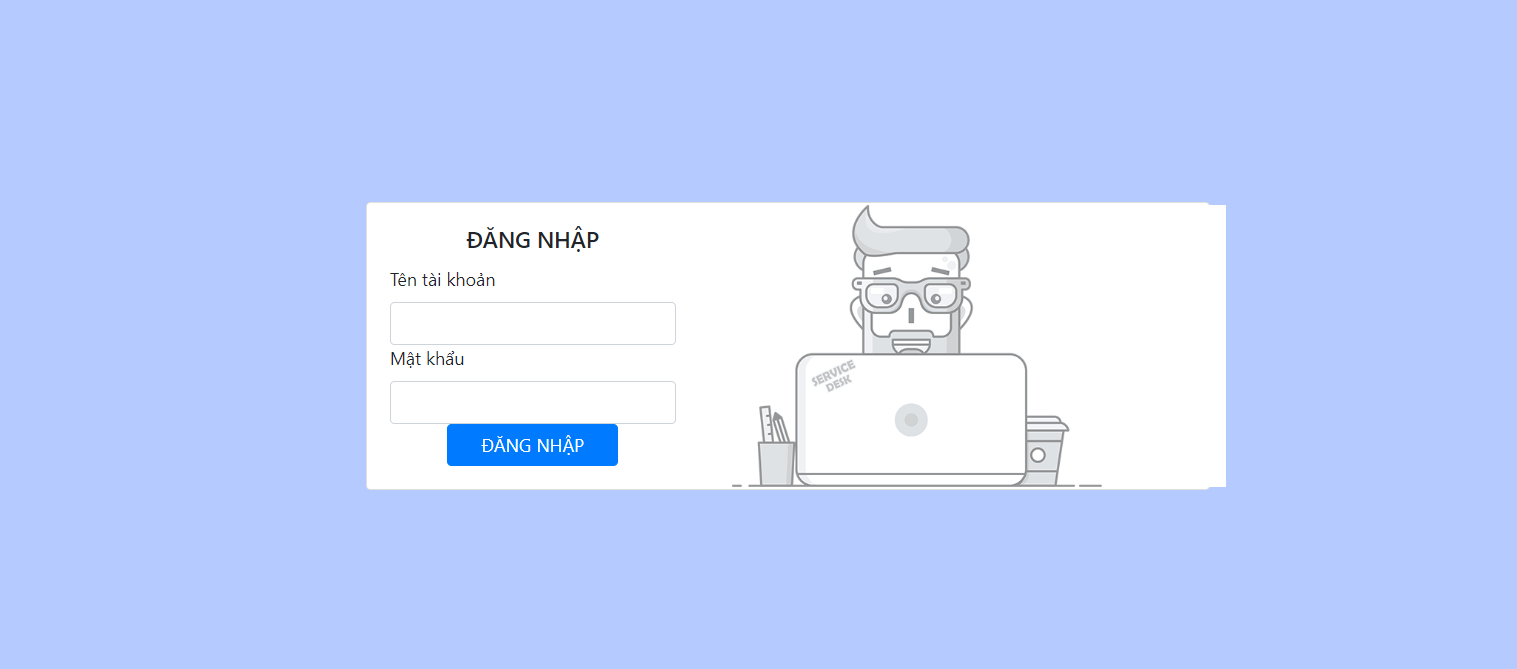
### 3.4.1 Staff day off screen

* *Function trigger:* Manager want to staff day off
* *Actor:* manager
* *Function description:* Manager want to see list of staff day off
* *Function detail:* Manager can see staff name, staff phone and day off reason, confirm or cancel
* *Business rule:* N/A
* *Screen layout:*

**

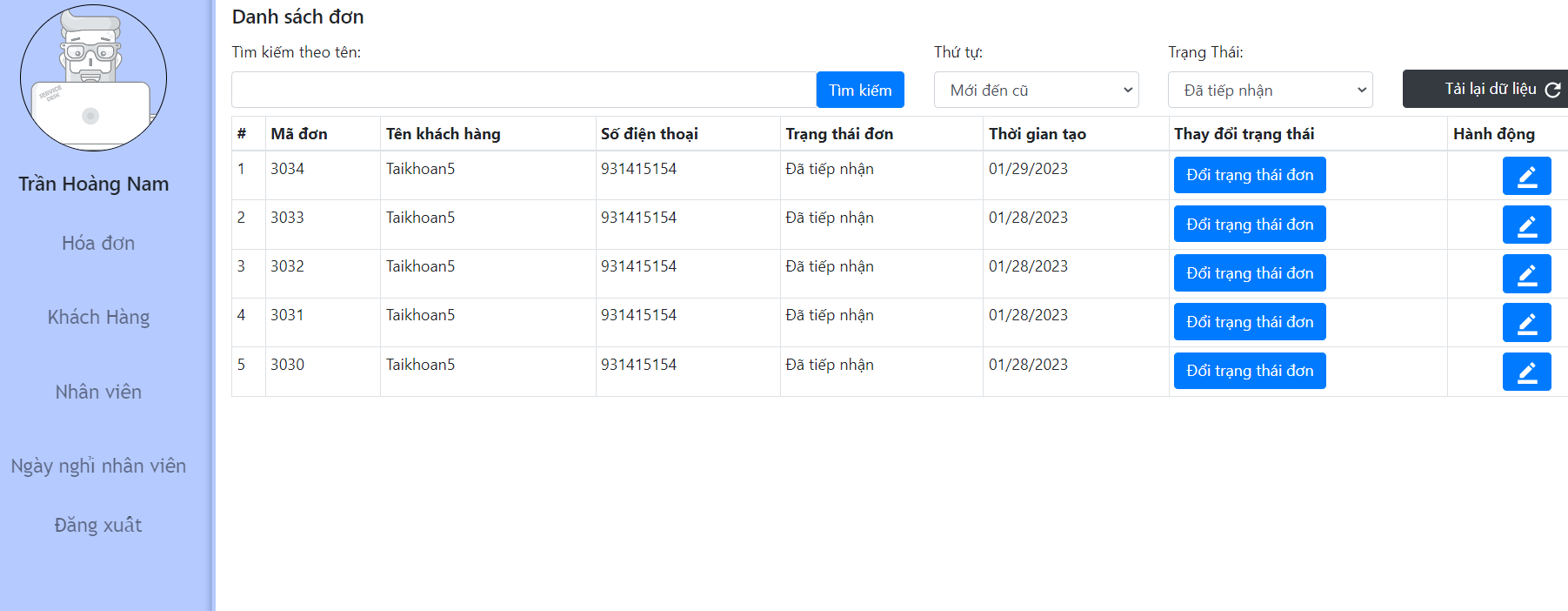
### 3.5 Login screen

* *Function trigger:* Manager want to login the system
* *Actor:* manager
* *Function description:*Manager use username and password to be able to login
* *Function detail:* Manager logs into the system by entering username and password. The system verifies the user’s identity and confirms whether they are allowed to access or not.
* *Business rule:* BR-009
* *Screen layout:*

**

### 3.6 Logout screen

* *Function trigger:* Manager want to logout of the system
* *Actor:* manager
* *Function description:* Manager want to see list of staff day off
* *Function detail:* Manager click logout and logout of the system
* *Business rule:* N/A
* *Screen layout:*

**

## 4. Non-Functional Requirements

### 4.1 External Interfaces

### UI-1: The user interfaces must be simple, friendly.

### UI-2: Icons used must be easy to recognize.

### UI-3: Language used in the web and mobile application is Vietnamese.

## 4.2.1 Usability

## US-1: The mobile application for the customer should be simple and easy to use.

## US-1: The mobile application for the staff should be simple and easy to use.

## US-2: The web manager for the manager should be easy to use and have an eye-catching user interface.

## US-3: The web application for the manager should require no more than one day of training to be used.

## US-4: Staff are not required to have technical knowledge before using.

## 4.2.2 Reliability

RE-1: The system should handle 500 requests per minute.

## 5. Requirement Appendix

### 5.1 Business Rules

| **ID** | **Rule Definition** |
| --- | --- |
| BR-001 | One phone number can only register one account for customer. |
| BR-002 | Phone number register on account is regarded as username. |
| BR-003 | Account register via register on mobile application has default role is customer. |
| BR-004 | After account created, the customer can use this account to login into the Furniture Repair and Maintenance Service System. |
| BR-005 | Default account include: username ,phone number, password, email, birthday, address. |
| BR-006 | Clear an authorised user session when they log out. |
| BR-007 | The user must have logged in to the Computer Service  System. |
| BR-008 | After login mobile app, users will be redirected to a specific view based on their role on the system:   * If the role is “Staff”, the system will display the Staff home screen on a mobile application. * If the role is “Customer”, the system will display the customer home screen on a mobile application**.** |
| BR-009 | After login to the web app, users will be redirected to a specific view based on their role on the system:   * If the role is “Manager”, the system will display the Manager Order on a web application. |
| BR-010 | The User can not change their username once it has been registered. |
| BR-011 | The User can change phone number in profile. |
| BR-012 | Creating booking on mobile applications is for door to door service. |
| BR-013 | The Customer can view order and booking details history. |
| BR-014 | When booking, the customer must choose date time, services and fill in specific information:  Required: name, phone number, address.  Optional: computer model, computer condition, description. |
| BR-015 | When booking, The customer can only choose one day and one slot. |
| BR-016 | Date created booking must from today to next 7 days. |
| BR-017 | The Customer can not choose slots unavailable for creating booking. |
| BR-018 | Booking have status:   * ”Đã tiếp nhận” : When it was first created and not accepted by the manager. * “Đang thực hiện”: When it have been accepted by manager. * “Hủy”:  When it have been denied by manager. |
| BR-019 | When a customer creates a new booking, the customer must wait for the confirmation by Manager. |
| BR-020 | The Customer can edit/cancel their booking. |
| BR-021 | The User can change their password. |
| BR-022 | Customer can only edit/cancel booking when booking’s status is “Đang xử lý” or order status is “Đang chờ”. |
| BR-023 | When booking is accepted, order of booking will be created. |
| BR-024 | Booking can be denied by the manager. |
| BR-025 | Order have status:   * “Đang chờ”: When order has been created. * “Chờ xác nhận”: When staff update services or accessories to order. * “Đang khảo sát”: When services and accessories are accepted by the manager. * “Chờ thanh toán”: When a customer accepts an order after manager acceptance. * “Hoàn tất đơn”: When staff complete all services in order and The customer pays the bill. * “Hủy”: When staff cannot complete. |
| BR-026 | The Customer can view the staff's name, their computer information, services being added and price in order. |
| BR-027 | Staff can see the list of the order which they are assigned. |
| BR-028 | Staff must assign the customer's computer information in order when order status is “Chờ xác nhận”. |
| BR-035 | Managers can get the customer’s account list. |
| BR-036 | Managers can get their staff account list. |
| BR-037 | When a staff member starts to work in the store, his/her information will be added to the system by the Manager. |
| BR-044 | The User can view the services list. |
| BR-045 | The User can view the accessories list. |
| BR-050 | Accessory name is unique. |
| BR-051 | Service name is unique. |
| BR-052 | Manager can view customer bookings history. |
| BR-054 | System report data is weekly, monthly |
| BR-055 | System report data is data of booking, order, and new customer. |
| BR-056 | Manager can only accept order details when the order status is ‘Đang tiến hành’. |
| BR-057 | Manager can accept order details many times. |
| BR-058 | Staff can add new services or accessory when order status is “Đang chờ”, “Chờ xác nhận” or “Quản lý xác nhận”. |
| BR-059 | Manager can only approved order details when the order status is ‘Đang thực hiện’. |
| BR-060 | Manager can view the customer's details. |
| BR-061 | Manager can view schedule details. |
| BR-062 | Manager can view the staff's detailed information. |
| BR-069 | The Customer can accept service that is used in their order. |
| BR-070 | Staff can only start to perform repair work when The customer accepts. |
| BR-071 | Staff must confirm their completion after performing repair successfully. |
| BR-072 | Staff can cancel orders when staff can not have agreement with The customer. |

*Table 15. Business rules*

### 5.2 Common Requirements

N/A

### 5.3 Application Messages List

N/A

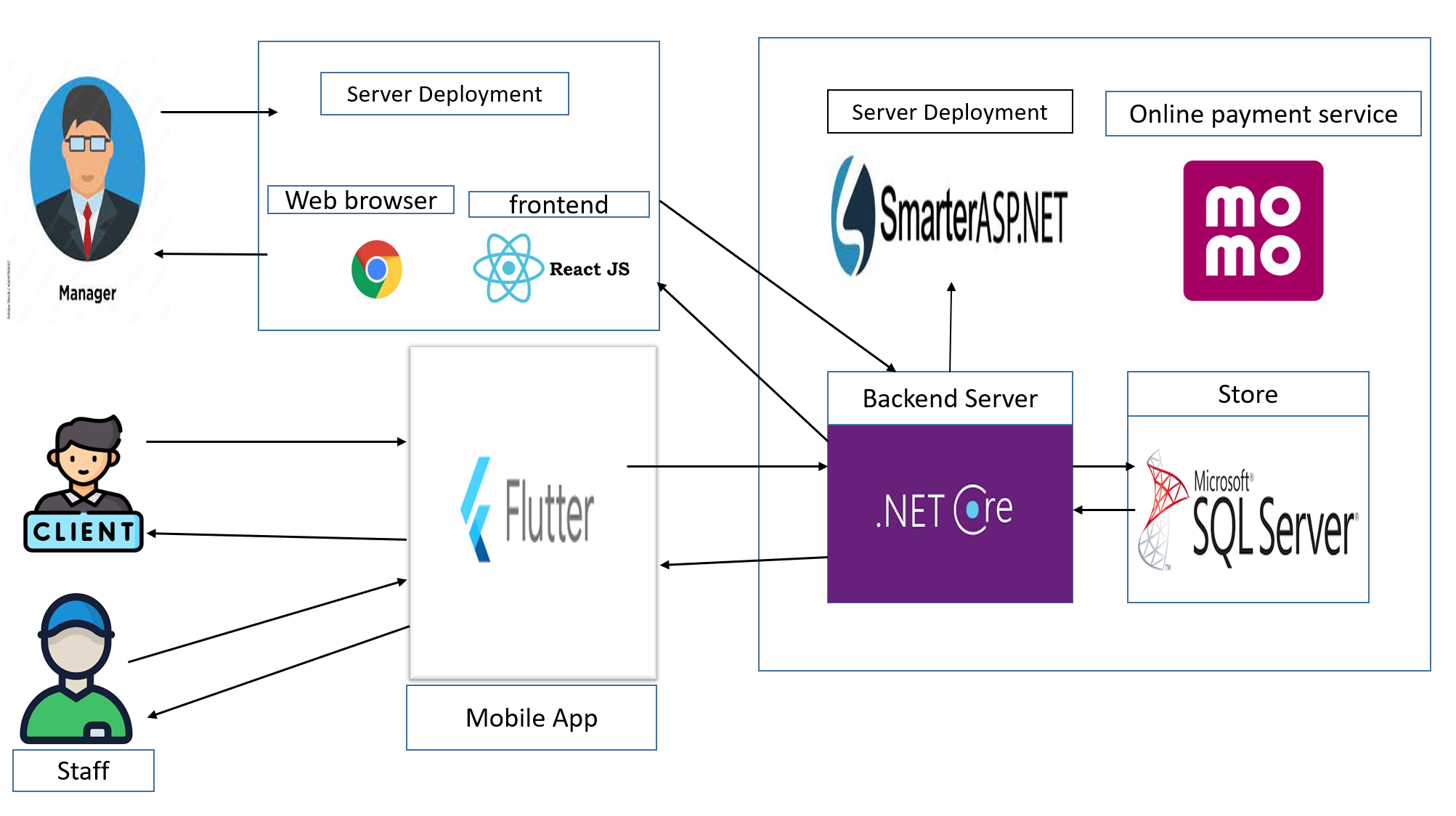
### 5.4 Other Requirements…

N/A

# IV. Software Design Description

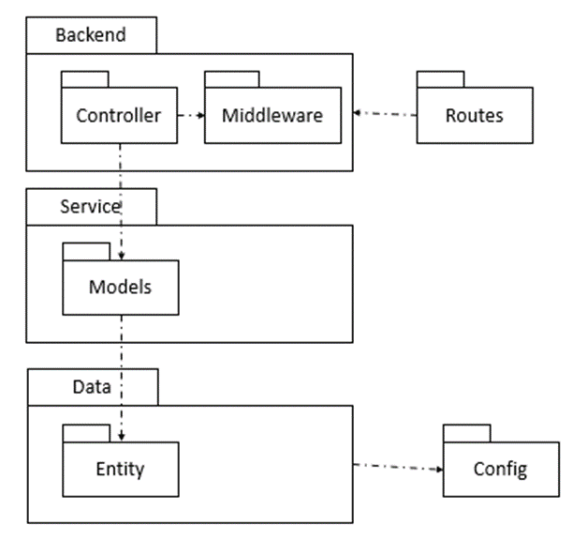
## 1. System Design

### 1.1 System Architecture



System Architecture

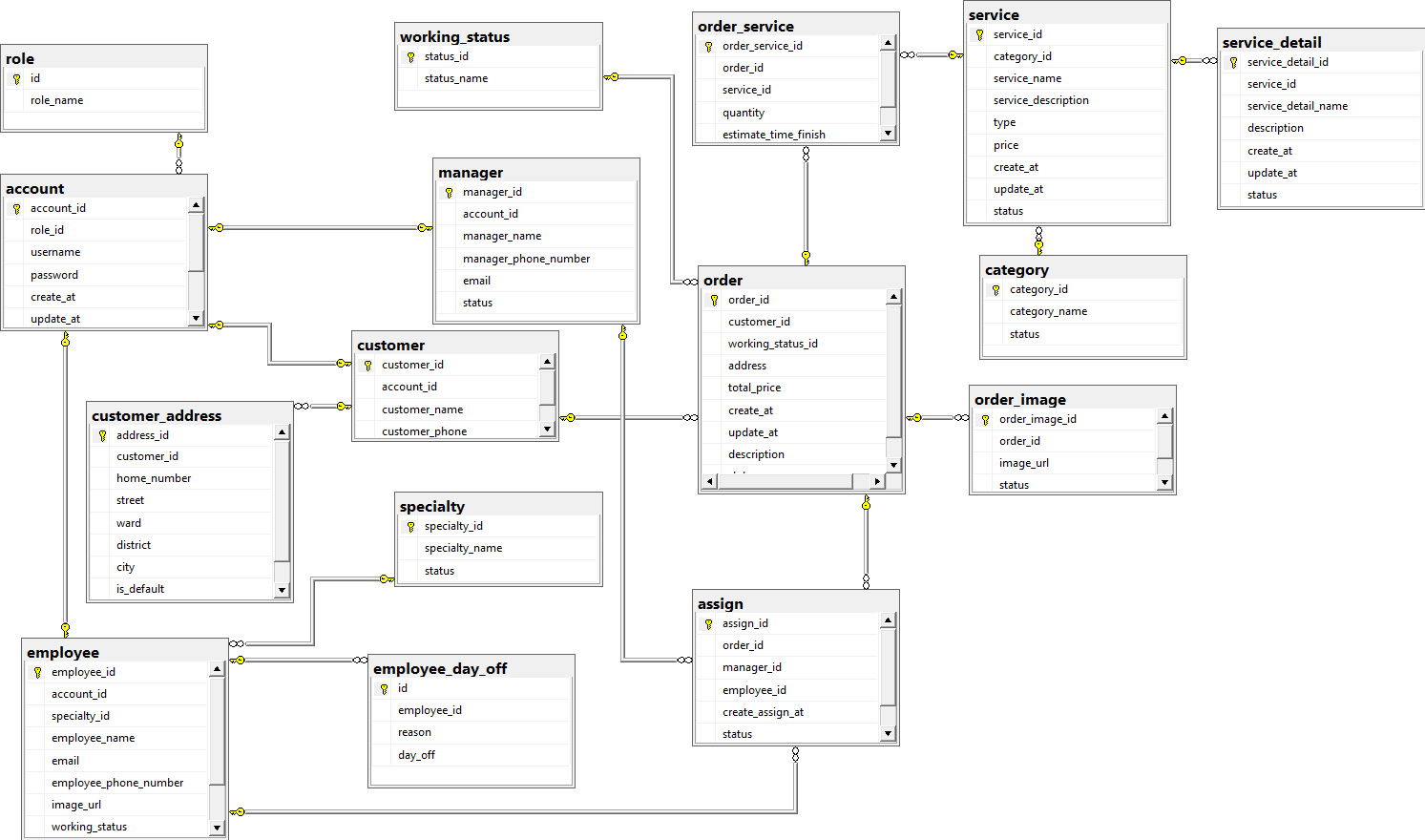
### 1.2 Package Diagram



| **No** | **Package** | **Description** |
| --- | --- | --- |
| 01 | Controller | *All controllers classes will define in this folder* |
| 02 | Routes | *All routers classes will define in this folder* |
| 03 | Models | All classes define here mapped with database collections |
| 04 | Middleware | All middleware classes will define in this folder |
| 05 | Config | All config classes will define in this folder |

Describe backend package diagram

## 2. Database Design



Database Diagram

#### 2.1. Table Role

| **Field name** | **Type** | **Size** | **Unique** | **Not Null** | **PK/FK** | **Notes** |
| --- | --- | --- | --- | --- | --- | --- |
| id | int |  | yes | yes | PK |  |
| role\_name | nvarchar | max |  |  |  |  |

#### 2.2. Table Account

| **Field name** | **Type** | **Size** | **Unique** | **Not Null** | **PK/FK** | **Notes** |
| --- | --- | --- | --- | --- | --- | --- |
| Account\_id | int |  | yes | yes | PK |  |
| Role\_id | int |  |  | yes | FK |  |
| username | nvarchar | max |  | Yes |  |  |
| password | nvarchar | max |  | Yes |  |  |
| Account\_status | boolean |  |  | Yes |  |  |
| Create\_at | datetime |  |  | Yes |  |  |
| Update\_at | datetime |  |  |  |  |  |
| Refresh\_token | nvarchar | max |  |  |  |  |

#### 2.3. Table Manager

| **Field name** | **Type** | **Size** | **Unique** | **Not Null** | **PK/FK** | **Notes** |
| --- | --- | --- | --- | --- | --- | --- |
| id | int |  |  | yes | PK |  |
| account\_id | int |  | Yes | yes | FK |  |
| manager\_name | nvarchar | max |  | Yes |  |  |
| phone\_number | varchar |  |  | Yes |  |  |
| email | nvarchar | max |  | Yes |  |  |

#### 2.4. Table Customer

| **Field name** | **Type** | **Size** | **Unique** | **Not Null** | **PK/FK** | **Notes** |
| --- | --- | --- | --- | --- | --- | --- |
| id | int |  |  | yes | PK |  |
| account\_id | int |  | Yes | yes | FK |  |
| customer\_name | varchar |  |  | Yes |  |  |
| phone\_number | varchar |  |  | Yes |  |  |
| email | varchar |  |  | Yes |  |  |

#### 2.5. Table Employee

| **Field name** | **Type** | **Size** | **Unique** | **Not Null** | **PK/FK** | **Notes** |
| --- | --- | --- | --- | --- | --- | --- |
| id | int |  |  | yes | PK |  |
| account\_id | int |  | Yes | yes | FK |  |
| employee\_name | nvarchar | max |  | Yes |  |  |
| phone\_number | varchar |  |  | Yes |  |  |
| Email | nvarchar | max |  | Yes |  |  |
| Address | varchar |  |  |  |  |  |
| Image\_url | nvarchar | max |  |  |  |  |
| status | boolean |  |  | Yes |  |  |

#### 2.6. Table Customer Address

| **Field name** | **Type** | **Size** | **Unique** | **Not Null** | **PK/FK** | **Notes** |
| --- | --- | --- | --- | --- | --- | --- |
| address\_id | int |  |  | yes | PK |  |
| customer\_id | int |  |  | yes | FK |  |
| home\_number | varchar |  |  | Yes |  |  |
| street | varchar |  |  | Yes |  |  |
| ward | varchar |  |  | Yes |  |  |
| district | varchar |  |  | Yes |  |  |
| city | varchar |  |  | Yes |  |  |
| Is\_default | boolean |  |  | Yes |  |  |

#### 2.7. Table Assign

| **Field name** | **Type** | **Size** | **Unique** | **Not Null** | **PK/FK** | **Notes** |
| --- | --- | --- | --- | --- | --- | --- |
| id | int |  |  | yes | PK |  |
| order\_id | int |  |  | yes | FK |  |
| employee\_id | int |  |  | Yes | FK |  |
| manager\_id | int |  |  | Yes | FK |  |
| assign\_at | datetime |  |  | Yes |  |  |
| status | boolean |  |  | Yes |  |  |

#### 2.8. Table Order

| **Field name** | **Type** | **Size** | **Unique** | **Not Null** | **PK/FK** | **Notes** |
| --- | --- | --- | --- | --- | --- | --- |
| order\_id | int |  |  | yes | PK |  |
| working\_status\_id | int |  |  | yes | FK |  |
| total\_price | nvarchar | max |  | Yes |  |  |
| address | varchar |  |  | Yes |  |  |
| create\_at | datetime |  |  | Yes |  |  |
| update\_at | datetime |  |  |  |  |  |
| description | nvarchar | max |  |  |  |  |
| status | boolean |  |  | Yes |  |  |

#### 2.9. Table Working Status

| **Field name** | **Type** | **Size** | **Unique** | **Not Null** | **PK/FK** | **Notes** |
| --- | --- | --- | --- | --- | --- | --- |
| status\_id | int |  |  | yes | PK |  |
| status\_name | varchar |  |  | yes |  |  |

#### 2.10. Table Order Service

| **Field name** | **Type** | **Size** | **Unique** | **Not Null** | **PK/FK** | **Notes** |
| --- | --- | --- | --- | --- | --- | --- |
| order\_service\_id | int |  |  | yes | PK |  |
| order\_id | int |  |  | yes | FK |  |
| service\_id | int |  |  | Yes | FK |  |
| quantity | int |  |  | Yes |  |  |
| estimate\_time | varchar |  |  |  |  |  |

#### 2.11. Table Order Image

| **Field name** | **Type** | **Size** | **Unique** | **Not Null** | **PK/FK** | **Notes** |
| --- | --- | --- | --- | --- | --- | --- |
| order\_image\_id | int |  |  | yes | PK |  |
| order\_id | int |  |  | yes | FK |  |
| image\_url | nvarchar | max |  | Yes |  |  |
| status | boolean |  |  | Yes |  |  |

#### 2.12. Table Specialty

| **Field name** | **Type** | **Size** | **Unique** | **Not Null** | **PK/FK** | **Notes** |
| --- | --- | --- | --- | --- | --- | --- |
| id | int |  |  | yes | PK |  |
| employee\_id | int |  |  | yes | FK |  |
| specialty\_name | nvarchar | max |  | Yes |  |  |

#### 2.13. Table Service

| **Field name** | **Type** | **Size** | **Unique** | **Not Null** | **PK/FK** | **Notes** |
| --- | --- | --- | --- | --- | --- | --- |
| service\_id | int |  |  | yes | PK |  |
| category\_id | int |  |  | yes | FK |  |
| name | nvarchar | max |  | Yes |  |  |
| description | nvarchar | max |  |  |  |  |
| price | nvarchar | max |  |  |  |  |
| create\_at | datetime |  |  | yes |  |  |
| update\_at | datetime |  |  |  |  |  |
| status | boolean |  |  | yes |  |  |

#### 2.14. Table Service Detail

| **Field name** | **Type** | **Size** | **Unique** | **Not Null** | **PK/FK** | **Notes** |
| --- | --- | --- | --- | --- | --- | --- |
| id | int |  |  | yes | PK |  |
| service\_id | int |  |  | yes | FK |  |
| service\_detail\_name | nvarchar | max |  | Yes |  |  |
| create\_at | datetime |  |  | yes |  |  |
| update\_at | datetime |  |  |  |  |  |
| status | boolean |  |  | yes |  |  |

#### 

#### 2.15. Table Category

| **Field name** | **Type** | **Size** | **Unique** | **Not Null** | **PK/FK** | **Notes** |
| --- | --- | --- | --- | --- | --- | --- |
| category\_id | int |  |  | yes | PK |  |
| category\_name | int |  |  | yes |  |  |
| status | boolean |  |  | yes |  |  |

#### 

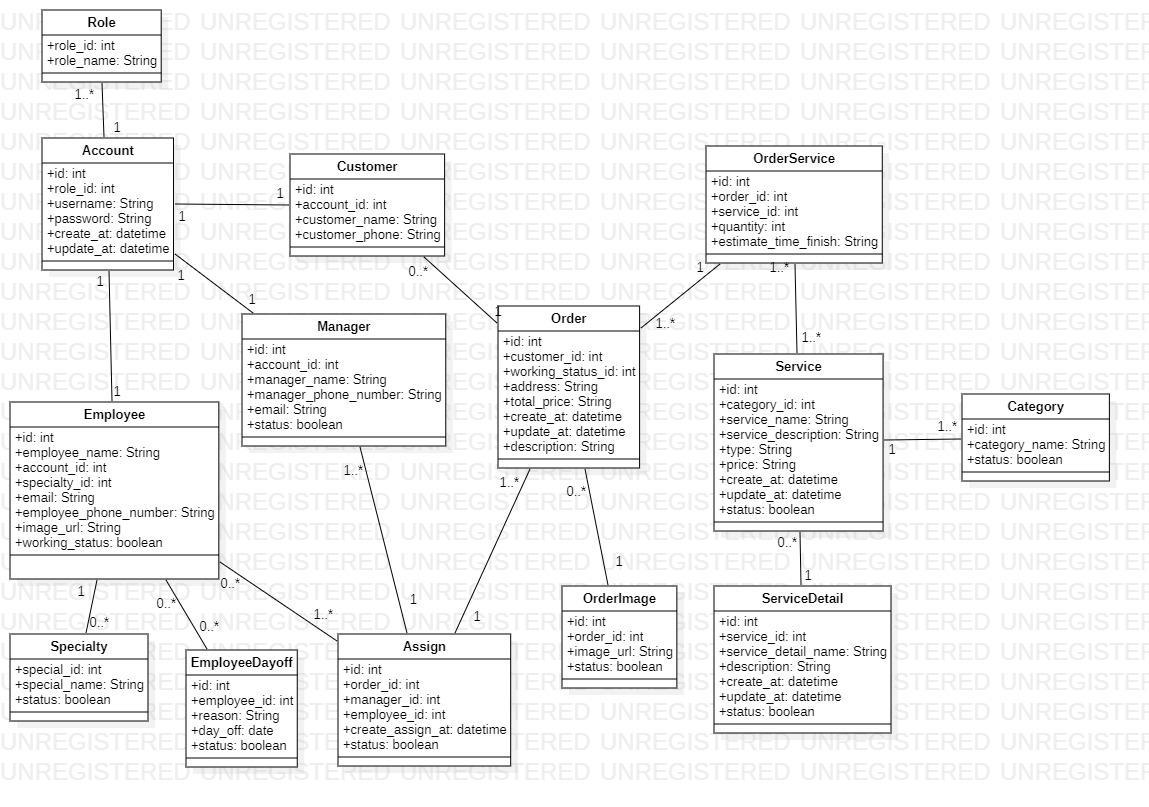
#### 2.16. Table Employee Day Off

| **Field name** | **Type** | **Size** | **Unique** | **Not Null** | **PK/FK** | **Notes** |
| --- | --- | --- | --- | --- | --- | --- |
| id | int |  |  | yes | PK |  |
| employee\_id | int |  |  | yes | FK |  |
| day\_off | date |  |  | yes |  |  |
| reason | nvarchar | max |  | yes |  |  |

## 

## 3. Detailed Design

#### 3.1.1 Class Diagram



Class Diagram

##### **3.1.2. Account**

| **No** | **Modifier** | **Type** | **Field** | **Description** |
| --- | --- | --- | --- | --- |
| 01 | protected | int | id | Unique key |
| 02 | protected | string | username | Username of user |
| 03 | protected | string | password | Encode password of user |
| 04 | protected | string | status | Status of account |
| 05 | protected | int | role\_id | Role of account |
| 06 | protected | datetime | create\_at | Date create account |
| 07 | protected | datetime | update\_at | Date update account |

*Account class attribute*

| **No** | **Modifier** | **Method** | **Description** |
| --- | --- | --- | --- |
| 01 | public | register() | Register new account into system |
| 02 | public | findOne() | function support by entity framework core |
| 03 | public | save() | function support by entity framework core |
| 04 | public | findById() | function support by entity framework core |
| 05 | public | updateOne | function support by entity framework core |

*Account class method*

##### **3.1.3. Customer**

| **No** | **Modifier** | **Type** | **Field** | **Description** |
| --- | --- | --- | --- | --- |
| 01 | protected | int | id | Unique key |
| 02 | protected | string | account\_id | Account id of customer is unique |
| 03 | protected | string | customer\_phone | Phone number of customer |
| 04 | protected | string | customer\_name | Name of customer |

*Customer class attribute*

| **No** | **Modifier** | **Method** | **Description** |
| --- | --- | --- | --- |
| 01 | public | register() | Register new account into system |
| 02 | public | findOne() | function support by entity framework core |
| 03 | public | save() | function support by entity framework core |
| 04 | public | findById() | function support by entity framework core |
| 05 | public | updateOne | function support by entity framework core |

*Customer class method*

##### **3.1.4. Order**

| **No** | **Modifier** | **Type** | **Field** | **Description** |
| --- | --- | --- | --- | --- |
| 01 | protected | int | id | Unique key |
| 02 | protected | string | customer\_id | Username of user |
| 03 | protected | string | working\_status \_id | Encode password of user |
| 04 | protected | string | Total price | Status of account |
| 05 | protected | int | address | Role of account |
| 06 | protected | datetime | create\_at | Date create account |
| 07 | protected | datetime | update\_at | Date update account |
| 08 | protected | datetime | description | Employee description |

*Order class attribute*

| **No** | **Modifier** | **Method** | **Description** |
| --- | --- | --- | --- |
| 01 | public | register() | Register new account into system |
| 02 | public | findOne() | function support by entity framework core |
| 03 | public | save() | function support by entity framework core |
| 04 | public | findById() | function support by entity framework core |
| 05 | public | updateOne | function support by entity framework core |

*Order class method*

##### **3.1.5. Order Service**

| **No** | **Modifier** | **Type** | **Field** | **Description** |
| --- | --- | --- | --- | --- |
| 01 | protected | int | id | primary key |
| 02 | protected | int | order\_id | Foreign key of order |
| 03 | protected | int | service\_id | Foreign key of service |
| 04 | protected | int | quantity | Quantity of service |
| 05 | protected | String | estimate\_time\_finish | Time manager assign for employee |

*OrderService class attribute*

| **No** | **Modifier** | **Method** | **Description** |
| --- | --- | --- | --- |
| 01 | public | Create() | Create new orderservice by entity framework core |
| 02 | public | findOne() | function support by entity framework core |
| 03 | public | save() | function support by entity framework core |
| 04 | public | findById() | function support by entity framework core |
| 05 | public | updateOne | function support by entity framework core |

*OrderService class method*

##### **3.1.6. Working Status**

| **No** | **Modifier** | **Type** | **Field** | **Description** |
| --- | --- | --- | --- | --- |
| 01 | protected | int | id | primary key |
| 02 | protected | int | working\_status\_name | Name of order working status |

*WorkingStatus class attribute*

| **No** | **Modifier** | **Method** | **Description** |
| --- | --- | --- | --- |
| 01 | public | Create() | Create new orderservice by entity framework core |
| 02 | public | findOne() | function support by entity framework core |
| 03 | public | save() | function support by entity framework core |
| 04 | public | findById() | function support by entity framework core |
| 05 | public | updateOne | function support by entity framework core |

*WorkingStatus class method*

##### **3.1.7. Service**

| **No** | **Modifier** | **Type** | **Field** | **Description** |
| --- | --- | --- | --- | --- |
| 01 | protected | int | id | primary key |
| 02 | protected | int | category\_id | Foreign key of category |
| 03 | protected | int | service\_name | Name of service |
| 04 | protected | int | service\_description | Service description |
| 05 | protected | String | type | Type of service(wood, metal…) |
| 06 | protected | String | price | Price of service |
| 07 | protected | datetime | create\_at | Date create service |
| 08 | protected | datetime | update\_at | Date update service |
| 09 | protected | Boolean | status | Status of service |

*Service class attribute*

| **No** | **Modifier** | **Method** | **Description** |
| --- | --- | --- | --- |
| 01 | public | Create() | Create new service by entity framework core |
| 02 | public | findOne() | function support by entity framework core |
| 03 | public | save() | function support by entity framework core |
| 04 | public | findById() | function support by entity framework core |
| 05 | public | updateOne | function support by entity framework core |

*Service class method*

##### **3.1.8. Category**

| **No** | **Modifier** | **Type** | **Field** | **Description** |
| --- | --- | --- | --- | --- |
| 01 | protected | int | id | primary key |
| 02 | protected | int | category\_name | Name of category |
| 03 | protected | int | status | Status of category |

*Category class attribute*

| **No** | **Modifier** | **Method** | **Description** |
| --- | --- | --- | --- |
| 01 | public | Create() | Create new orderservice by entity framework core |
| 02 | public | findOne() | function support by entity framework core |
| 03 | public | save() | function support by entity framework core |
| 04 | public | findById() | function support by entity framework core |
| 05 | public | updateOne | function support by entity framework core |

*Category class method*

##### **3.1.9. Assign**

| **No** | **Modifier** | **Type** | **Field** | **Description** |
| --- | --- | --- | --- | --- |
| 01 | protected | int | id | primary key |
| 02 | protected | int | order\_id | Foreign key of order |
| 03 | protected | int | manager\_id | Foreign key of manager |
| 04 | Protected | int | employee-id | Foreign key of employee |
| 05 | Protected | datetime | create\_assign\_at | Date time create assign |
| 06 | protected | boolean | status | Status of assign |

*Assign class attribute*

| **No** | **Modifier** | **Method** | **Description** |
| --- | --- | --- | --- |
| 01 | public | Create() | Create new Assign by entity framework core |
| 02 | public | findOne() | function support by entity framework core |
| 03 | public | save() | function support by entity framework core |
| 04 | public | findById() | function support by entity framework core |
| 05 | public | updateOne | function support by entity framework core |

*Assign class method*

##### **3.1.10. Manager**

| **No** | **Modifier** | **Type** | **Field** | **Description** |
| --- | --- | --- | --- | --- |
| 01 | protected | int | id | primary key |
| 02 | protected | int | Account\_id | Unique key of account |
| 03 | protected | String | Manager\_name | Name of manager |
| 04 | Protected | String | Manager\_phone\_number | Phone number of manager |
| 05 | Protected | String | Email | Email of manager |
| 06 | Protected | boolean | status | Status of manager |

*Manager class attribute*

| **No** | **Modifier** | **Method** | **Description** |
| --- | --- | --- | --- |
| 01 | public | Create() | Create new orderservice by entity framework core |
| 02 | public | findOne() | function support by entity framework core |
| 03 | public | save() | function support by entity framework core |
| 04 | public | findById() | function support by entity framework core |
| 05 | public | updateOne | function support by entity framework core |

*Manager class method*

##### **3.1.11. Employee**

| **No** | **Modifier** | **Type** | **Field** | **Description** |
| --- | --- | --- | --- | --- |
| 01 | protected | int | id | primary key |
| 02 | protected | int | employee\_name | Name of category |
| 03 | protected | int | account\_id | Status of category |
| 04 | protected | Int | specialty\_id | Specialty of employee |
| 05 | protected | String | email | Email of employee |
| 06 | protected | String | employee\_phone\_number | Employee phone number |
| 07 | protected | String | image\_url | Image of employee |
| 08 | protected | boolean | working\_status | Status of employee |

*Employee class attribute*

| **No** | **Modifier** | **Method** | **Description** |
| --- | --- | --- | --- |
| 01 | public | Create() | Create new employee by entity framework core |
| 02 | public | findOne() | function support by entity framework core |
| 03 | public | save() | function support by entity framework core |
| 04 | public | findById() | function support by entity framework core |
| 05 | public | updateOne | function support by entity framework core |

*Employee class method*

##### **3.1.12. Order**

| **No** | **Modifier** | **Type** | **Field** | **Description** |
| --- | --- | --- | --- | --- |
| 01 | protected | int | id | Unique key |
| 02 | protected | string | customer\_id | Username of user |
| 03 | protected | string | working\_status \_id | Encode password of user |
| 04 | protected | string | Total price | Status of account |
| 05 | protected | int | address | Role of account |
| 06 | protected | datetime | create\_at | Date create account |
| 07 | protected | datetime | update\_at | Date update account |
| 08 | protected | datetime | description | Employee description |

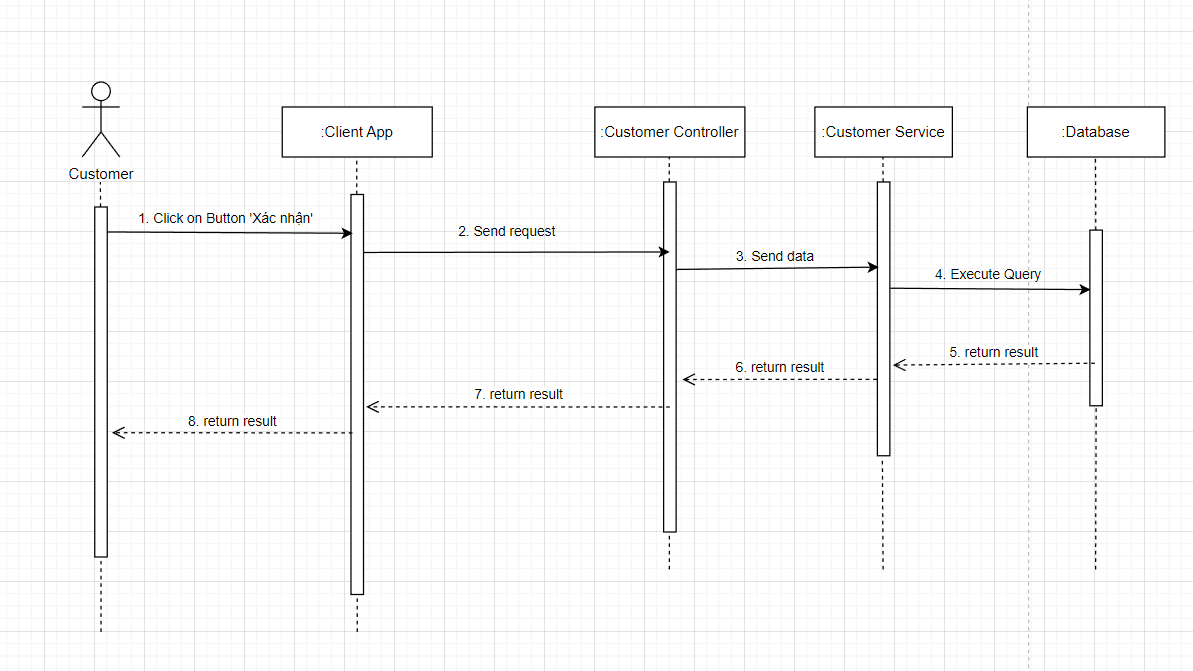
*Order class attribute*

| **No** | **Modifier** | **Method** | **Description** |
| --- | --- | --- | --- |
| 01 | public | register() | Register new account into system |
| 02 | public | findOne() | function support by entity framework core |
| 03 | public | save() | function support by entity framework core |
| 04 | public | findById() | function support by entity framework core |
| 05 | public | updateOne | function support by entity framework core |

*Order class method*

### 3.2 Sequence diagram

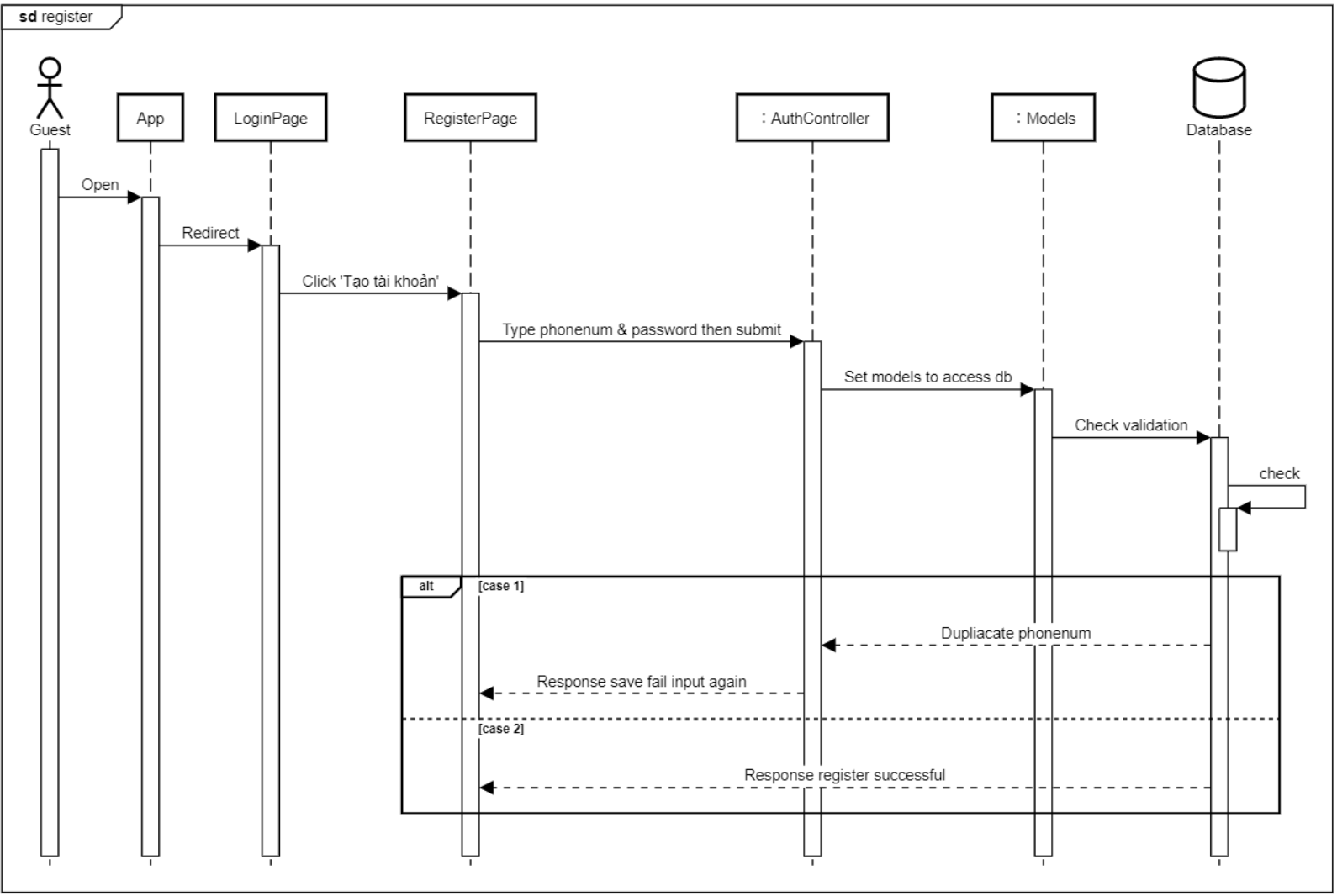
**3.2.1 Customer booking service**



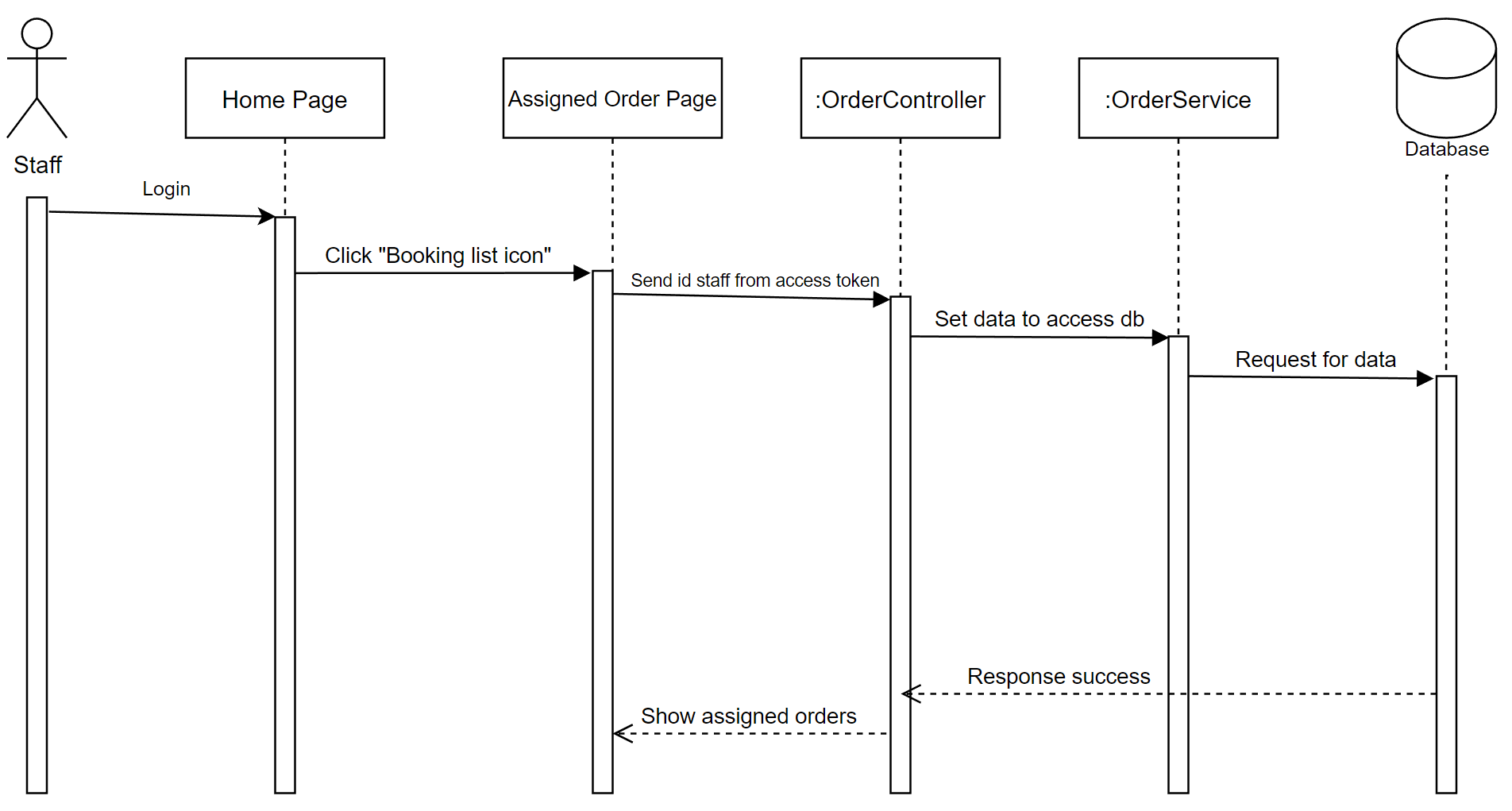
**3.2.2 Customer view all booking**



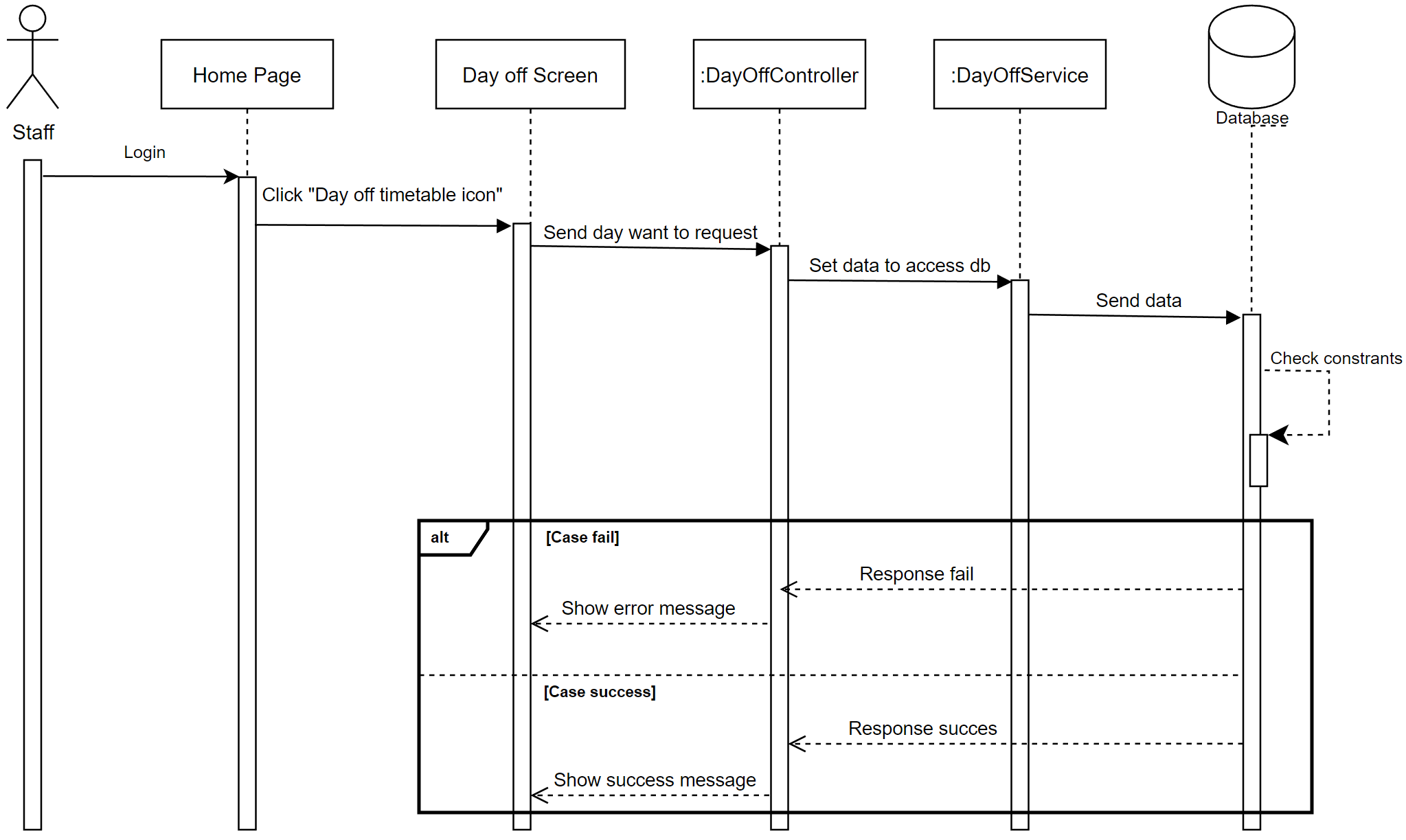
**3.2.3 Customer register account**



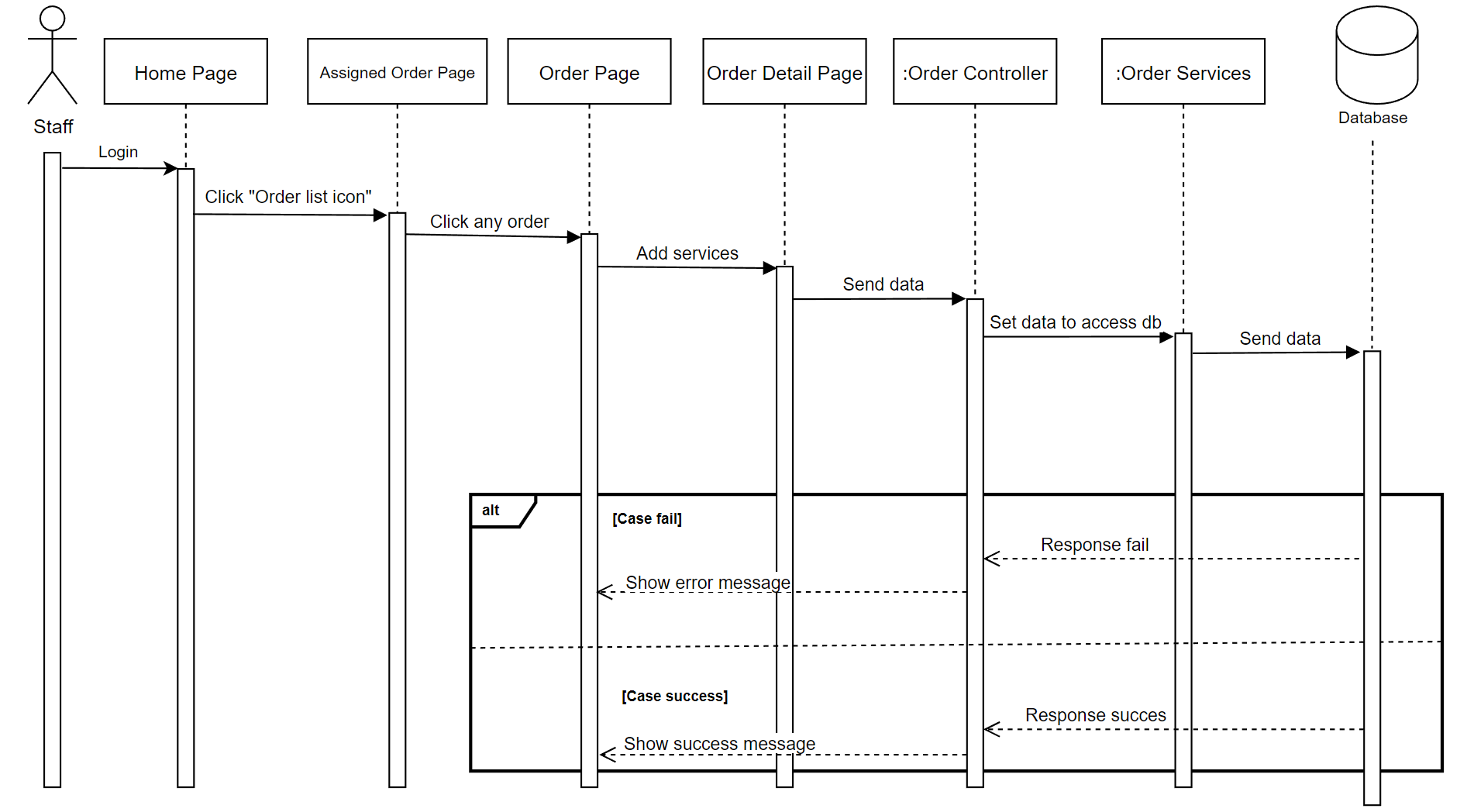
**3.2.4 Staff view assigned order**

****

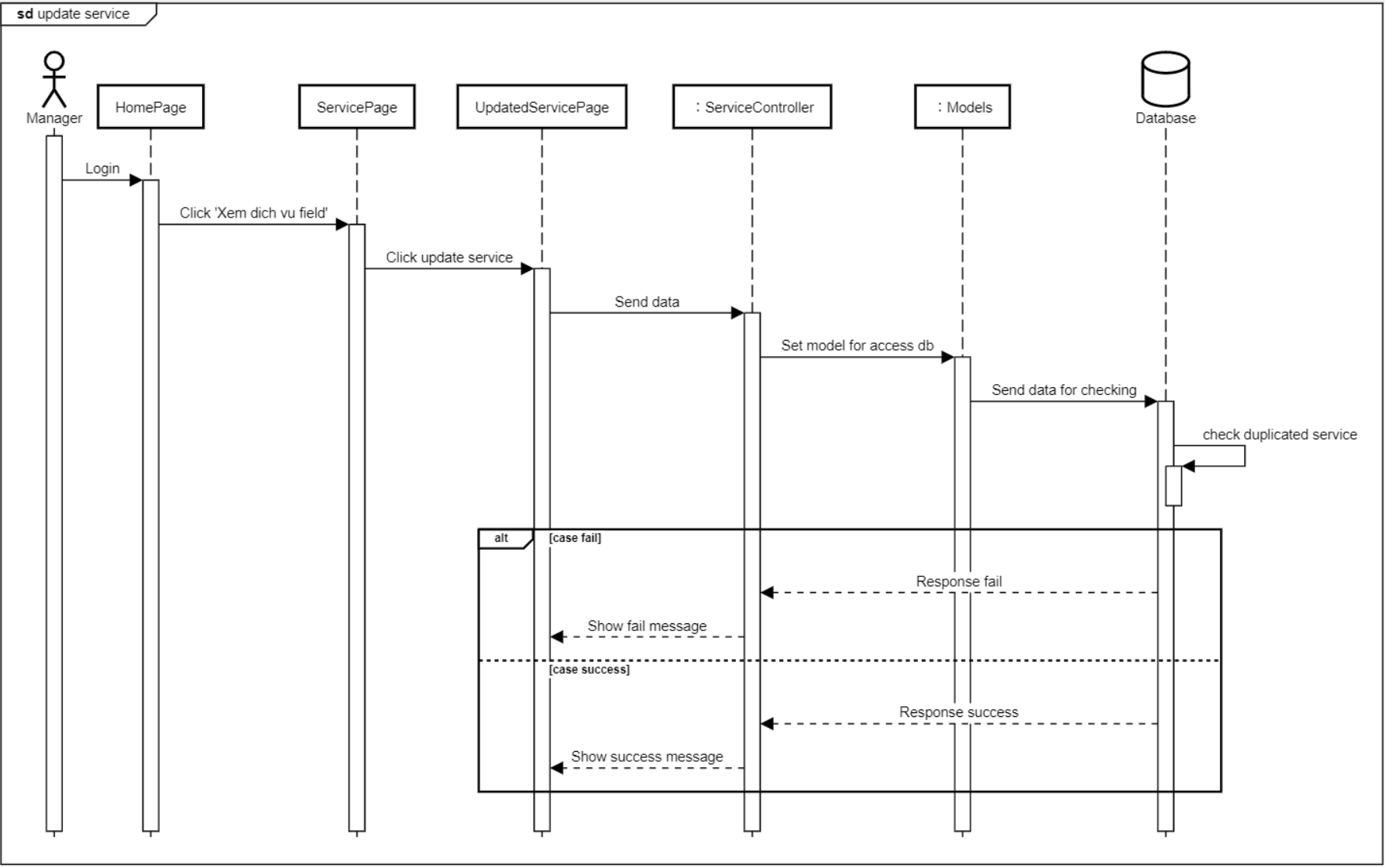
**3.2.5 Staff send day off request**

****

**3.2.6 Staff add services for assigned order**

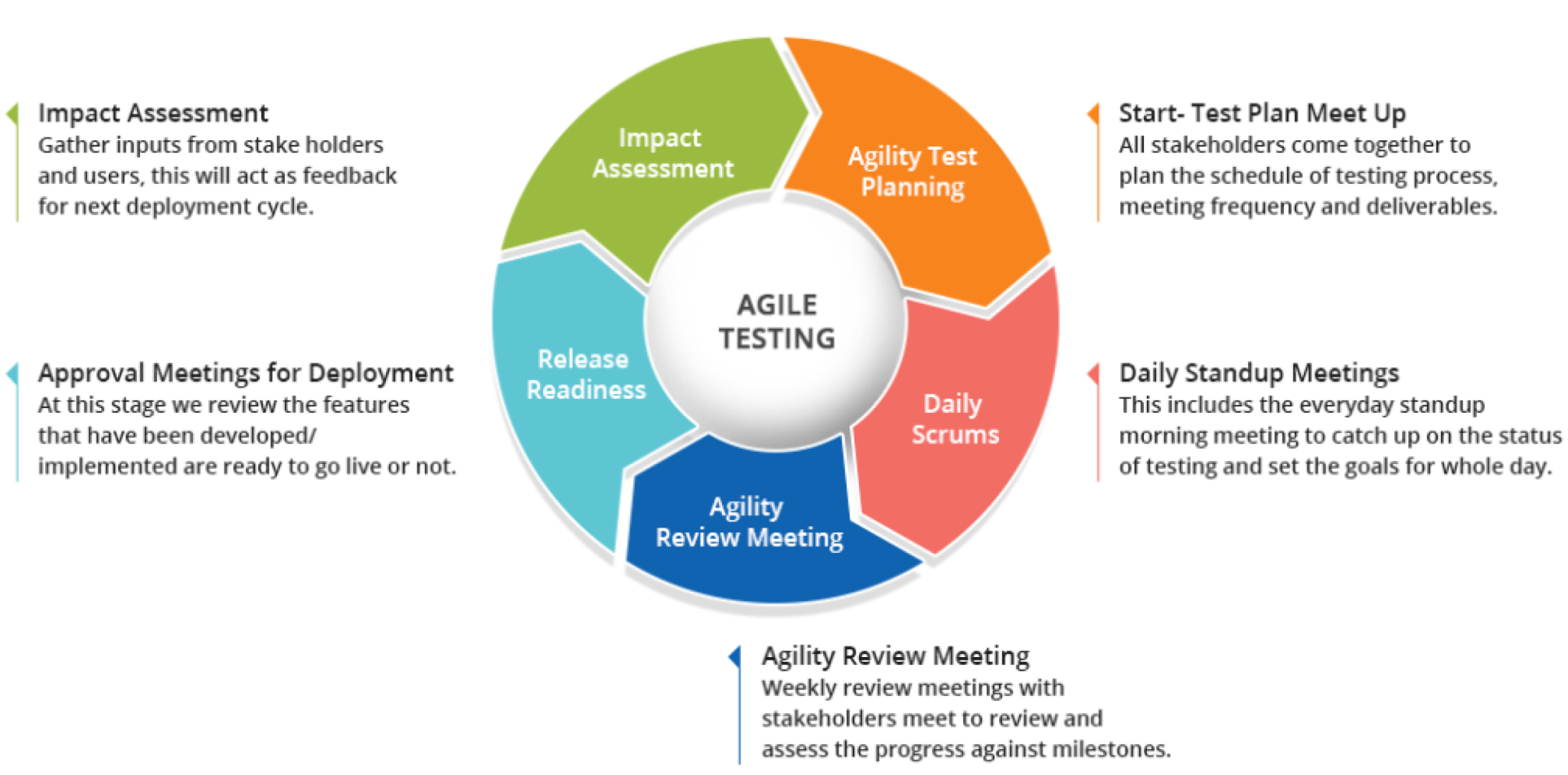
****

**3.2.7 Manager update service**

****

# V. Software Testing Documentation

## 1. Scope of Testing



**In project, we apply Agile testing for many aims:**

* Detect the errors in the system as soon as possible.
* Flexibility in providing solutions to resolve the errors.
* Out of order (Executed only after code completion).
* Suitable for the constantly changing requirements.

## 2. Test Strategy

### 2.1 Testing Types

### 

| **Verification** | **Validation** |
| --- | --- |
| User requirement | Acceptance testing   * Check user interface to ensure front-end met requirement * Test load defect * Test performance defect |
| System requirement | System testing   * Test system functionality and the communication of system * Cover all hardware and software problems |
| Detail design | Unit testing   * System run correctly * Cover all defect |

*Table 219. Testing types*

### 2.2 Test Levels

### 

| **Testing Levels** | **Description** |
| --- | --- |
| Unit testing | Unit testing is developed during the module design phase, executed to detect and limit bugs at function level or unit level |
| System testing | System testing is developed during the system design phase. System testing ensures that project expectations are met. |
| Acceptance testing | Acceptance testing is developed during the requirements analysis phase. Our team will be testing on a real server with real data. Acceptance testing ensures that our system meets the requirement to use in real life |

*Table 218. Testing levels*

### 2.3 Supporting Tools

N/A

## 3. Test Plan

### 3.1 Human Resources

#### 

| **Worker/Doer** | **Role** | **Specific Responsibilities/Comments** |
| --- | --- | --- |
| SangNT(se140026) | Dev, Tester | Unit Test, System Test |
| SangNT(se130221) | Dev, Tester | Unit Test, System Test, User Acceptance Test |
| HuyTH | Dev, Tester | Unit Test, System Test, User Acceptance Test |
| NamTH | Dev, Tester | Unit Test, System Test, User Acceptance Test |

*Table 221. Human resources*

### 3.2 Test Environment

#### 

| **Purpose** | **Tool** | **Provider** | **Version** |
| --- | --- | --- | --- |
| Functional Test | VSCode | Microsoft | 9.29.0 |
| User Interface Test | Google Chrome | Google | 104.0.5112.101 |

*Table 222. Environment*

### 3.3 Test Milestones

### 

| **Milestone Task** | **Effort (md)** | **Start Date** | **End Date** |
| --- | --- | --- | --- |
| Create test plan | 1 | 10/09/2022 | 12/09/2022 |
| Review and update test plan (Iterator 1) | 3 | 20/10/2022 | 23/10/2022 |
| Create test case (Iterator 1) | 3 | 23/10/2022 | 26/10/2022 |
| Execute unit test (Iterator 1) | 3 | 26/10/2022 | 29/10/2022 |
| Execute system test (Iterator 1) | 3 | 29/10/2022 | 30/10/2022 |
| Execute acceptance test (Iterator 1) | 2 | 31/10/2022 | 1/11/2022 |
| Review and update test plan (Iterator 2) | 3 | 1/11/2022 | 4/11/2022 |
| Create test case (Iterator 2) | 3 | 4/11/2022 | 7/12/2022 |
| Execute unit test (Iterator 2) | 3 | 10/11/2022 | 13/11/2022 |
| Execute system test (Iterator 2) | 3 | 13/11/2022 | 16/11/2022 |
| Execute acceptance test (Iterator 2) | 4 | 16/11/2022 | 20/11/2022 |

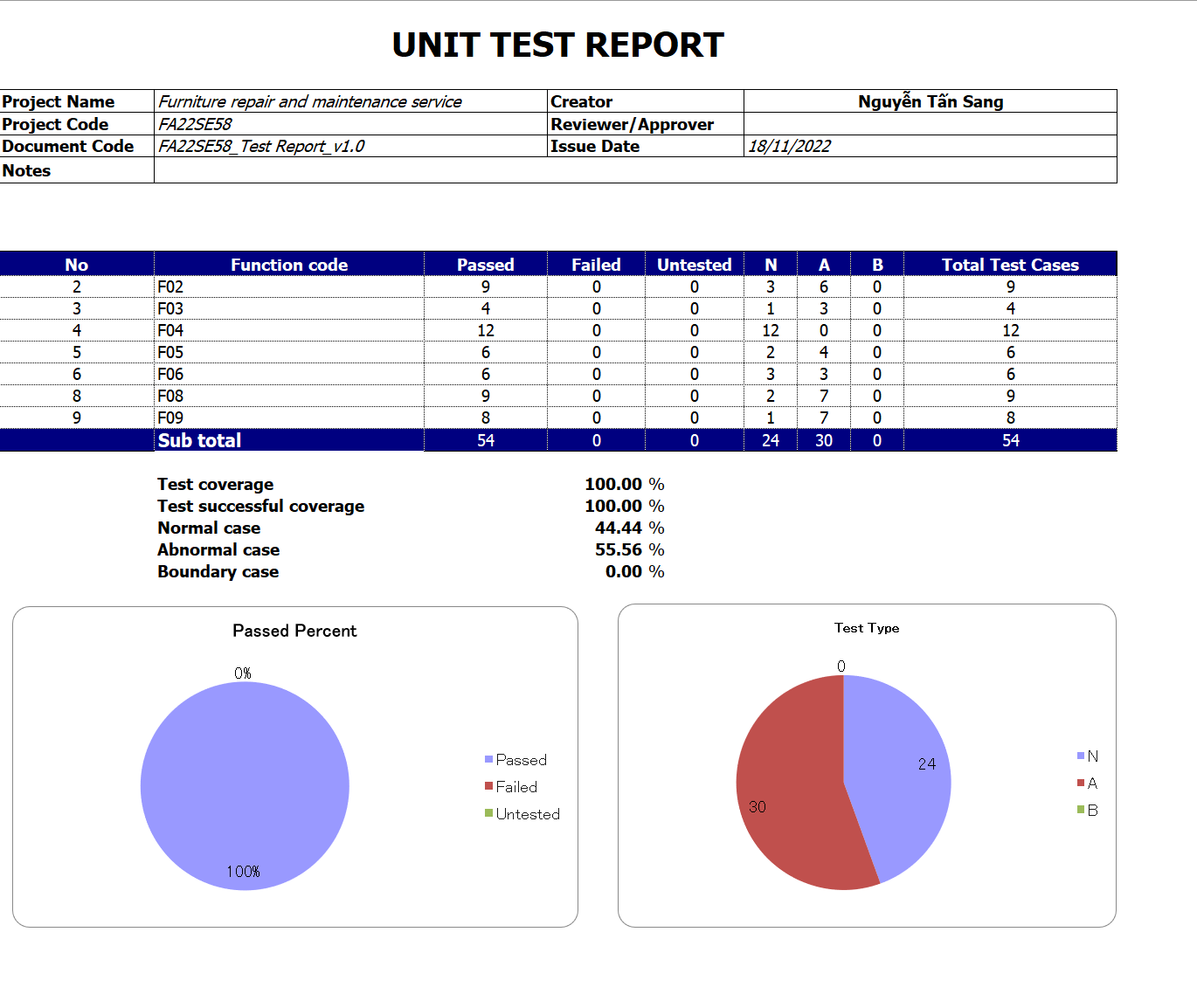
*Table 223. Test milestones*

## 4. Test Cases

* Unit Test Cases: [Report5\_Unit Test Case.xlsx](https://docs.google.com/spreadsheets/d/1qy_rUGMqxMhY1yzJfrI1zpVk38vXGYX_/edit?usp=share_link&ouid=112679804450864610045&rtpof=true&sd=true)
* Other Test Cases: [Report5\_Test Case Document.xlsx](https://docs.google.com/spreadsheets/d/1EShZ6iLyj3LkHcBp1Lqc5mCuDgiDUZ83/edit?usp=share_link&ouid=112679804450864610045&rtpof=true&sd=true)

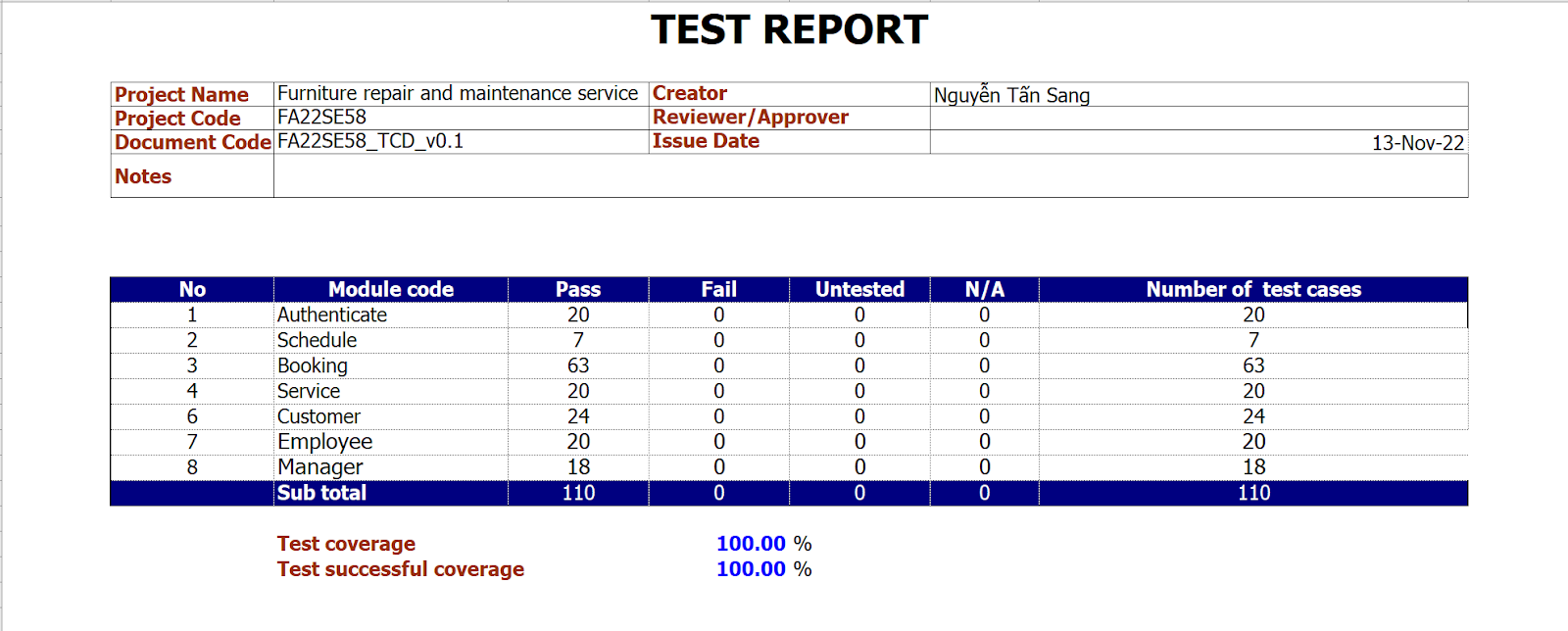
## 5. Test Reports

### 5.1 Unit Test



*Figure 131. Unit test*

### 5.2 Test Documents



*Figure 132. Test case document*

# VI. Release Package & User Guides

## 1. Deliverable Package

### 1.1 Source codes & documents

| **No.** | **Items** | **Sub-Items** | **Type** | **Version** |
| --- | --- | --- | --- | --- |
| ***Code Package*** | | | | |
| 1 | SV | Back end Server | New |  |
| 2 | WA | Web app for Manager | New |  |
| 3 | MA | Mobile app for Staff and Customer | New |  |
| ***Documents*** | | | | |
| 1 | Introduction | Project Introduction.docx | Modify |  |
| 2 | Management | Project Management Plan.docx | Modify |  |
| 3 | Requirement | System Requirement Specification.docx | Modify |  |
| 4 | Designing | Software Design Document.docx | Modify |  |
| 5 | Testing | Test Case Document.xlsx, Test Documentation.docx, Unit Test Case.xlsx | Modify |  |
| 6 | Guiding | Software User Guides.docx | Modify |  |

*Table 225. Deliverable source codes & documents*

### 

### 1.2 Known Issues, Limitations & Restrictions

Limitations:

* The system does not support multiple languages
* The system hasn’t supported online payment now. We will develop it in the future
* The system hasn’t supported multiple brands of company. We will develop it in future
* The system hasn’t supported map for customer and employee for searching
* The system hasn’t supported OTP verify

Restrictions:

* System still need manager to assign employee manually
* Booking process still have some limit (take time, not automatically accept/deny)

## 2. Installation Guides

### 2.1 System Requirements

#### a. For PC

|  | **Minimum Requirements** | **Recommended** |
| --- | --- | --- |
| **Internet Connection** | Cable, Wi-Fi (8 Mbps) | Cable, Wi-Fi (50 Mbps or more) |
| **Operating System** | Window 10 | Window 10 |
| **Computer Processor** | Intel Core i3-4010U | Intel Core i3-10100TE |
| **Computer Memory** | 4GB RAM | 16GB or more |
| **Storage Space** | 2GB | 10GB or more |

*Table 226. PC system requirements*

#### b. For Android

|  | **Minimum Requirements** | **Recommended** |
| --- | --- | --- |
| **Internet Connection** | 3G, Wi-Fi (2 Mbps) | 4G, Wi-Fi (50 Mbps or more) |
| **Operating System** | Android 10 | Android 11 and upper |
| **Storage Space Require** | 1 GB | 2 GB |

*Table 227. Android system requirements*

### 2.2 Installation Instruction

#### a. Mobile

To use Computer services application, download fms-setup.apk files attached from this document to an android device, or setup from github source code by steps below:

* Install dart from [Dart programming language | Dart](https://dart.dev/)
* Download source code from github: <https://github.com/COCONUC/FMS_Employee_App>
* Open source code folder, open Command prompt and execute command:
  + flutter build apk --release
* Navigate to:
  + build\app\outputs\flutter-apk\
* Copy fms-setup.apk (app-release.apk) to your phone

Open the setup file, and click the “Install” button.

#### b. Front End

* Set up Front End in local:
  + Install React.js from: <https://reactjs.org/docs/create-a-new-react-app.html>
  + Download and unzip front end project folder (computer-services)
  + Open command line (cmd) in computer-services folder and execute these commands:
    - npm install
    - npm start
* Deploy Front End in Netlify server:
  + Download and unzip front end project folder (computer-services)
  + Push source code Github repository
  + Create an account from <https://www.netlify.com/>
  + Connect to Github provider
  + Choose your repository, deploy branch and confirm deploy site

#### c. Back End

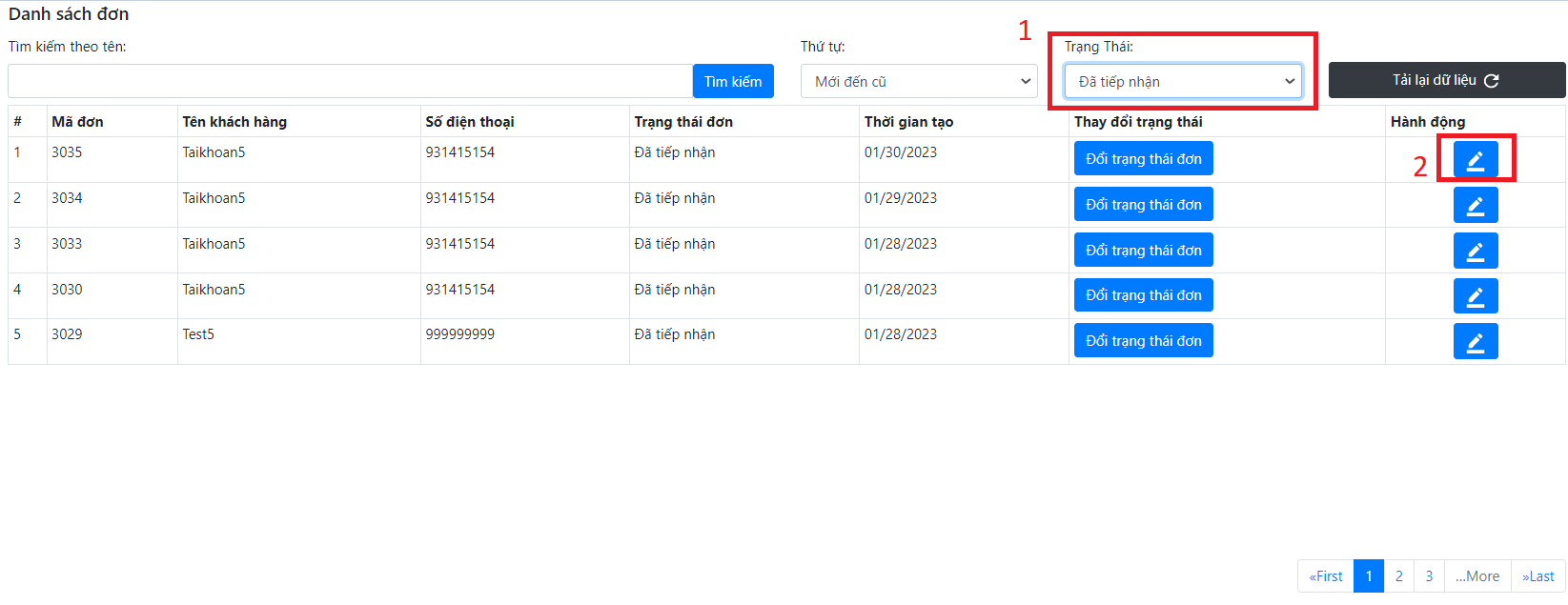
* Set up Back End in local:
  + Install Node.js from <https://nodejs.org/en/>
  + Download and unzip back end project folder (project)
  + Open command line (cmd) in project folder and execute these commands:
    - npm install
    - npm start
* Deploy Back End in Heroku server:
  + Download and unzip back end project folder (project)
  + Push source code Github repository
  + Create an account from <https://www.heroku.com/>
  + Create new app at dashboard
  + Connect your Github provider
  + Choose your repository, deploy branch and confirm deploy site

## 3. User Manual

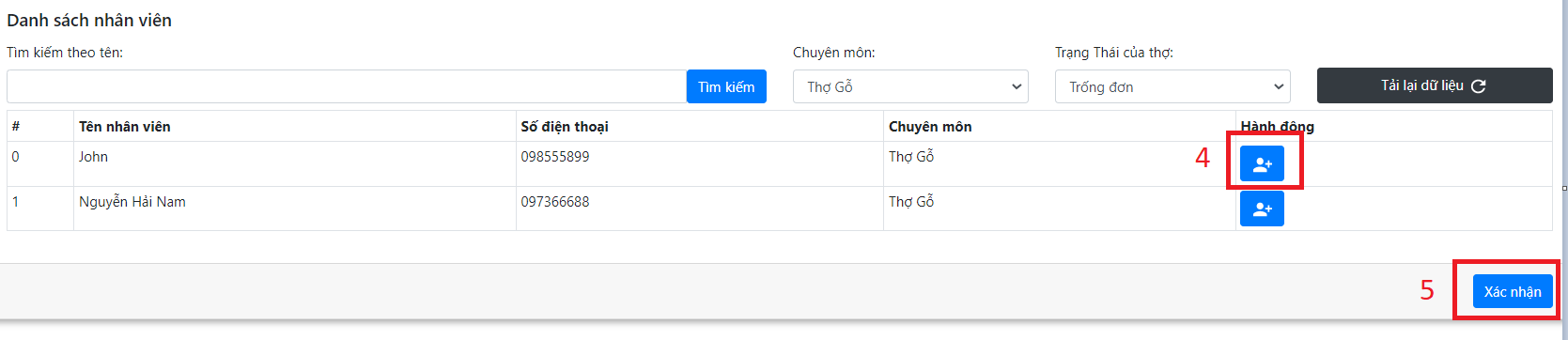
### 3.1 Overview

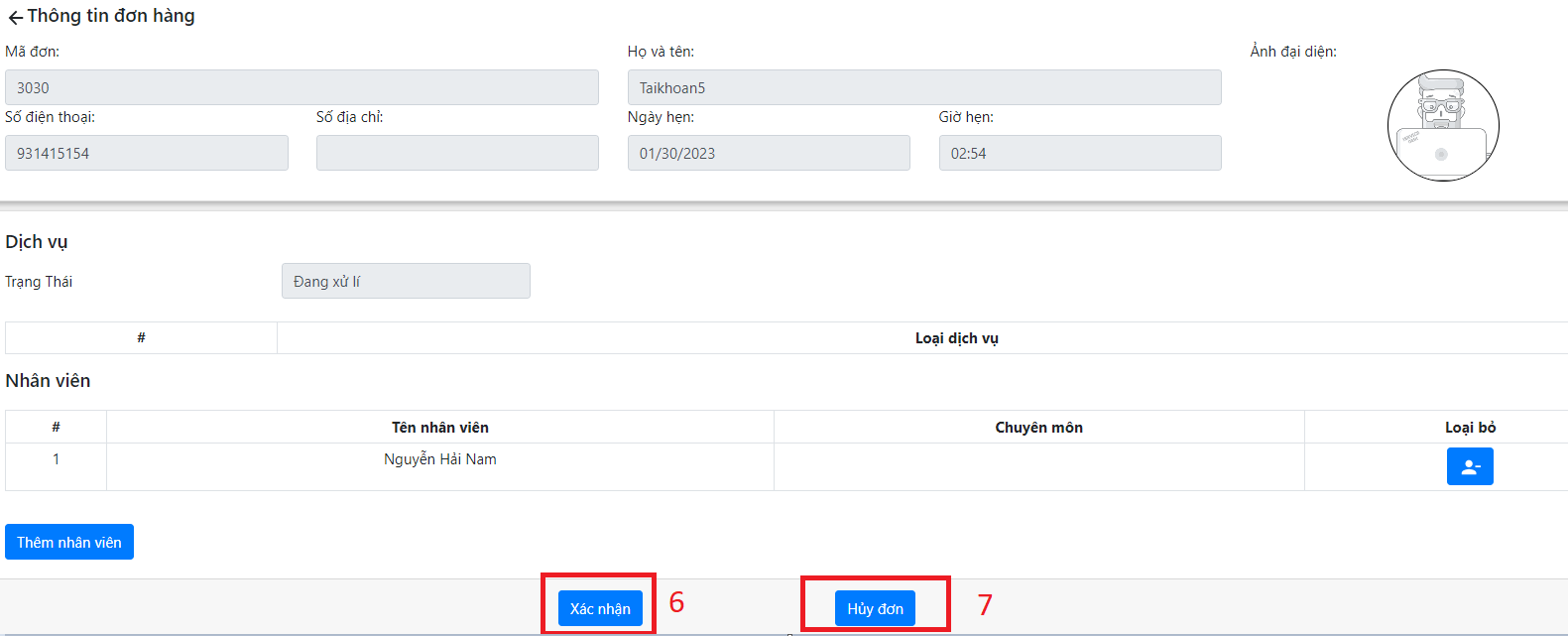
We provide a customer mobile application to be able to book repair and maintenance service and a staff mobile application to receive assigned booking and also send detailed reports to manager . Finally, the web admin system to manage and handle requests from both applications. Detailed instructions on how to use it are described below.

### 3.2 Assign Staff

**

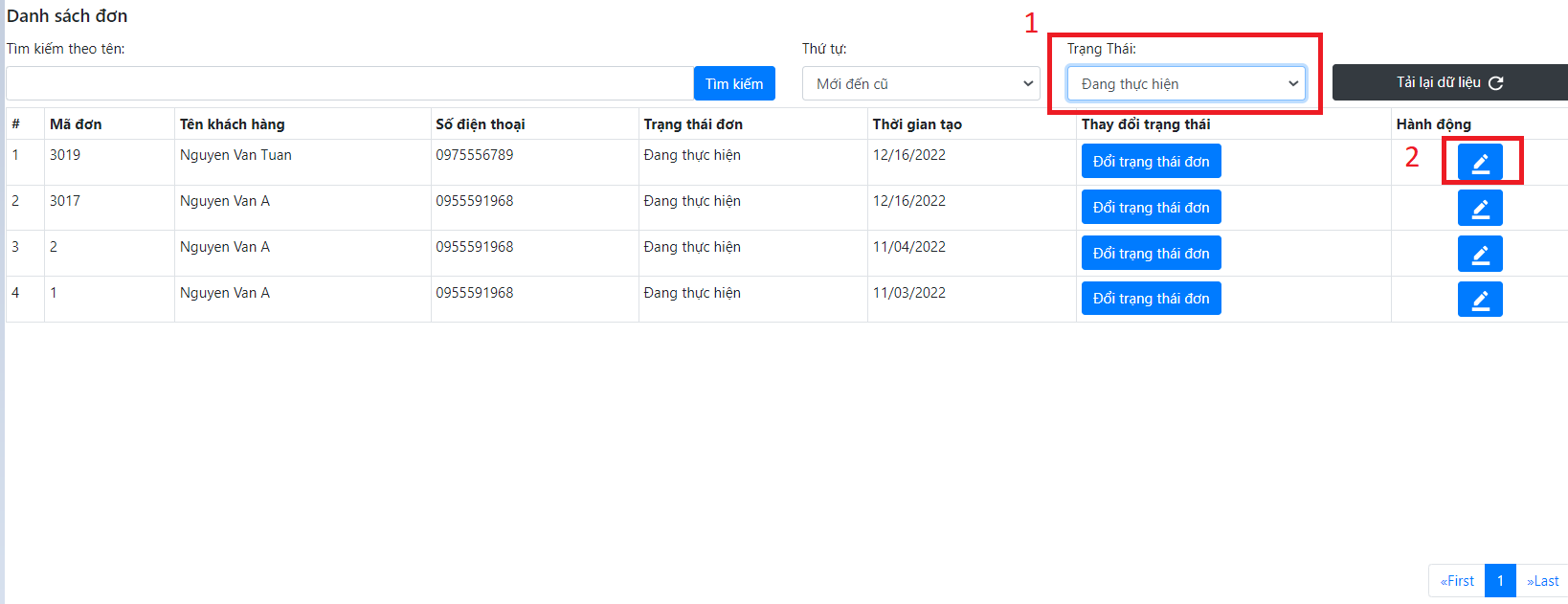
**

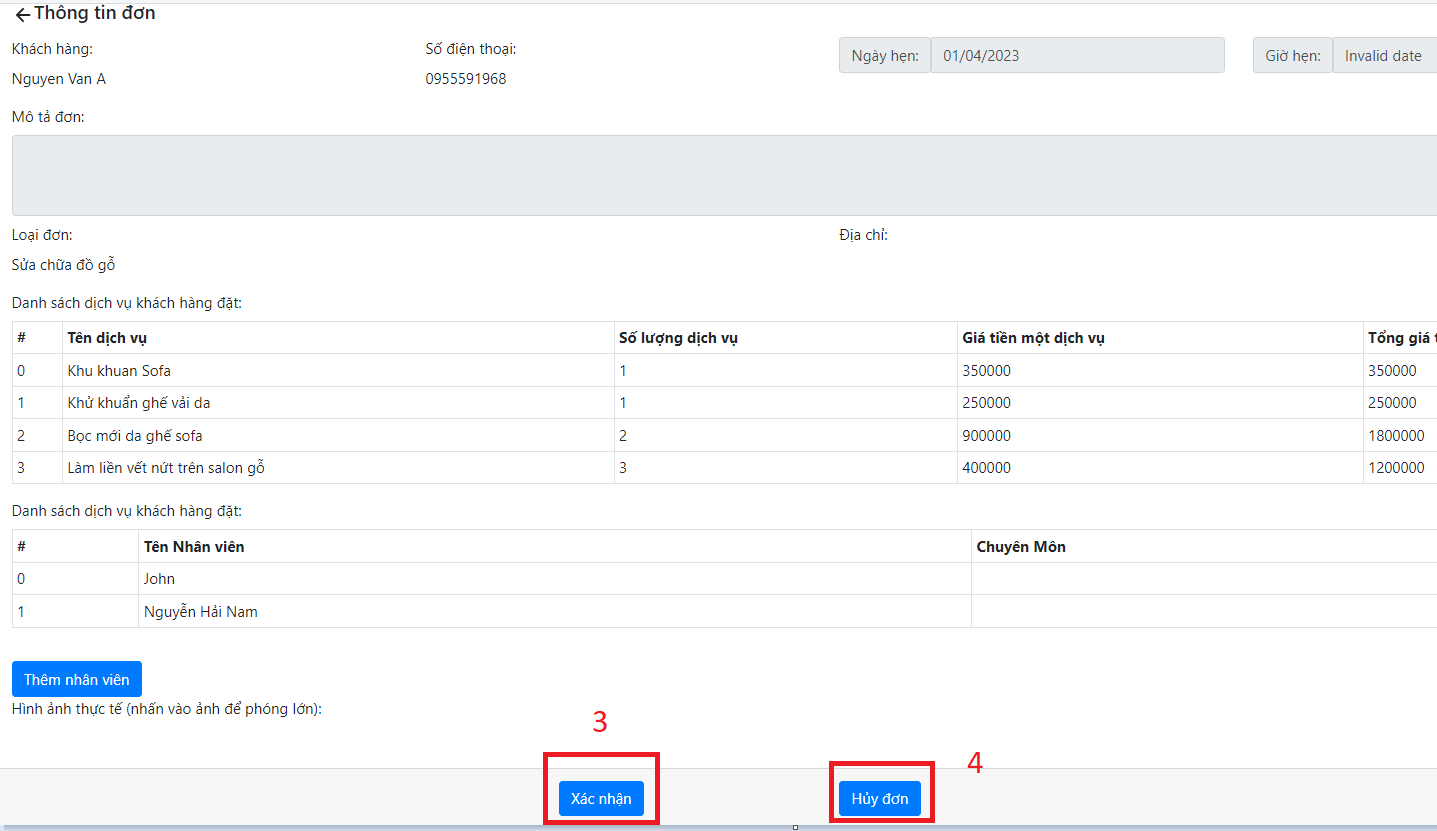
**

**

| No. | Step | Note |
| --- | --- | --- |
| 1 | Select order status “Đã tiếp nhận” | required |
| 2 | Select order | required |
| 3 | Select add staff | required |
| 4 | Select staff | required |
| 5 | Confirm select staff | required |
| 6 | Click “Xác nhận” to confirm | required |
| 7 | Click “Hủy đơn” to cancel | mandatory |

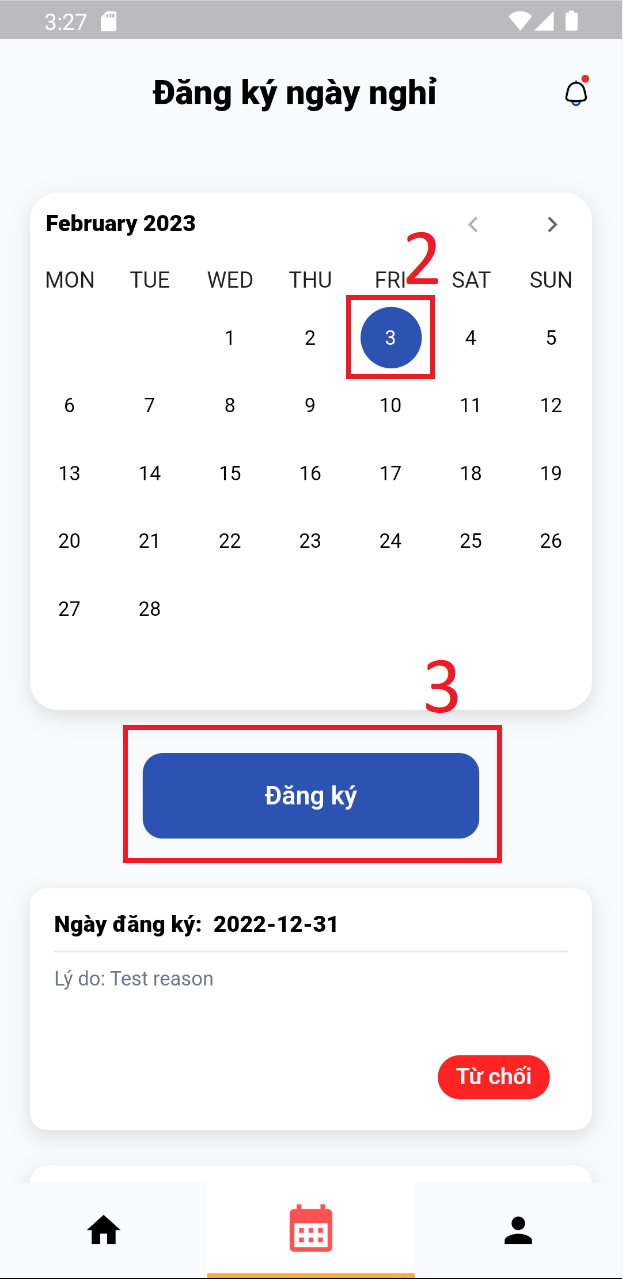
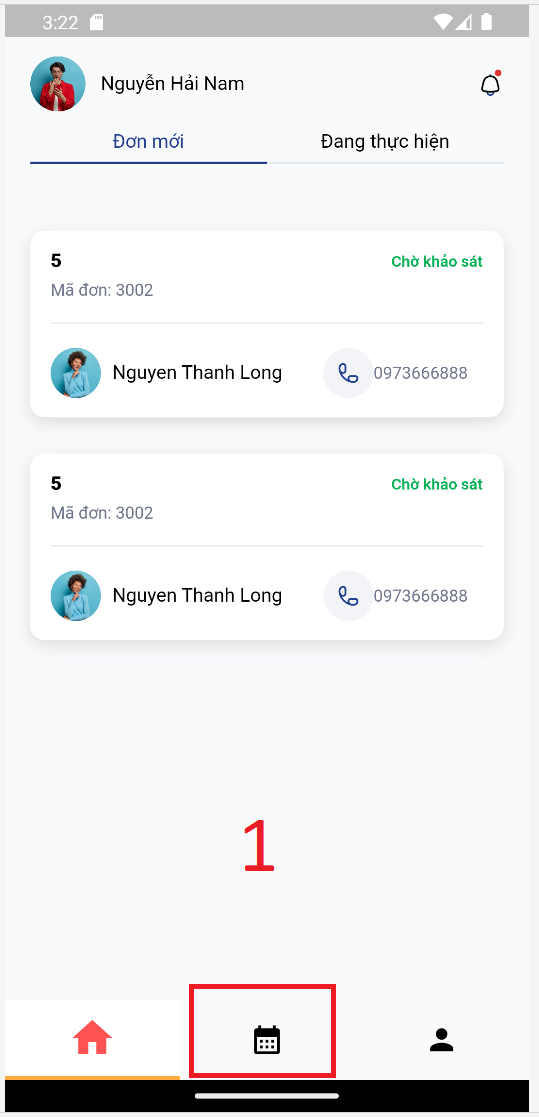
### 3.3 Approved order

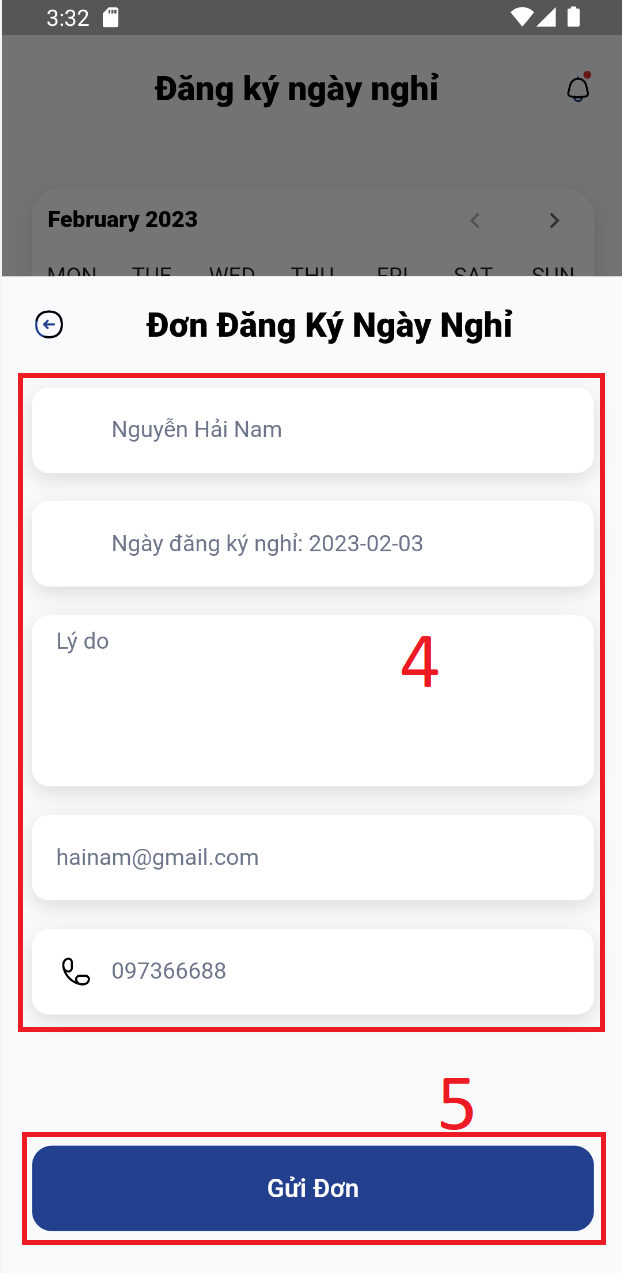




| No. | Step | Note |
| --- | --- | --- |
| 1 | Select order status “Đang thực hiện” | required |
| 2 | Select order | required |
| 3 | Confirm order | required |
| 4 | Cancel order | mandatory |

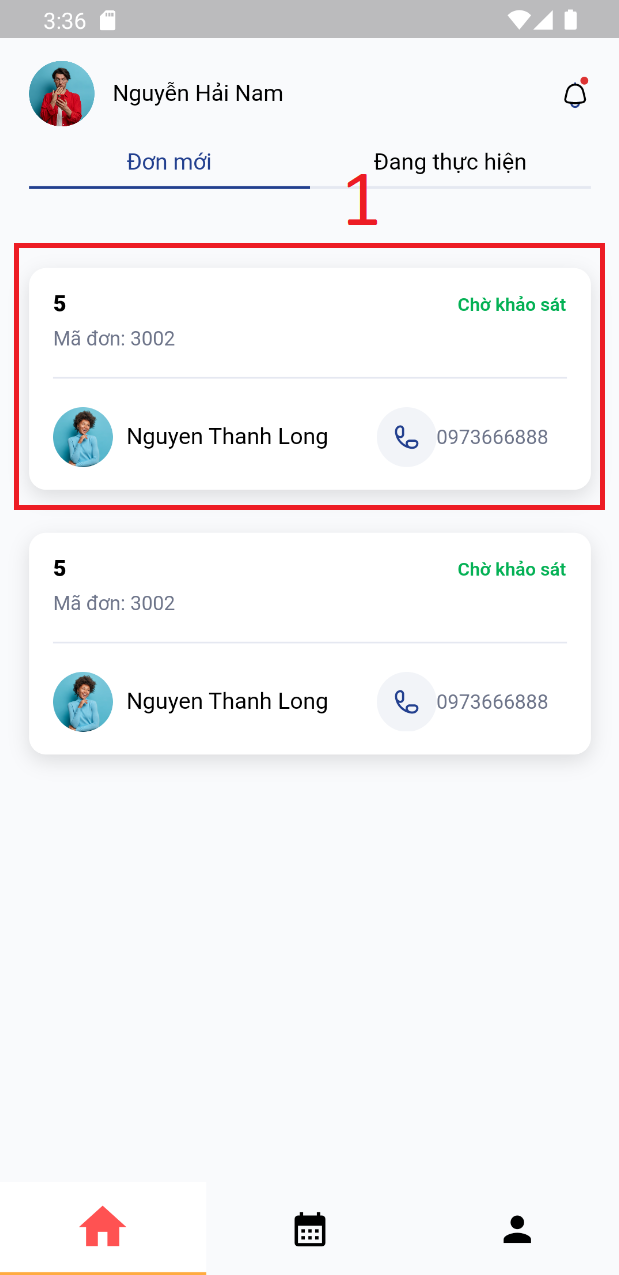
### 3.4 Staff send day off request

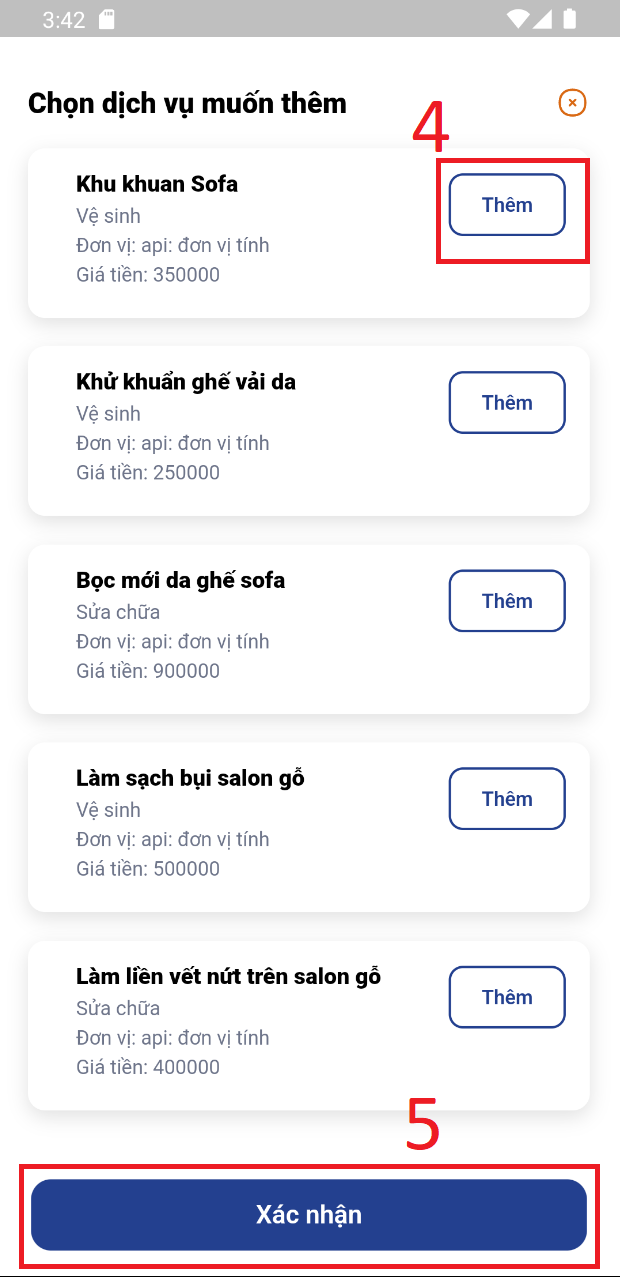
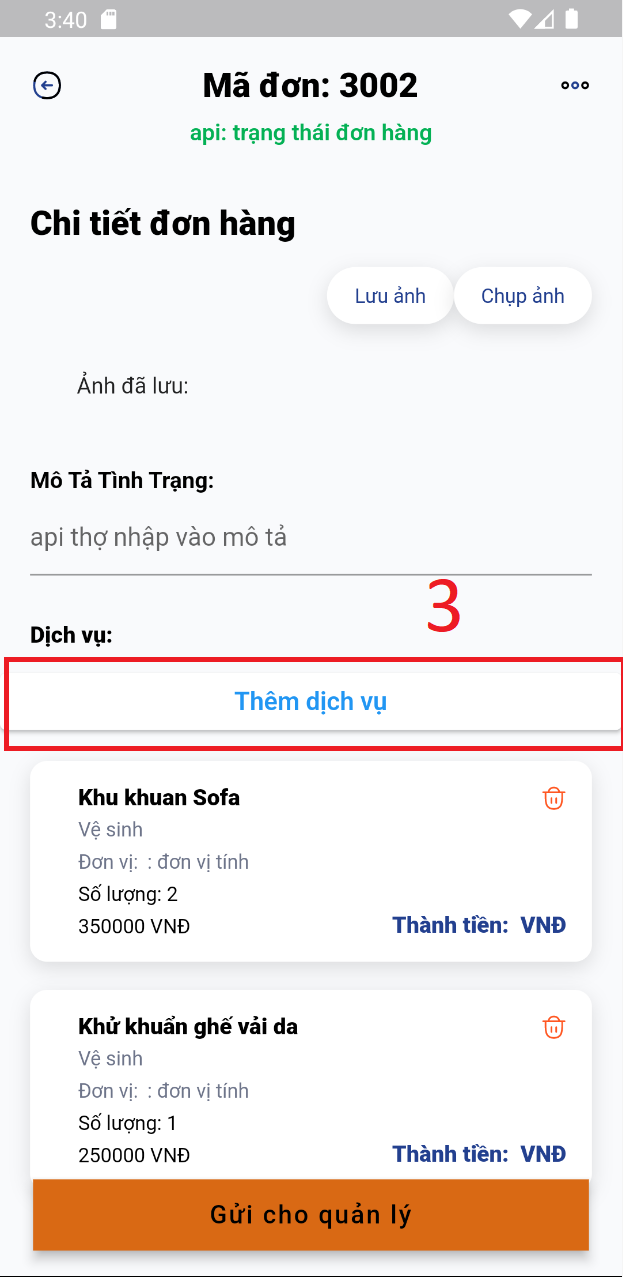


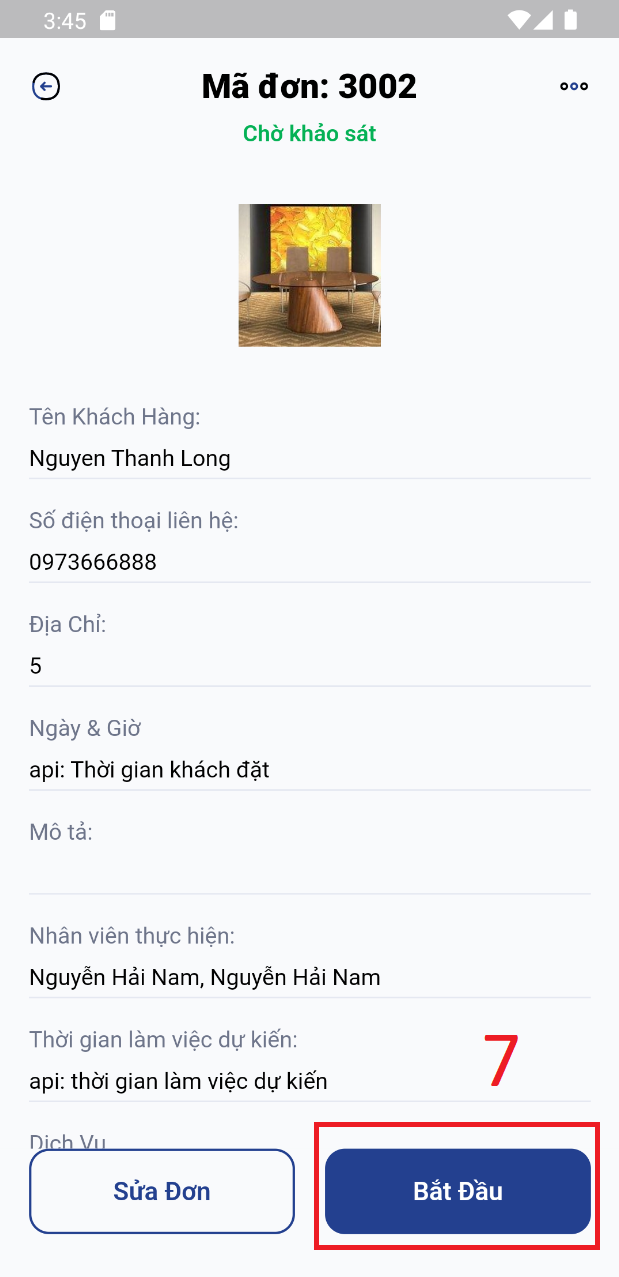
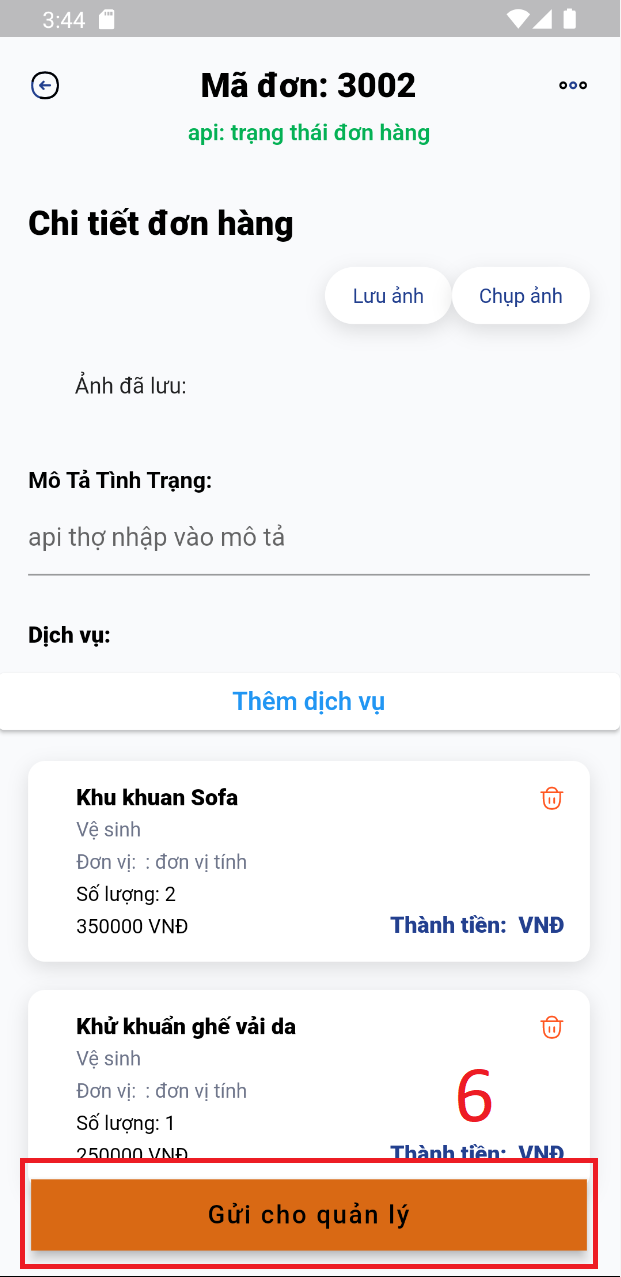


| No. | Step | Note |
| --- | --- | --- |
| 1 | Select timetable icon | required |
| 2 | Select day wanted to request | required |
| 3 | Confirm selection | required |
| 4 | Fill required fields | mandatory |
| 5 | Send request | mandatory |

### 3.5 Staff fulfil order



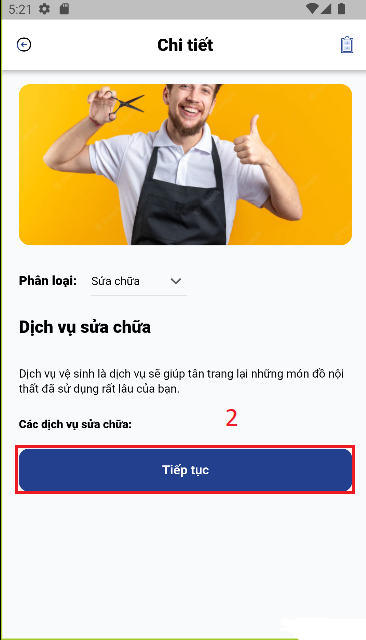
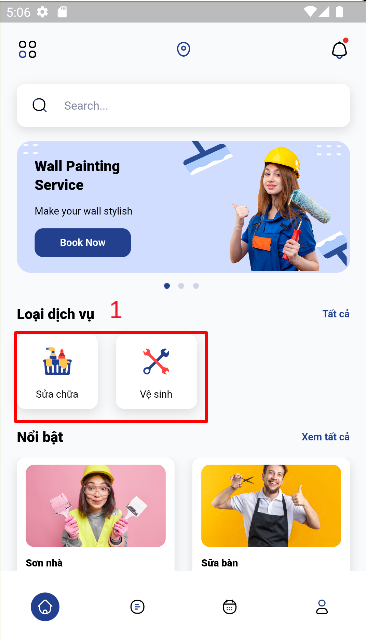


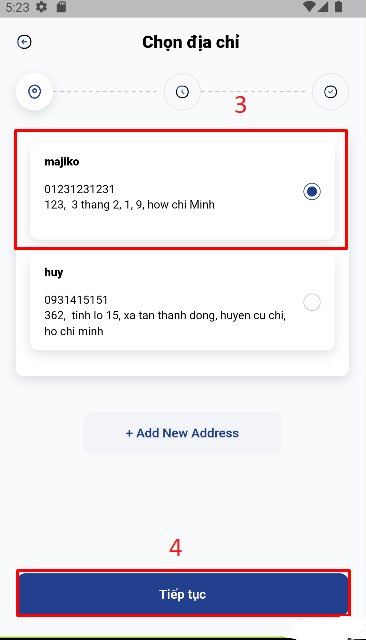


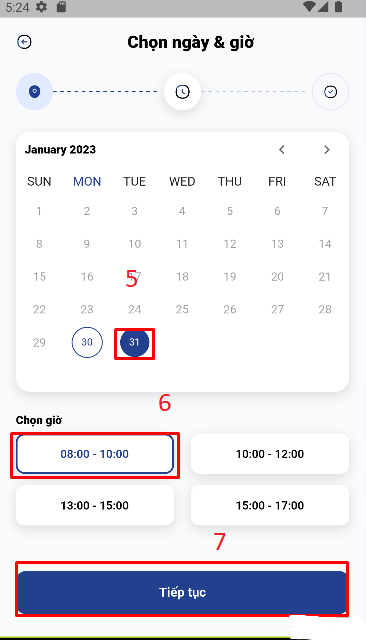
| No. | Step | Note |
| --- | --- | --- |
| 1 | Select order | required |
| 2 | Select survey button | required |
| 3 | Select add service button | optional |
| 4 | Select service | optional |
| 5 | Confirm selection | required |
| 6 | Select send to manager button | mandatory |
| 7 | Select start job button | mandatory |

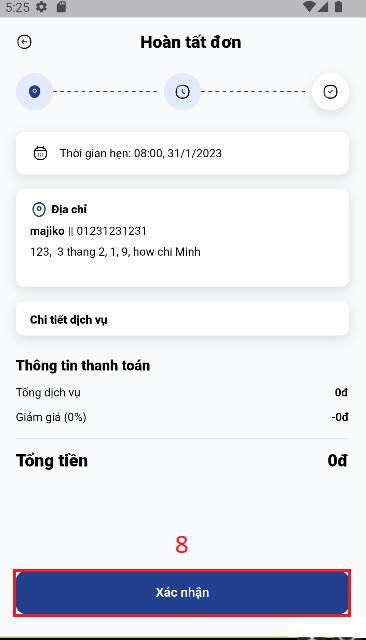
### 

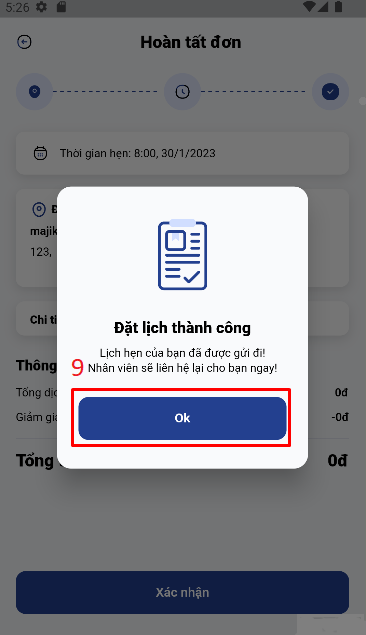
### 3.6 Customer create a booking











| No. | Step | Note |
| --- | --- | --- |
| 1 | Select service | required |
| 2 | Select next button | required |
| 3 | Select an address | required |
| 4 | Select next button | required |
| 5 | Select date | required |
| 6 | Select Time | required |
| 7 | Click next button | required |
| 8 | Click confirm button | required |
| 9 | Click Ok button | required |

### 