



# FPT UNIVERSITY

## Capstone Project Document

### VinFlat - Dormitory management system for VinFlat modal

GSP23SE06	
Group Members	Đào Bảo Trâm - tramdbse140878@fpt.edu.vn Phạm Minh Đức - ducpmse140445@fpt.edu.vn Trần Khải Minh Khôi - khoitkmse150850@fpt.edu.vn (Drop) Dương Đình Nguyên - nguyenddse62732@fpt.edu.vn (Drop)
Supervisor	Lâm Hữu Khánh Phương Nguyễn Thế Hoàng
Ext Supervisor	N/A
Capstone Project code	SP23SE35

- HCM, 05/2023 -

## Table of Contents

Table of Contents .....	2
List of Table .....	7
Table of figures .....	8
Acknowledgement .....	9
Definition and Acronyms .....	10
I. Project Introduction .....	11
1. Overview .....	11
1.1 Project Information .....	11
1.2 Project Team .....	11
1.2.1 Supervisor .....	11
1.2.2 Team member .....	11
2. Product Background .....	11
3. Existing Systems .....	11
3.1 PMSS .....	11
3.2 Itro .....	12
Link: <a href="http://quanlynhatro.com">quanlynhatro.com</a> .....	12
3.3 Nha tro sach se .....	12
4. Business Opportunity .....	12
5. Software Product Vision .....	12
6. Project Scope & Limitations .....	12
6.1 Major Features .....	12
6.1.1 Web application for Admin .....	12
6.1.2 Web application for Supervisor .....	12
6.1.3 Mobile application for Technician .....	13
6.1.4 Mobile application for Renter .....	13
6.2 Limitations & Exclusions .....	14
II. Project Management Plan .....	15
1. Overview .....	15
1.1 Scope & Estimation .....	15
1.2 Project Objectives .....	18
1.3 Project Risks .....	18
2. Management Approach .....	19
2.1 Project Process .....	19
2.2 Quality Management .....	19
2.3 Training Plan .....	20

3. Project Deliverables .....	20
4. Responsibility Assignments.....	21
5. Project Communications.....	21
6. Configuration Management.....	22
6.1 Document Management .....	22
6.2 Source Code Management.....	22
6.3 Tools & Infrastructures .....	22
III. Software Requirements Specification.....	23
1. Product Overview.....	23
2. User Requirements .....	24
2.1 Actors .....	24
2.2 Use Cases .....	25
2.2.1 Diagram(s).....	25
2.2.2 Use Case Description.....	26
3. Functional Requirements.....	28
3.1 System Functional Overview.....	28
3.1.1 Screens Flow .....	28
a. Web admin .....	28
b. Web supervisor .....	28
c. Mobile technician.....	29
d. Mobile renter .....	29
3.1.2 Functional requirements.....	30
3.1.2.1 Admin functional requirements.....	30
3.1.2.2 Supervisor functional requirements .....	31
3.1.2.3 Technician functional requirements .....	43
3.1.2.4 Renter functional requirements .....	44
3.1.2.5 Member functional requirements .....	49
3.1.3 Screen Authorization.....	52
3.1.3.1 Web Application Screen Authorization.....	52
3.1.3.1 Mobile Application Screen Authorization.....	52
3.1.4 Non-Screen Functions .....	52
3.2 Entity Relationship Diagram.....	53
4. Non-Functional Requirements.....	53
4.1 External Interfaces .....	53
4.2 Quality Attributes.....	54
4.2.1 Usability.....	54
4.2.2 Reliability.....	54

4.2.3 Performance.....	54
4.2.4 Security.....	54
5. Requirement Appendix .....	54
5.1 Business Rules .....	54
5.2 Common Requirements .....	57
IV. Software Design Description.....	61
1. System Design .....	61
1.2 Package Diagram.....	62
.....	62
2. Database Design.....	63
3. Detailed Design .....	80
3.1 Admin Login Feature .....	80
3.1.1 Class diagram .....	80
3.1.2 Class specification .....	80
3.1.3 Login Activity Diagram .....	81
3.1.4 Login Sequence Diagram.....	82
3.2 Admin Manage Building Feature.....	83
3.2.1 Class Diagram .....	83
3.2.2 Class Diagram Specification .....	83
3.2.3 Supervisor Create Building Activity Diagram .....	84
3.2.4 Supervisor Create Building Sequence Diagram.....	85
3.3 Supervisor Manage Flat / Room.....	86
3.3.1 Class Diagram .....	86
3.3.2 Class Diagram Specification .....	86
3.3.3 Supervisor Create Flat Activity Diagram .....	87
3.3.4 Sequence Create Flat Activity Diagram.....	88
3.4 Supervisor Manage Contract .....	89
3.4.1 Class Diagram .....	89
3.4.2 Class Diagram Specification .....	89
3.4.3 Supervisor Create Renter by Contract Activity Diagram.....	90
3.4.4 Sequence Create Renter by Contract Activity Diagram .....	91
3.5 Supervisor Renew Contract.....	91
3.5.1 Class Diagram .....	91
3.5.2 Class Diagram Specification .....	92
3.5.3 Supervisor Renew Contract Activity Diagram .....	93
3.5.4 Sequence Renew Contract Activity Diagram .....	94
3.6 Supervisor View Contract.....	95

3.6.1 Class Diagram .....	95
3.6.2 Class Diagram Specification .....	95
3.6.3 Supervisor View Contract Activity Diagram .....	96
3.6.4 Supervisor View Contract Sequence Diagram .....	97
3.7 Supervisor Manage Ticket .....	98
3.7.1 Class Diagram .....	98
3.7.2 Class Diagram Specification .....	98
3.7.3 Supervisor Edit Ticket Activity Diagram .....	99
3.7.4 Supervisor Edit Ticket Sequence Diagram .....	100
3.8 Supervisor Manage Invoice .....	<b>Error! Bookmark not defined.</b>
3.8.1 Class Diagram .....	<b>Error! Bookmark not defined.</b>
3.8.2 Class Diagram Specification .....	<b>Error! Bookmark not defined.</b>
3.8.3 Supervisor Create Invoice Activity Diagram .....	<b>Error! Bookmark not defined.</b>
3.8.4 Supervisor Create Invoice Sequence Diagram .....	<b>Error! Bookmark not defined.</b>
3.9 State Machine Diagram .....	100
3.9.1 Ticket State Diagram .....	100
3.9.3 State Room Diagram .....	101
V. Software Testing Documentation .....	101
1. Scope of Testing .....	101
1.1 Testing model .....	101
1.2 Testing level .....	102
2. Test Strategy .....	102
2.1 Testing Types .....	102
2.2 Test Levels .....	102
2.3 Supporting Tools .....	102
3. Test Plan .....	102
3.1 Human Resources .....	102
3.2 Test Environment .....	102
3.3 Test Milestones .....	103
4. Test Cases .....	103
5. Test Reports .....	103
VI. Release Package & User Guides .....	105
1. Deliverable Package .....	105
2. Installation Guides .....	105
2.1 System Requirements .....	105
2.1.1 Hardware requirements .....	105
2.1.1.1 Web Application .....	105

2.1.1.2 Mobile Application.....	105
2.1.2 Software requirements .....	106
2.3 Installation Instruction .....	106
2.3.1 Setup Environments.....	107
2.3.1.1 Flutter.....	107
2.3.1.2 Android Studio .....	110
3. User Manual.....	<b>Error! Bookmark not defined.</b>
3.1 Overview .....	<b>Error! Bookmark not defined.</b>
3.2 Workflow 1.....	<b>Error! Bookmark not defined.</b>
3.3 Workflow 2.....	<b>Error! Bookmark not defined.</b>

## List of Table

Table 1 - Definition and Acronyms.....	10
Table 2 - Supervisor.....	11
Table 3- Team member.....	11
Table 4- Scope & Estimation .....	18
Table 5- Project Risks .....	18
Table 6 - Training Plan.....	20
Table 7- Project Deliverables .....	20
Table 8 - Responsibility Assignments.....	21
Table 9 - Project Communications.....	21
Table 10 - Tools & Infrastructures .....	22
Table 11 - Actors .....	24
Table 12 - Use Case Description.....	27
Table 13 - Admin functional requirements.....	31
Table 14 - Supervisor functional requirements .....	43
Table 15 - Technician functional requirements .....	44
Table 16 - Renter functional requirements.....	49
Table 17 - Member functional requirements .....	51
Table 18 - Web Application Screen Authorization .....	52
Table 19 - Mobile Application Screen Authorization .....	52
Table 20 - Non-Screen Functions .....	52
Table 21 - Entity Relationship Diagram.....	53
Table 22- Business Rules.....	57
Table 23 - Application Messages List .....	60
Table 24 - Table Description.....	64
Table 25 - Attribute Data Dictionary .....	79

## Table of figures

Figure 1 - Project Process.....	19
Figure 2 - Product Overview Diagram.....	23
Figure 3 - Use Case Diagram .....	25
Figure 4 - Web admin screen flow .....	28
Figure 5 - Web supervisor screen flow .....	28
Figure 6 - Mobile technician screen flow .....	29
Figure 7 - Mobile renter screen flow .....	29
Figure 8 - System Design .....	61
Figure 9 - Package Diagram.....	62
Figure 10 - Database Diagram.....	63
Figure 11 - Login Activity Diagram .....	81
Figure 12 - Login Sequence Diagram.....	82
Figure 13 - Supervisor Create Building Activity Diagram .....	84
Figure 14 - Supervisor Create Building Sequence Diagram .....	85
Figure 15 - Supervisor Create Flat Activity Diagram .....	87
Figure 16 - Sequence Create Flat Activity Diagram.....	88
Figure 17 - Supervisor Create Renter by Contract Activity Diagram.....	90
Figure 18 - Sequence Create Renter by Contract Activity Diagram .....	91
Figure 19 - Supervisor Renew Contract Activity Diagram.....	93
Figure 20 - Sequence Renew Contract Activity Diagram .....	94
Figure 21 - Supervisor View Contract Activity Diagram .....	96
Figure 22 - Supervisor View Contract Sequence Diagram .....	97
Figure 23 - Supervisor Edit Ticket Activity Diagram .....	99
Figure 24 - Supervisor Edit Ticket Sequence Diagram .....	100
Figure 25 - Supervisor Create Invoice Activity Diagram.....	<b>Error! Bookmark not defined.</b>
Figure 26 - Supervisor Create Invoice Sequence Diagram .....	<b>Error! Bookmark not defined.</b>



## Acknowledgement

First, we would like to express our heartfelt gratitude to our supervisor, Mr. Nguyễn Thế Hoàng and Mr. Lâm Hữu Khánh Phương, for their guidance throughout the entire project. Their suggestions and recommendations have significantly improved the overall quality of the project.

As we complete this capstone project, we would also like to extend our appreciation to:

- FPT University lecturers who shared their invaluable knowledge with us and guided us as we embarked on our academic journey.
- Mr. Thân Văn Sử, Mr. Trần Thanh Nguyên, and all other lecturers who pointed out flaws in our project and provide constructive feedback to enhance its quality.
- We would like to express my sincere thanks to our family and our friends for always being there and supporting me through the toughest times.
- Finally, a special thanks to Mr. Hoàng and Mr. Phương, our direct supervisors who provide extensive support in areas such as technology, business, and documentation. Last but not least, we want to thank our family and friends who supported us throughout this project.

We acknowledge that due to our lack of experience and unfamiliarity with certain technologies used in the project, mistakes were inevitable. We hope that people will overlook and forgive any errors or shortcomings. Thank you for your unwavering support.

Regards.

## Definition and Acronyms

Acronym	Definition
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

*Table 1 - Definition and Acronyms*

# I. Project Introduction

## 1. Overview

### 1.1 Project Information

- Project name: Build a dormitory management system for VinFlat modal
- Project code: SP23SE35
- Group name: GSP23SE06
- Software type: Web Application, Mobile Application

### 1.2 Project Team

#### 1.2.1 Supervisor

Full Name	Role	Email	Mobile
Lam Huu Khanh Phuong	Lecturer	phuonglkh@fpt.edu.vn	0915353001
Nguyen The Hoang	Lecturer	hoangnt@fpt.edu.vn	

Table 2 - Supervisor

#### 1.2.2 Team member

Full Name	Role	Email	Mobile
Dao Bao Tram	Leader	tramdbse140878@fpt.edu.vn	0934972646
Pham Minh Duc	Member	ducpmse140445@fpt.edu.vn	0917878438
Tran Khai Minh Khoi	Member	khoitkmse150850@fpt.edu.vn	0903185430
Duong Dinh Nguyen	Member	nguyenddse62732@fpt.edu.vn	0776148927

Table 3- Team member

## 2. Product Background

VinFlat is a dormitory model operating in the form of renting a place in an apartment, with some equipment and services available.

It also helps VinFlat's staff manage and operate rental utility service activities, the renter's needs should be handled quickly, clearly, and correctly.

Based on those needs, we built a web management system for VinFlat's staff and a mobile app to fulfill renters' needs.

## 3. Existing Systems

### 3.1 PMSS

Link: [Property Management System Solutions](#)

PMSS is the solution suitable for many models of property management & operation, which researched and developed. PMSS provides operation management solutions through the property system software diversely. Built and developed from the ERP platform, digitalize processes and documents to simplify and professionalize the property management process.

### 3.2 Itro

Link: [quanlynhatro.com](http://quanlynhatro.com)

ITRO is a complete ecosystem in rental housing including management software, tenant finder software for landlords, and home search software for renters.

### 3.3 Nha tro sach se

Link: [nhatrosachse.com](http://nhatrosachse.com)

As a company that invests and operates hostels for students and low-income workers. Includes web for renters to view information. Mobile App for the hostel's manager to use in the Nhatrosachse system.

## 4. Business Opportunity

The client requires three main parts:

1. Web-based software system to help VinFlat's staffs manage dormitory activities, such as managing renters, dorm rooms, finance, room-by-room assets, incidents, etc.,
2. Mobile software so that renters can easily use the services provided by VinFlat, and track information related to their rental.
3. Mobile software so that technicians can easily use related functions, view requests from renters and perform their work on the system.

## 5. Software Product Vision

The system was born to help VinFlat:

1. Solve problems in managing the dormitory rental process, effectively manage renter information, and assist in managing and archiving contracts, invoices, and time schedules at different locations with optimal solutions. The system creates convenience, saves time, and minimizes errors compared to traditional management.
2. Solve the problem of demands from the renters - dormitory people.
3. Solve the problem of technicians - technical staff in the system

## 6. Project Scope & Limitations

### 6.1 Major Features

#### 6.1.1 Web application for Admin

As a user in the position of an admin using web application, I would like to use the following features:

**FE-01.** Login/Logout.

**FE-02.** Manage personal profile: View profile, update profile, change password.

**FE-03.** Manage employee's account: view employee profile, add employee account, ban/unban account.

**FE-04.** Manage areas: view area, update area, add new area.

#### 6.1.2 Web application for Supervisor

As a user in the position of a supervisor using web application, I would like to use the following features:

**FE-05.** Login/Logout.

**FE-06.** Manage personal profile: View profile, update profile, change password.

**FE-07.** Manage theirs building: view building information, update building.

**FE-08.** Manage flat: view flat information, add new flat, update flat information.

- FE-09.** Manage room: view room information, add new room, update room information.
- FE-10.** Manage flat type: view flat type information, add new flat type, update flat type information.
- FE-11.** Manage room type: view room type information, add new room type, update room type information.
- FE-12.** Manage contract: view list contract, add new contract, update contract information.
- FE-13.** Manage renter: view list renter account, add renter account, update renter account status.
- FE-14.** Manage invoice: view invoice information, add new invoice, update invoice information.
- FE-15.** Manage ticket: view ticket information, update ticket status.
- FE-16.** Manage service: view service information, add new service, update service information.
- FE-17.** View list statistic

### ***6.1.3 Mobile application for Technician***

As a user in the position of a technician using mobile application, I would like to use the following features:

- MO-01.** Login/Logout.
- MO-02.** Manage personal profile: View profile, update profile, change password.
- MO-03.** View list tickets and detail tickets, and can select ticket(s) from list to solve.

### ***6.1.4 Mobile application for Renter***

As a user in the position of a renter using mobile application, I would like to use the following features:

- MO-04.** Login/Logout.
- MO-05.** Manage personal profile: View profile, update profile, change password.
- MO-06.** View their personal contract information.
- MO-07.** View their flat information.
- MO-08.** View list invoice information.
- MO-09.** View service, add service to invoice.
- MO-10.** Manage ticket: view list ticket, add new ticket, update ticket information, cancel ticket when hasn't been received.

## **6.2 Limitations & Exclusions**

### Limitation:

- LI-01.** The system is not the best and most optimal solution.
- LI-02.** The mobile application is for viewing information mainly.
- LI-03.** The mobile application only supports Android OS.

### Exclusions:

- EX-01.** The system can only provide available services from the building.
- EX-02.** The system does not manage the building's services.

## II. Project Management Plan

### 1. Overview

#### 1.1 Scope & Estimation

#	WBS Item	Complexity	Est. Effort (man-days)
<b>1</b>	<b>Project Initiating</b>		
1.1	Requirement Analysis	Medium	2
1.2	Problem Analysis	Complex	3
1.3	Determine Project Scope	Complex	2
1.4	Listing Requirements	Medium	1
<b>2</b>	<b>Project Planning</b>		
2.1	Featural List	Medium	1
2.2	User Stories	Medium	2
2.3	Use Case Diagram	Medium	2
2.4	Conceptual Diagram	Medium	1
2.5	ERD	Medium	1
2.6	Physical Diagram	Medium	1
2.7	Team meeting	Simple	1
2.8	Supervisor meeting	Medium	1
2.9	Create Source Base	Complex	6
<b>3</b>	<b>Project Executing</b>		
<b>3.1</b>	<b>Iteration 1</b>		
3.1.1	Weekly Meeting	Simple	1
3.1.2	Requirement & Design	Medium	2
3.1.3	Code & Implementation	Complex	4
	<b>Back-end</b>		
3.1.3.1	Configuration and set up environment	Complex	1
3.1.3.2	Create entities and DTO, Mapping DTOs	Simple	1.5
3.1.3.3	Repository, Service layers	Medium	3
3.1.3.4	Authorization service	Simple	1
3.1.3.5	Validation	Complex	3

3.1.3.6	Set up mail service	Medium	0.5
3.1.3.7	Create CRUD function for selected entities	Medium	5
3.1.3.8	Middleware layer (try catch exception, error)	Medium	1
3.1.3.9	Configure Azure (Azure storage, Azure SQL)	Medium	1
3.1.3.10	Configure Redis cache service	Medium	1
3.1.3.11	Review Code with Supervisor	Medium	0.5
3.1.3.12	Fix Review Code	Medium	3
3.1.3.13	Code review before merging branches	Medium	0.5
	<b>Front-end</b>		
3.1.3.10	Configuration and set up environment	Medium	2
3.1.3.11	Home Screen	Medium	1.5
3.1.3.12	Code & Implementation: Home Screen	Medium	1.5
3.1.3.13	Login, sign in & forgot password Screen	Simple	1
3.1.3.14	Flat Screens (CRU)	Medium	2
3.1.3.15	Flat Type Screen (CRUD)	Medium	2
3.1.3.16	Room Screen (CRU)	Medium	2
3.1.3.17	Room Type Screen (CRUD)	Medium	2
3.1.3.18	Contract Screens (CRU)	Medium	2
3.1.3.19	Ticket Screen (RU)	Medium	2
3.1.3.20	Renter Screen (RU)	Simple	1
3.1.3.21	Invoice screen (CRU)	Medium	2
3.1.3.22	Statistic screen (R)	Simple	1
3.1.3.23	Front end deployment setting	Medium	1
	<b>Mobile</b>		
3.1.3.24	Configuration and set up environment	Medium	1
3.1.3.25	Login, sign in & forgot password Screen	Simple	0.5
3.1.3.25	Renter Home Screen	Simple	0.5
3.1.3.26	Renter Profile Screen (RU)	Medium	0.5
3.1.3.27	Renter Ticket Screen (CRU)	Medium	2
3.1.3.28	Renter Contract Screen (R)	Simple	0.5



3.1.3.29	Renter Invoice Screen (R)	Simple	0.5
3.1.3.30	Renter Rental Screen	Medium	1
<b>3.1.4</b>	<b>Test</b>		
3.1.4.1	Create test case Iteration 1	Medium	4
3.1.4.2	Test Iteration 1	Medium	2
3.1.4.3	Verify Test Iteration 1	Medium	2
3.1.4.4	Fix bug Iteration 1	Medium	3
<b>3.1.5</b>	<b>Set up</b>		
3.1.5.1	Setup Env deploy (CI-CD)	Complex	3
3.1.5.2	Setup domain	Simple	1
3.1.5.3	Move source code to CI-CD (check settings)	Medium	2
<b>3.1.6</b>	<b>Summary &amp; Evaluation Iteration 1</b>	Medium	2
<b>3.2</b>	<b>Iteration 2</b>		
3.2.1	Weekly Meeting	Simple	1
3.2.2	Requirement & Design	Medium	3
3.2.3	Code & Implementation	Complex	4
	<b>Back-end</b>		
3.2.3.1	Project Service API	Medium	2
3.2.3.2	Redis cache issues	Medium	1
3.2.3.3	Deployment issue with Azure	Medium	1
	<b>Front-end</b>		
3.2.3.4	Connect Project Service API	Simple	2
3.2.3.5	Fix bug	Medium	2
3.2.3.6	Deployment issue with Digital Ocean WebApp	Medium	2
	<b>Mobile</b>		
3.2.3.7	Technician Home Screen	Simple	0.5
3.2.3.8	Technician Profile Screens (RU)	Medium	1
3.2.3.9	Technician Ticket Screen (CRU)	Medium	1.5
3.2.3.9	Build APK	Simple	0.5
<b>3.2.4</b>	<b>Deploy Setting</b>	Medium	2

<b>3.2.5</b>	<b>Test</b>		
3.2.5.1	Summary & Evaluation Iteration 2	Medium	2
3.2.5.2	Finish Project Executing	Medium	2
<b>4</b>	<b>Project Finishing</b>		
4.1	Collect & prepare Documents for project closure		5
4.2	Supervisor weekly meeting 16		1
4.3	Team weekly meeting 16		1
4.4	Close project		1
<b>Total Estimated Effort (man-days)</b>			<b>137</b>

Table 4- Scope & Estimation

## 1.2 Project Objectives

Timeliness: 90%

Allocated Effort: 2 (members) \* 4 (months) \* 30 (days/month) = 116 man-days

## 1.3 Project Risks

#	Risk Description	Impact	Possibility	Response Plans
1	Technology risk: cannot find the suitable solution for technique problems, the software cannot fulfill the expectation.	High	Medium	Do more research on official resources, discuss with mentor
2	Technology risk: host server dies	Medium	Low	Change host server
3	Requirement changes: requirements unexpected changes.	Critical	Medium	Chose another way of solution to respond to those changes.
4	Time risk: Cannot finish on time because of human issue	Critical	Low	Reduce scope, make assumptions, optimize schedule

Table 5- Project Risks

## 2. Management Approach

### 2.1 Project Process

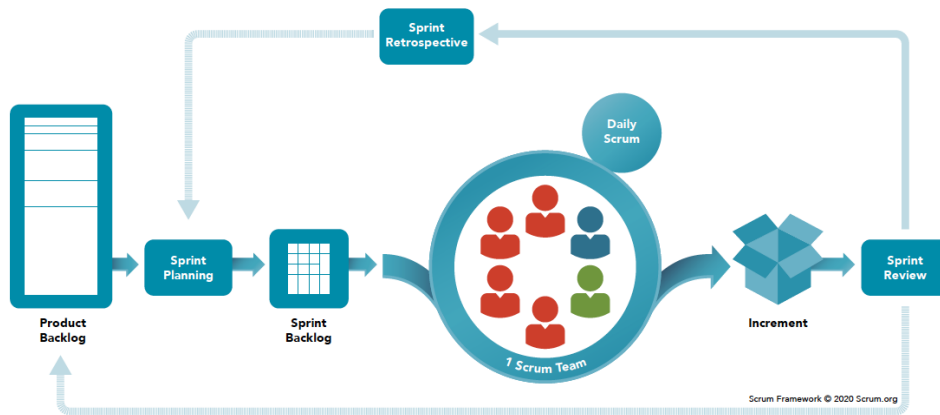


Figure 1 - Project Process

This project was developed using the Scrum model, which is part of the Agile framework for project development, for the following reasons:

- The planning and documentation phases in advance allow large or changing teams to stay informed and work towards a common goal.
- The force is structured and disciplined.
- Is simple to understand, follow and organize the tasks.
- Allows for easy implementation of early specification or design changes.
- Clearly define milestones and deadlines.
- Face to face meeting, online meeting.
- Work together to define and unify solutions.

### 2.2 Quality Management

To increase the project quality and user experience, the team apply the:

- Do a survey to understand what users want and their need.
- Acceptance testing:
  - Do a survey to understand how users receive the application and collect feedback.
  - Based on the users' feedback, the team fix bug and update.
- Hold a daily meeting to track project progress and support each other.

### 2.3 Training Plan

Training Area	Participants	When, Duration	Waiver Criteria
ReactJS, Flutter	Everyone	1-2 weeks	Mandatory
Asp .NET 6 Web API	Everyone	1-2 weeks	Mandatory
SQL Server	Everyone	1-2 weeks	Mandatory
Git desktop, GitHub	Everyone	1-2 weeks	Mandatory

Table 6 - Training Plan

### 3. Project Deliverables

#	Deliverable	Due Date	Deliverable Scope
1	Project Plan document	23/01/2023	Overall project plan
2	SRS documents	15/02/2023	Software requirements
3	Design documents	22/03/2023	Architecture Design, Detailed design, Database, UI design
4	Software Testing Document	12/05/2023	Testing Reports
5	User Guides documents	22/04/2023	Installation guides, User manual guides
6	Web and mobile application	25/04/2023	Code, System test cases
7	Code Front-end and mobile package	25/05/2023	Code, System test cases
8	Code Back-end package	25/05/2023	Code, System test cases
9	Final project package	25/05/2023	Final codes & documents

Table 7- Project Deliverables

#### 4. Responsibility Assignments

Full name	Roles	Responsibility
Lâm Hữu Khánh Phương Nguyễn Thế Hoàng	Supervisor	<ul style="list-style-type: none"> <li>• Instruct project team</li> <li>• Supervise project status</li> <li>• Review deliverables</li> <li>• Answer questions about the project</li> </ul>
Đào Bảo Trâm	Leader	<ul style="list-style-type: none"> <li>• Plan projects/tasks</li> <li>• Determine impediments and provide solutions to complete tasks</li> <li>• Track member's progress</li> <li>• Resolve conflicts impact on the team</li> </ul>
Đào Bảo Trâm Phạm Minh Đức Trần Khải Minh Khôi (DROP) Dương Đình Nguyên (DROP)	Team members	<ul style="list-style-type: none"> <li>• Gather and analysed requirements</li> <li>• Design UI</li> <li>• Implement code for the web application</li> <li>• Test the web application</li> <li>• Implement API</li> <li>• Test API</li> <li>• Review code</li> <li>• Contribute to improving project quality</li> <li>• Write documents</li> </ul>

Table 8 - Responsibility Assignments

#### 5. Project Communications

Communication Item	Target	Purpose	Frequency	Type, Tool, Methods
Daily meeting	Team member	Report task progress Identify obstacles	Always	Discord Google meet
Weekly Report	Supervisors Team members	Discuss project's status, solution Review diagram, document	Weekly	Zalo Google meet Offline meetings
Weekly meeting	Customer Team members	Get requirements	Weekly	Zalo Google meet Offline meetings

Table 9 - Project Communications

## 6. Configuration Management

### 6.1 Document Management

- Use OneDrive to save document and for real-time editing on Microsoft Word.
- Use OneDrive to save meeting report, meeting record, diagram, image, design, document to each category folder

### 6.2 Source Code Management

- GitHub server to store source code.
- Follow the rule of team when using Git:
  - Commit and note the changes details often.
  - Review carefully before push into develop branch.
  - Fetch, review and resolve conflicts when merging usually

### 6.3 Tools & Infrastructures

Category	Tools / Infrastructure
Technology	JavaScript (Front-end), C# (Back-end), Redis (Cache), Dart (Mobile)
Framework	.NET, ReactJS, Flutter
Database	SQL Server
IDEs/Editors	Visual Studio Code, Visual Studio, IntelliJ
Diagramming	Draw.io, DbForge
Documentation	MS Office, Microsoft 365
Version Control	GitHub (Source Codes), OneDrive (Documents)
Deployment server	Azure, VPS (Digital Ocean), Firebase
Project management	Jira

*Table 10 - Tools & Infrastructures*

### III. Software Requirements Specification

#### 1. Product Overview

VinFlat system manages the dormitory rental process, effectively manage renter information, and assist in managing and archiving contracts, invoices, and time schedules at different locations with optimal solutions. The system creates convenience, saves time, and minimizes errors compared to traditional management.

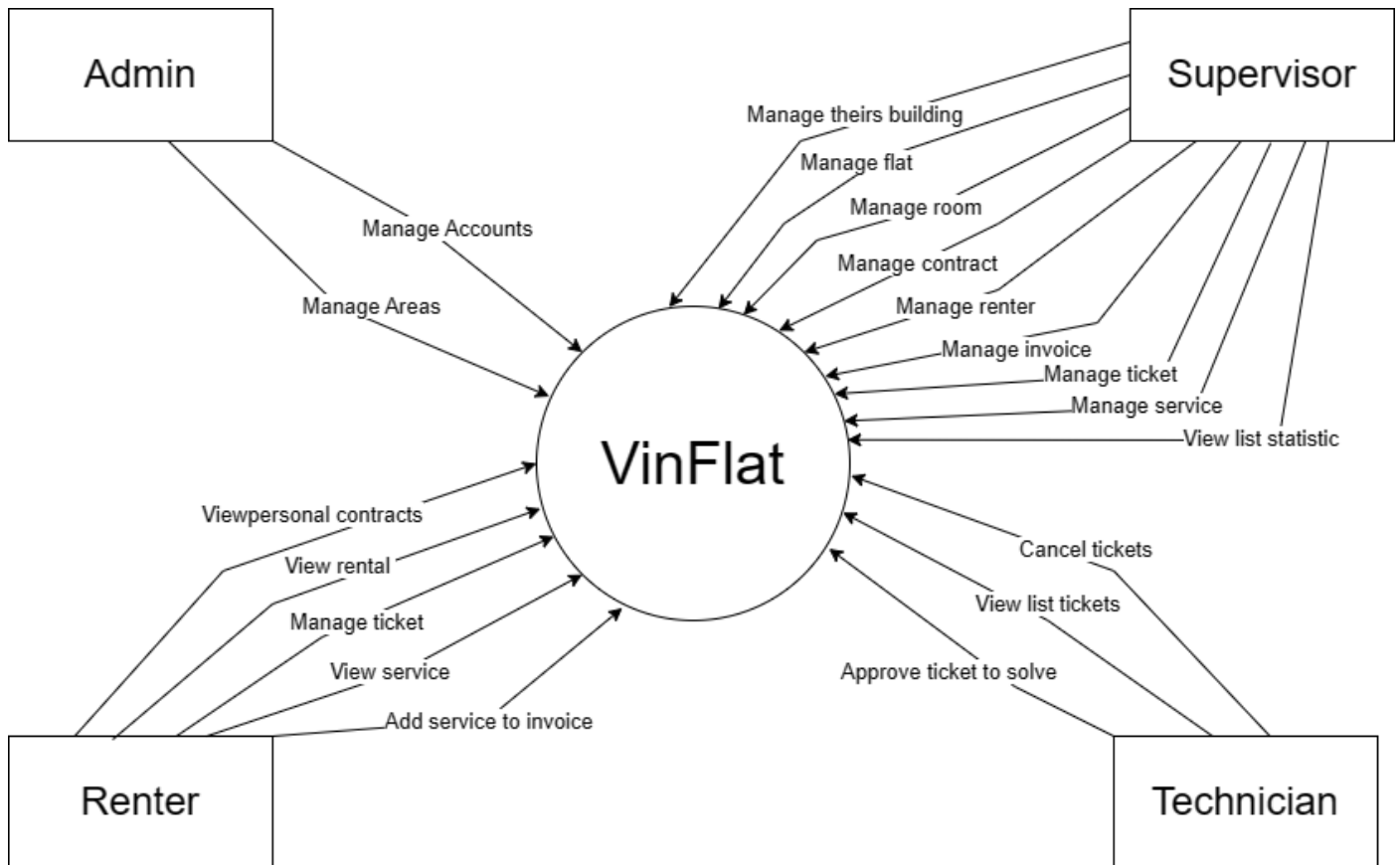


Figure 2 - Product Overview Diagram

## 2. User Requirements

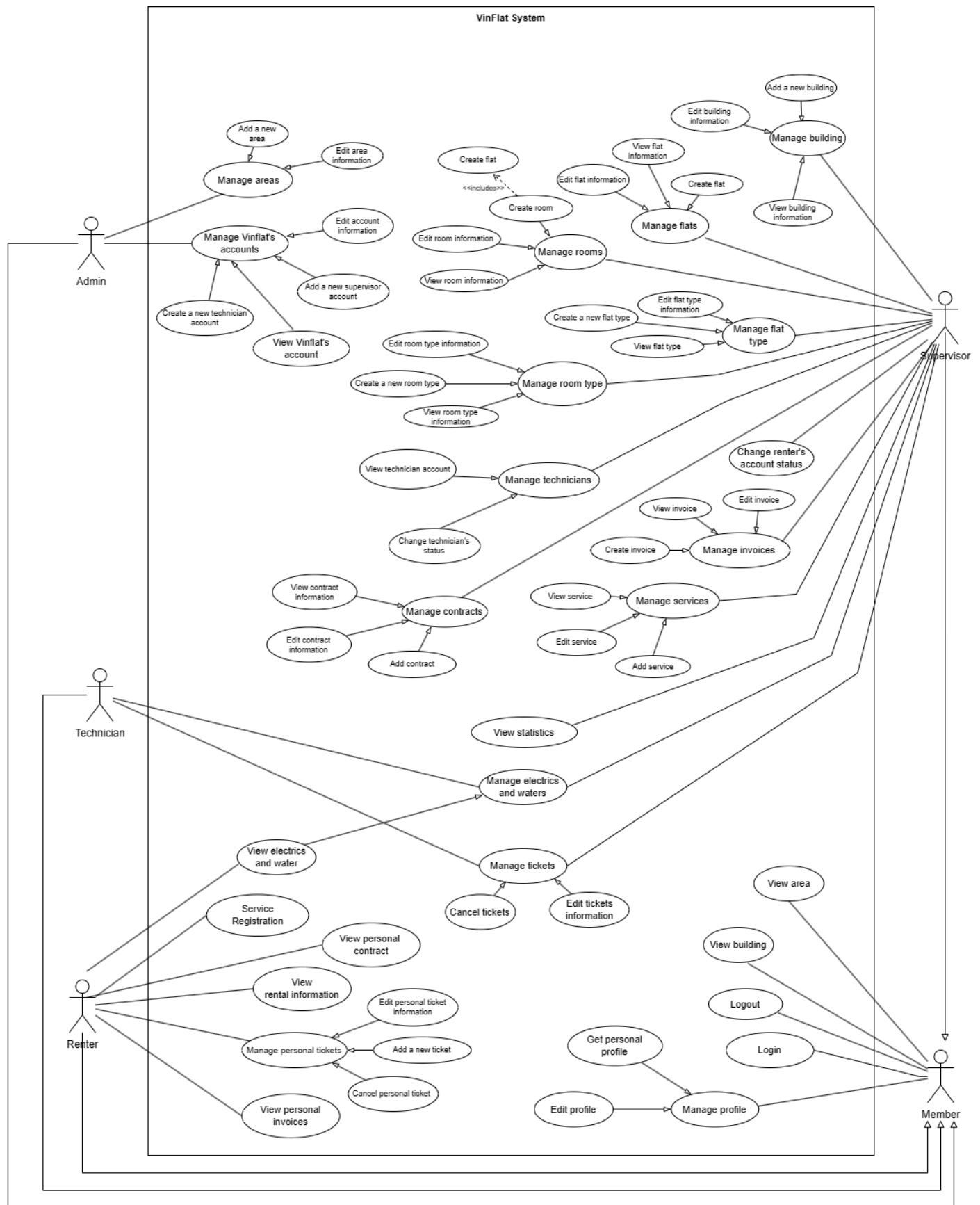
### 2.1 Actors

#	Actor	Description
1	Renter (tenant)	As a renter who uses the dormitory's services, keeps track of information related to own rental and payment information, orders services and reports problems, ...
2	Supervisor (Manager of VinFlat)	As a supervisor, has the right to manage Buildings, Flats and their properties, renters, handles financing task like managing renters' accounts, contracts and payments, monthly income and outcome and sends announcement to renters via mails and notifications
3	Administrator (Admin of VinFlat)	As a system administrator, has the right to do every task a manager can do plus account managing task like create new manager account, ban/unban manager account
4	Technician	As a technician, performs duties like recording water and electricity meter, resolving reported problem related to households and public appliances malfunction
5	Member	As a member, I have the right to view general information when not logged in, log in when needed, view and edit my personal profile, etc.

*Table 11 - Actors*



### 2.2.1 Diagram(s)



### 2.2.2 Use Case Description

ID	Use Case	Actors	Description
01	Login/Logout	Member	Unauthenticated User login to VinFlat to verify role, and give them access to more features in the system. Then allow all user can logout the system
02	View profile		This feature allows all members view their profile
03	Update profile		This feature allows members update their profile
04	Change password		This feature allows members change their password
05	Search Building		This feature allows member search building
06	Search Area		This feature allows member search building by area
07	View Building information		This feature allows member view building information
08	View Area information		This feature allows member view area information
09	View Building list		This feature allows member view building list
10	Create Area	Admin	This feature allows admin create a new area
11	Update Area information		This feature allows admin update area information
12	View Area list		This feature allows admin view list of all area
13	Create Supervisor account		This feature allows admin create supervisor account
14	View Supervisor Information		This feature allows admin view supervisor information
15	Search Supervisor		This feature allows admin search supervisor
16	Update Supervisor account status		This feature allows admin update supervisor account status
17	Create Technician account		This feature allows admin create technician account
18	View Technician information		This feature allows admin view technician information
19	Search Technician		This feature allows admin search technician in system
20	Update Technician account status	Admin Supervisor	This feature allows admin or supervisor update technician account status
21	Create Renter account	Supervisor	This feature allows supervisor create renter account
22	Search Renter		This feature allows supervisor search renter information
23	Update renter account status		This feature allows supervisor update renter account status
24	View Renter list		This feature allows supervisor view renter list
25	View Renter Information		This feature allows supervisor view renter information
26	Create Building		This feature allows supervisor create building
27	Update Building information		This feature allows supervisor update building information
28	Create Flat		This feature allows supervisor create flat
29	Update Flat information		This feature allows supervisor update flat information
30	View Flat list		This feature allows supervisor view flat list
31	View Flat information		This feature allows supervisor view flat information
32	Search Flat		This feature allows supervisor search flat
33	Create Flat Type		This feature allows supervisor create flat type
34	Update Flat Type information		This feature allows supervisor update flat type information
35	View Flat Type list		This feature allows supervisor view flat type list
36	View Flat Type information		This feature allows supervisor view flat type information
37	Search Flat Type		This feature allows supervisor search flat type
38	Create Room Type		This feature allows supervisor create room type
39	Update Room Type information		This feature allows supervisor update room type information
40	View Room Type list		This feature allows supervisor view room type list
41	View Room Type information		This feature allows supervisor view room type information
42	Search Room Type		This feature allows supervisor search room type

43	Update Room Type information		This feature allows supervisor update room type information
44	Add signed Contract		This feature allows supervisor add signed contract
45	View Contract list		This feature allows supervisor view contract list
46	View Contract information		This feature allows supervisor view contract information
47	Search Contract		This feature allows supervisor search contract
48	Update Contract information		This feature allows supervisor update contract information
49	Generate Invoice		This feature allows supervisor generate invoice
50	View Invoice list		This feature allows supervisor view invoice list
51	View Invoice information		This feature allows supervisor view invoice information
52	Search Invoice		This feature allows supervisor search invoice
53	Update Invoice information		This feature allows supervisor update invoice information
54	View Ticket list	Supervisor Technician	This feature allows supervisor and technician view ticket list
55	View Ticket information		This feature allows supervisor and technician view view ticket information
56	Update Ticket information		This feature allows supervisor and technician view update ticket information
57	Approve / Disapprove Ticket		This feature allows supervisor and technician view approve / disapprove ticket
58	Confirm solving ticket		This feature allows supervisor and technician view confirm solving ticket
59	Record water meter		This feature allows supervisor and technician view record water meter
60	Record electricity meter	Renter	This feature allows supervisor and technician view record electricity meter
61	View own contract		This feature allows renter view own contract
62	View own flat/room information		This feature allows renter view own flat/room information
63	View own Invoices information		This feature allows renter view own invoices information
64	View own rental information		This feature allows renter view own rental information
65	Make request		This feature allows renter make request
66	View available services in building		This feature allows renter view available services in building
67	Order service		This feature allows renter order service
68	Submit problem ticket		This feature allows renter submit problem ticket
69	View sent problem tickets		This feature allows renter view sent problem tickets

Note: Member included all logged in admins, supervisors, technicians, renters

Table 12 - Use Case Description

### 3. Functional Requirements

#### 3.1 System Functional Overview

##### 3.1.1 Screens Flow

###### a. Web admin

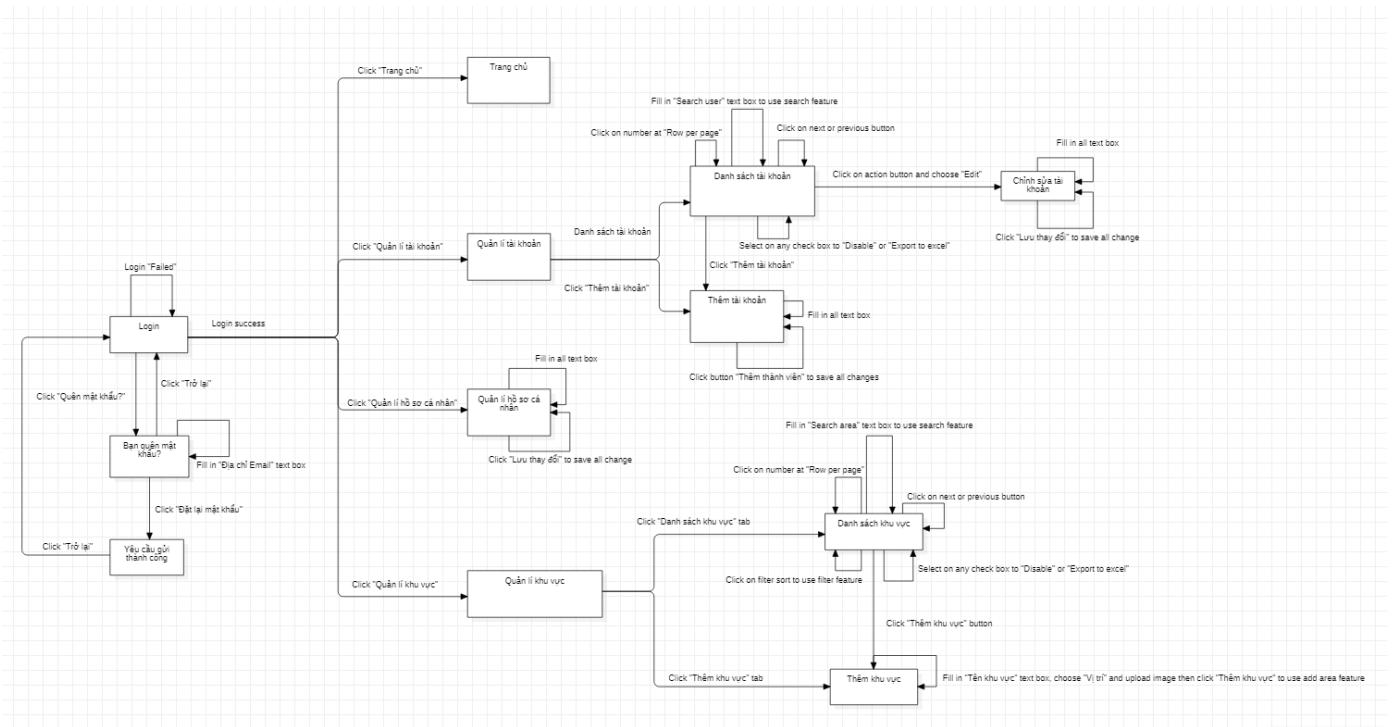


Figure 4 - Web admin screen flow

###### b. Web supervisor

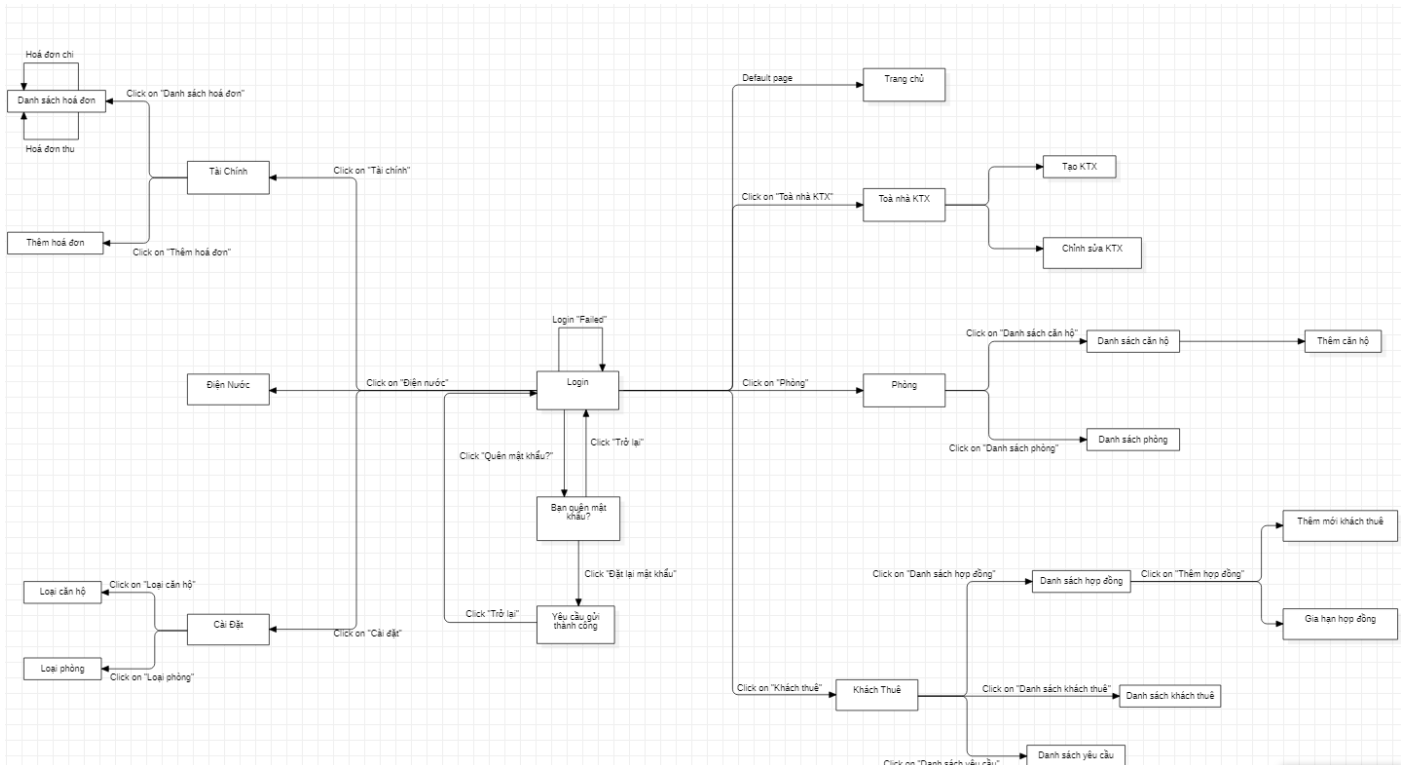


Figure 5 - Web supervisor screen flow

### c. Mobile technician

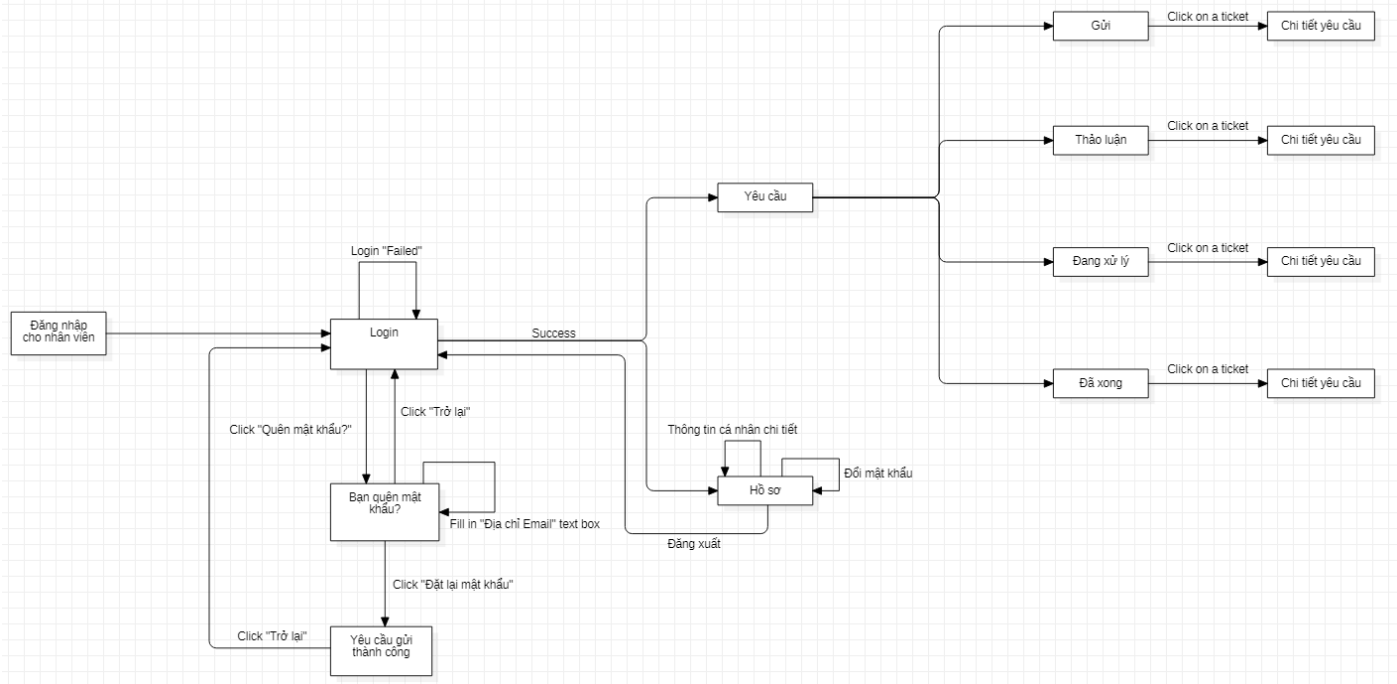


Figure 6 - Mobile technician screen flow

### d. Mobile renter

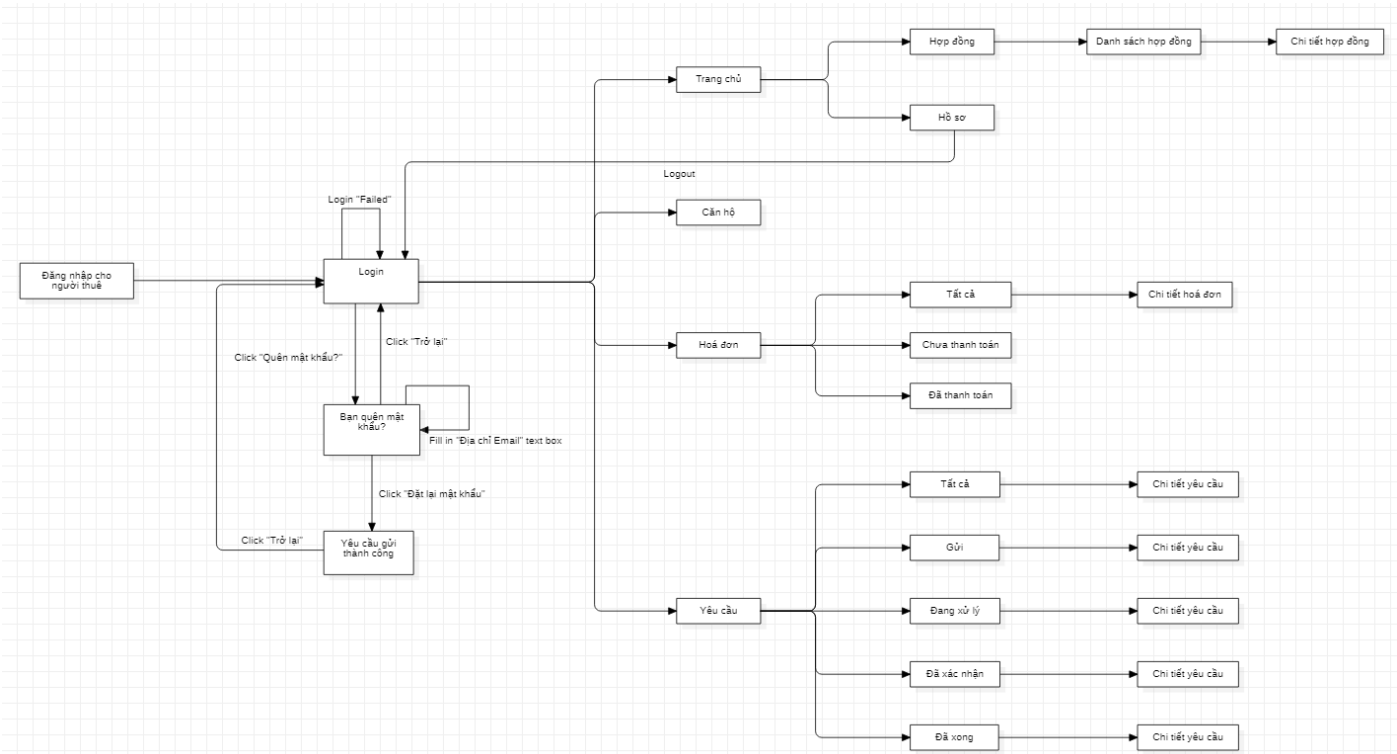


Figure 7 - Mobile renter screen flow

### 3.1.2 Functional requirements

#### 3.1.2.1 Admin functional requirements

No	Actor	Function Name	Purpose	Data requirements	Data validation	Business Rules
1	Admin	View VinFlat's account list	View account list	Employee name	FullName : + Type: Text Field (string) + Max length = 100	N/A
2	Admin	View account	View an account	Employee name	FullName : + Type: Text Field (string) + Max length = 100	N/A
3	Admin	Create Account	Create an account	Username FullName Email Password Phone Number Status Address RoleId	Username : + Type: Text Field (string) + Max Length = 100 FullName : + Type: Text Field (string) + Max length = 100 Email : + Type: Text Field (string) + Max Length = 100 Phone Number : + Type: Text Field (string) + Max Length = 14 Status : + Type : Toggle button (boolean) RoleId : + Type: Text Field (int)	BR-9 BR-10
4	Admin	Update Account	Update account status	Employee Id Status	EmployeeId + Type : Text Field (int) Status + Type : Toggle button (boolean)	BR-9 BR-10
5	Admin	Create Area	Create an area	Name	Name	BR-11

				Status	+ Type : Text Field (string) + Max Length = 100 Status + Type : Toggle button (boolean)	BR-12 BR-13
6	Admin	Update Area	Update an area	Area Id Name Status	Area Id : + Type : Text Field (int) Name + Type : Text Field (string) + Max Length = 100 Status + Type : Toggle button (boolean)	BR-11 BR-12 BR-13

Table 13 - Admin functional requirements

### 3.1.2.2 Supervisor functional requirements

No	Actor	Function Name	Purpose	Data requirements	Data validation	Business Rules
1	Supervisor	View personal building	View their building Information	Building name	Building name : + Type : Text Field (string) + Max length = 100	N/A
2	Supervisor	Create personal building	Create a Building	Building name Building address Description Building phone number Building image Status EmployeeId AreaId	Building name : + Type : Text Field (string) + Max Length = 100 Description : + Type : Text area (string) + Max Length = 500 Building phone number : + Type : Text Field (string) + Max Length = 14 Status : + Type : Toggle button (boolean) Employee Id : + Type : Text field (int)	BR-16 BR-13

					Area Id : + Type : Text field (int)	
3	Supervisor	Update personal building	Update their building	Building name Building address Description Building phone number Building image Status Areald	Building name : + Type : Text Field (string) + Max Length = 100 Description : + Type : Text Field (string) + Max Length = 500 Building phone number : + Type : Text Field (string) + Max Length = 14 Status : + Type : Toggle button (boolean) Area Id : + Type : Text field (string)	BR-16 BR-13
4	Supervisor	View flat	View flat information	Name Description	Name + Type : Text field (string) + Max length = 100 Description : + Type : Text Field (string) + Max Length = 500	N/A
5	Supervisor	Create flat	Create a flat	Name Description Status WaterMeterBefore WaterMeterAfter ElectricityMeterBefore ElectricityMeterAfter ImageUrl FlatTypeId	Name + Type : Text field (string) + Max length = 100 Description : + Type : Text area (string) + Max Length = 500 Status : + Type : Text field (string) + Max length = 20	BR-24 BR-25



				BuildingId	WaterMeterBefore + Type : Text field (decimal) WaterMeterAfter + Type : Text field (decimal) ElectricityMeterBefore + Type : Text field (decimal) ElectricityMeterAfter + Type : Text field (decimal) ImageUrl + Type : File (byte) FlatTypeId : + Type : Text field (int) BuildingId : + Type : Text field (int)	
6	Supervisor	Update flat	Update flat information	Flat Name Description Status WaterMeterBefore WaterMeterAfter ElectricityMeterBefore ElectricityMeterAfter ImageUrl FlatTypeId	Id Flat Id : + Type : Text field (int) Name + Type : Text field (string) + Max length = 100 Description : + Type : Text Field (string) + Max Length = 500 Status : + Type : Text field (string) + Max length = 20 WaterMeterBefore + Type : Text field (decimal) WaterMeterAfter + Type : Text field (decimal) ElectricityMeterBefore + Type : Text field (decimal) ElectricityMeterAfter	BR-24

					+ Type : Text field (decimal) ImageUrl + Type : File (byte) FlatTypeid : + Type : Text field (int)	
7	Supervisor	View room	View room information	Room Name	Room Name : + Type : Text field (string) + Max length : 100	N/A
8	Supervisor	View room list	View list of room information in managed building	Room Name Room Type id	Room Name + Type : Text field (string) + Max length : 100 Room Type id + Type : Text field (int) +	N/A
9	Supervisor	Create room	Create a room	Room Name Status Electricity Attribute Water Attribute Building Id Room Type Id Flat ImageUrl Id	Room Name : + Type : Text field (string) + Max length : 100 Status : + Type : Text field (string) + Max length : 15 Water Attribute : + Type : Text field (decimal) Electricity Attribute : + Type : Text field (decimal) Building Id : + Type : Text field (int) Flat Id : + Type : Text field (int) ImageUrl + Type : File (byte)	BR-31 BR-29
10	Supervisor	Update room	Update room information	Room Name Status	Room Name : + Type : Text field (string)	BR-31 BR-26

				Electricity Attribute Water Attribute Room Type Id ImageUrl	+ Max length : 100 Status : + Type : Text field (string) + Max length : 15 Water Attribute : + Type : Text field (decimal) Electricity Attribute : + Type : Text field (decimal) ImageUrl + Type : File (byte)	
11	Supervisor	View flat type	View flat type information	Flat Type Name	Flat Type Name + Type : Text field (string) + Max length : 100	N/A
12	Supervisor	View list flat type	View list of flat type information in managed building	Flat Type Name	Flat Type Name + Type : Text field (string) + Max length : 100	N/A
13	Supervisor	Create flat type	Create a flat type	Flat Type Name Room Capacity Status Building Id	Flat Type Name + Type : Text field (string) + Max length : 100 Room Capacity + Type : Text field (int) Status + Type : Toggle button (boolean) Building Id : + Type : Text field (int)	BR-21 BR-22 BR-23
14	Supervisor	Update flat type	Update flat type information	Flat Type Id Flat Type name Status Room Capacity	Flat Type Name + Type : Text field (string) + Max length : 100 Room Capacity + Type : Text field (int) Status	BR-21 BR-22 BR-23

					+ Type : Toggle button (boolean) Flat type Id : + Type : Text field (int)	
15	Supervisor	View room type	View room type information	Flat Type Name Status Room Capacity	Flat Type Name + Type : Text field (string) + Max length : 100 Room Capacity + Type : Text field (int) Status + Type : Toggle button (boolean) Flat type Id :	N/A
16	Supervisor	View room type list	View list of room type information in managed building	Room Type Name Total slots Status	Room Type Name + Type : Text field (string) + Max length : 100 Total slots + Type : Text field (int) Status + Type : Toggle button (boolean)	N/A
14	Supervisor	Create room type	Create a room type	Room Type Name Total Slots Status Electricity Attribute Water Attribute	Room Type Name + Type : Text field (string) + Max length : 100 Total slots + Type : Text field (int) Status + Type : Toggle button (boolean) Electricity Attribute + Type : Text field (decimal) Water Attribute + Type : Text field (decimal)	BR-27 BR-29
15	Supervisor	Update room type	Update room type information	Room Type Id Room Type Name Total Slots	Room Type Id + Type : Text field (int) Room Type name	BR-27 BR-29 BR-30

				Status Electricity Attribute Water Attribute	+ Type : Text field (string) + Max length : 100 Total Slots + Type : Text field (int) Electricity Attribute + Type : Text field (decimal) Water Attribute + Type : Text field (decimal)	
16	Supervisor	View contract list	View list of contract information in managed building	Contract serial number Contract name Start Date End Date	Contract serial number + Type : Text field (string) Contract name + Type : Text field (string) + Max length : 100 Start Date + Type : Text field (DateTime) End Date + Type : Text field (DateTime)	N/A
17	Supervisor	Create contract	Add a contract	Contract serial number Contract name Description Date signed Start date Create date End date Cancelled date Last updated Contract Status ImageUrl Price for rent Price for service Price for water Price for electricity	Contract serial number + Type : Text field (string) + Max length : 100 Contract name + Type : Text field (string) + Max length : 100 Description + Type : Text area(string) + Max length : 500 Date signed + Type : Text field (DateTime) Start date + Type : Text field (DateTime) Create date + Type : Text field (DateTime)	BR-59 BR-61 BR-77 BR-76 BR-20 BR-17 BR-15

				Building id Room id Renter id Flat id	End date + Type : Text field (DateTime) Cancelled date + Type : Text field (DateTime) Last updated + Type : Text field (DateTime) Contract Status + Type : Text field (string) ImageUrl + Type : Text field (byte[]) Price for rent + Type : Text field (decimal) Price for service + Type : Text field (decimal) Price for water + Type : Text field (decimal) Price for electricity + Type : Text field (decimal) Building id + Type : Text field (int) Room id + Type : Text field (int) Renter id + Type : Text field (int) Flat id + Type : Text field (int)	
18	Supervisor	Update contract	Update contract information	Contract Id Contract name Description Date signed Start date End date	Contract name + Type : Text field (string) + Max length : 100 Description + Type : Text area(string) + Max length : 500	BR-59 BR-61 BR-77 BR-76 BR-20 BR-17

				Cancelled date Last updated Contract Status ImageUrl Price for rent Price for service Price for water Price for electricity	Date signed + Type : Text field (DateTime) Start date + Type : Text field (DateTime) Create date + Type : Text field (DateTime) End date + Type : Text field (DateTime) Cancelled date + Type : Text field (DateTime) Last updated + Type : Text field (DateTime) Contract Status + Type : Text field (string) ImageUrl + Type : Text field (byte[]) Price for rent + Type : Text field (decimal) Price for service + Type : Text field (decimal) Price for water + Type : Text field (decimal) Price for electricity + Type : Text field (decimal)	BR-15
19	Supervisor	View a contract	View a contract in managed building	Contract : Contract table		N/A
20	Supervisor	View a renter	View a renter in managed building	Renter : Renter table		N/A
21	Supervisor	View list renter	View list of renter in managed building	FullName Phone number Status	Fullname + Type : text field (string) Phone number : + Type : Text field (string)	N/A

					Status : + Type : Toggle button (boolean)	
22	Supervisor	Create renter	Create a renter account	Username Email Password Phone number Fullname Birthdate Status ImageUrl CitizenNumber CitizenCardFrontImageUrl CitizenCardBackImageUrl Address Gender	Username + Type : Text field (string) + Max length : 100 Email + Type : Text field (string) + Max length : 100 Password + Type : Password (string) + Max length : 100 Phone number : + Type : Text field (string) + Max length : 14 Fullname : + Type : Text field (string) + Max length : 100 Birthdate : + Type : Text field (DateTime) Status : + Type : Toggle button (boolean) Citizen number : + Type : Text field (string) CitizenCardFrontImageUrl : + Type : Text field (byte[]) CitizenCardBackImageUrl : + Type : Text field (byte[]) Address : + Type : Address (string) Gender : + Type : Gender (string)	BR-81 BR-82 BR-83 BR-84
23	Supervisor	Update renter account	Update renter	Renter Id	Renter Id :	BR-81



			account status	Status	+ Type : Text field (int) Status : + Type : Toggle button (boolean)	BR-82 BR-83 BR-84 BR-85
24	Supervisor	View invoice	View invoice information	Invoice : Invoice table and invoice detail table		N/A
25	Supervisor	Create invoice	Create an invoice	Name Total amount Status Created Time Due Date Payment Time Detail Contract Id Renter Id Employee Id Building Id Invoice Type Id	Name : + Type : Text field (string) + Max length : 100 Total Amount + Type : Text field (decimal) Status : + Type : Text field (string) Created Time : + Type : Text field (DateTime) Due Date : + Type : Text field (DateTime) Payment Time : + Type : Text field (DateTime) Detail : + Type : Text field (string) Contract Id : + Type : Text field (int) Renter id : + Type : Text field (int) Employee id : + Type : Text field (int) Building Id : + Type : Text field (int) Invoice Type Id : + Type : Text field (int)	BR-37 BR-38 BR-39 BR-44 BR-49 BR-70 BR-86 BR-87
26	Supervisor	View list of invoices	View list of	Invoice name	Invoice name	N/A

			invoices in managed building	Name Total Amount Status Due Date Detail	Name Total Amount Status Due Date Detail	
27	Supervisor	Update invoice	Update invoice information	Invoice Id Name Total amount Status Due Date Payment Time Detail	Invoice Id Name Total amount Status Due Date Payment Time Detail	BR-86 BR-87
28	Supervisor	View ticket	View ticket information	Ticket : Ticket table		N/A
29	Supervisor	Update ticket	Update ticket status	Ticket Id Ticket status	Ticket Id Ticket status	BR-46 BR-45 BR-48 BR-51
30	Supervisor	View service	View service information	Service : service table		N/A
31	Supervisor	Create service	Create a service	Name Description Status Price ImageUrl Building Id Service Type Id	Name + Type : Text field (string) + Max length : 100 Description + Type : Text area (string) + Max length : 500 Status + Type : Toggle button (boolean) Price + Type : Text field (decimal)	BR-33 BR-34 BR-36

					ImageUrl + Type : byte[] Building Id + Type : Text field (int) Service Type Id + Type : Text field (int)	
32	Supervisor	Update service	Update service information	Service Id Name Description Status Price ImageUrl Service Type Id	Service Id + Type : Text field (int) Name + Type : Text field (string) + Max length : 100 Description + Type : Text area (string) + Max length : 500 Status + Type : Toggle button (boolean) Price + Type : Text field (decimal) ImageUrl + Type : byte[] Service Type Id + Type : Text field (int)	BR-33 BR-34 BR-36
33	Supervisor	View statistic	View list statistic information			N/A

Table 14 - Supervisor functional requirements

### 3.1.2.3 Technician functional requirements

No	Actor	Function Name	Purpose	Data requirements	Data validation	Business Rules
1	Technician	Approve Ticket	Approve a ticket to resolve	Ticket Id Status	Ticket Id + Type : Text field (int) Status	N/A

					+ Type : Text field (string)	
2	Technician	Resolve Ticket	Resolve a ticket	Ticket Id Status	Ticket Id + Type : Text field (int) Status + Type : Text field (string)	N/A
3	Technician	Cancel Ticket	Cancel a ticket	Ticket Id Status	Ticket Id + Type : Text field (int) Status + Type : Text field (string)	N/A
4	Technician	View ticket list	List of ticket information in managed building	Ticket Name Ticket Type id Status	Ticket Name + Type : Text field (string) + Max length : 100 Ticket Type id + Type : Text field (int) Status + Type : Text field (string)	N/A

Table 15 - Technician functional requirements

#### 3.1.2.4 Renter functional requirements

No	Actor	Function Name	Purpose	Data requirements	Data validation	Business Rules
1	Renter	View rental information	View own rental	Renter Id Flat Name Building Name	Renter Id + Type : Text field (int) Flat Name + Type : Text field (string) + Max length : 100 Building Name + Type : Text field (string) + Max length : 100	N/A
2	Renter	View service provided by building	View services provided by	Renter Id List of Service Name	Renter Id + Type : Text field (int)	N/A

			building		List of Service Name + Type : List	
3	Renter	View personal invoice list	View list of personal invoices	Invoice name Detail Total Due Date Amount	Invoice name + Type : Text field (string) + Max length : 100 Detail + Type : Text area (string) + Max length : 500 Total Amount + Type : Text field (decimal) Due Date + Type : Text field (DateTime)	N/A
4	Renter	View personal ticket list	View list of personal tickets	Ticket Name Ticket type id Description Status	Ticket Name + Type : Text field (string) + Max length : 100 Ticket type id Description Status + Type : Text field (string) + Max length : 100	N/A
5	Renter	View personal contract list	View list of personal contracts	Contract serial number Contract name Description Contract status Date signed Start date End date Last updated Contract Status	Contract serial number + Type : Text field (string) + Max length : 100 Contract name + Type : Text field (string) + Max length : 100 Description + Type : Text area(string) + Max length : 500 Contract status + Type : Text field (string) + Max length : 100	N/A

					Date signed + Type : Text field (DateTime) Start date + Type : Text field (DateTime) End date + Type : Text field (DateTime) Last updated + Type : Text field (DateTime) Contract Status + Type : Text field (string) + Max length : 100	
6	Renter	View personal ticket detail	View a personal ticket	Ticket Id Ticket name Ticket type id	Ticket Id Ticket name + Type : Text field (string) + Max length : 100 Ticket type id + Type : Text field (int)	N/A
7	Renter	View personal invoice detail	View a personal invoice	Invoice Id Amount Due Date Created Date Payment Date	Invoice Id + Type : Text field (int) Amount + Type : Text field (decimal) Due Date + Type : Text field (DateTime) Created Date + Type : Text field (DateTime) Payment Date + Type : Text field (DateTime)	N/A
8	Renter	View personal contract	View a personal contract	Contract Id Contract serial number Contract Name Contract serial number Contract name	Contract Id + Type : Text field (int) Contract serial number + Type : Text field (string) + Max length : 100	N/A

				Description Contract status Date signed Start date End date Last updated Contract Status Price for service Price for rental Price for water Price for electricity	Contract Name + Type : Text field (string) + Max length : 100 Description + Type : Text area(string) + Max length : 500 Contract status + Type : Text field (string) + Max length : 100 Date signed Start date End date Last updated Contract Status + Type : Text field (string) + Max length : 100 Price for service + Type : Text field (decimal) Price for rental + Type : Text field (decimal) Price for water + Type : Text field (decimal) Price for electricity + Type : Text field (decimal)	
9	Renter	Place a service from building	Place a service	Service Id Service name	Service Id + Type : Text field (int) Service name + Type : Text field (string) + Max length : 100	N/A
10	Renter	Change profile picture	Change profile picture	Renter id ImageUrl	Renter id + Type : Text field (int) ImageUrl	N/A

					+ Type : Byte[]	
11	Renter	View rental's flat information	View rental information of logged in renter	Building Name Flat name Services Room mates Renter Name	Building Name + Type : Text field (string) + Max length : 100 Flat name + Type : Text field (string) + Max length : 100 Services + Type : Dropdown list Room mates + Type : list Renter Name + Type : Text field (string)	N/A
12	Renter	Create ticket	Create a ticket	Ticket Name Description Ticket Type	Ticket Name + Type : Text field (string) + Max length : 100 Description + Type : Text area (string) + Max length : 500 Ticket Type + Type : Text field (int)	BR-43 BR-78
13	Renter	Edit personal ticket	Edit a ticket	Ticket Id Ticket Name Description Ticket Type	Ticket Id + Type : Text field (int) Ticket Name + Type : Text field (string) + Max length : 100 Description + Type : Text area (string) + Max length : 500 Ticket Type + Type : Text field (int)	N/A
14	Renter	Cancel ticket when	Cancel a ticket	Ticket Id	Ticket Id	BR-89



		hasn't received by management		Status	+ Type : Text field (int) Status + Type : Text field (string)	
--	--	-------------------------------	--	--------	---	--

Table 16 - Renter functional requirements

### 3.1.2.5 Member functional requirements

No	Actor	Function Name	Purpose	Data requirements	Data validation	Business Rules
1	Member	Login	Login to website/ app mobile	Username/Email/Phone number Password	Username: + Type : Text Field (string) + Max length = 100 Email : + Type : Text Field (string) +Max length = 100 Phone number : + Type : Text Field (string) + Max length = 100	N/A
2	Member	View profile	View profile	Account Id Full Name	Account id : + Type : Text field (int) Full name : + Type : Text field (string)	N/A
3	Member	Update profile	Update new data for profile	Account Id Full Name Address Email Phone Number	Account id : + Type : Text field (int) Full name : + Type : Text field (string) + Max length : 100 Address : + Type : Text field (string) + Max length : 100 Email : + Type : Text field (string) + Max length : 100 Phone number :	N/A

					+ Type : Text field (string) + Max length : 14	
4	Member	Change password	Change password.	Old password New password Confirm password	Old password + Type : Password (string) + Max length : 100 New password + Type : Password (string) + Max length : 100 Confirm password + Type : Password (string) + Max length : 100	BR-88
5	Member	Forgot password	Forgot password	Email	Email + Type : Text field (string) + Max length : 100	N/A
6	Member	View building information	View list of building information in managed building	Building Name Description Average Price Status	Building Name + Type : Text field (string) + Max length : 100 Description + Type : Text area(string) + Max length : 500 Average Price + Type : Text string (decimal) Status + Type : Toggle button (boolean)	N/A
7	Member	View building list	View a building	Building Name Address Description	Building Name + Type : Text field (string) + Max length : 100 Address + Type : Text field (string) + Max length : 100 Description	N/A

					+ Type : Text area(string) + Max length : 500	
8	Member	View area list	View list of Area information	Area name Address	Area Name + Type : Text field (string) + Max length : 100 Address + Type : Text field (string) + Max length : 100	N/A
9	Member	View area information	View a area	Area Name Address	Area Name + Type : Text field (string) + Max length : 100 Address + Type : Text field (string) + Max length : 100	N/A

*Table 17 - Member functional requirements*

### 3.1.3 Screen Authorization

#### 3.1.3.1 Web Application Screen Authorization

Screen	Admin	Supervisor	Member
Login	x	x	
Dashboard	x	x	
Landing	x	x	x
Account	x	x	
Profile	x	x	x
Area	x		
Building		x	
Flat		x	
Room		x	
Contract		x	
Ticket		x	
Invoice		x	

Table 18 18 - Web Application Screen Authorization

#### 3.1.3.1 Mobile Application Screen Authorization

Screen	Renter	Technician	Member
Login	x	x	
Dashboard	x		
Profile	x	x	x
Ticket	x	x	
Rental	x		
Contract	x		
Invoice	x		

Table 19 - Mobile Application Screen Authorization

#### 3.1.4 Non-Screen Functions

#	System function	Description
1	Push notification	System push notification to customers and translator

Table 20 - Non-Screen Functions

## 3.2 Entity Relationship Diagram

### Conceptual ERD

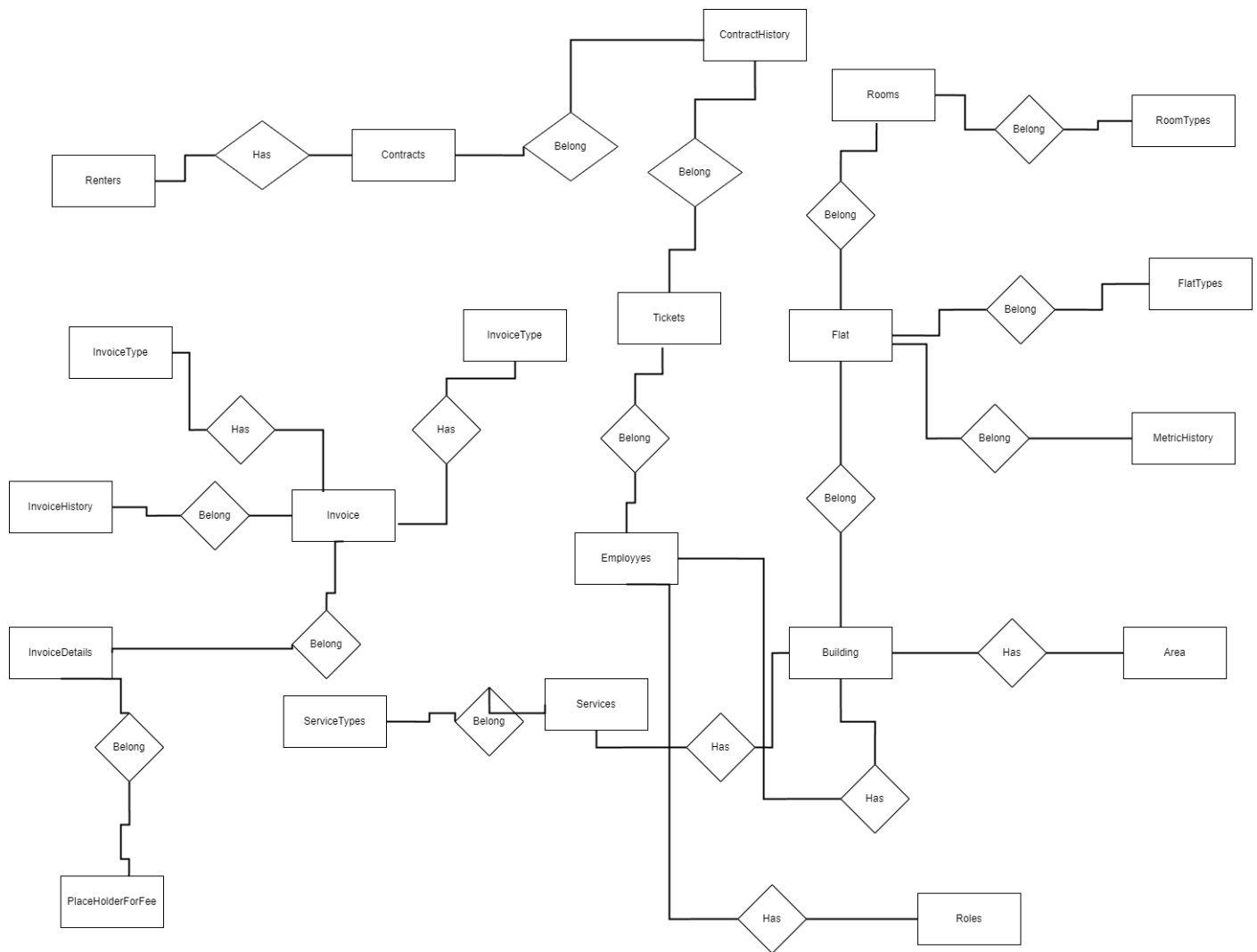


Table 21 - Entity Relationship Diagram

## 4. Non-Functional Requirements

### 4.1 External Interfaces

#### 4.1.1 User Interface

UI-1: The language used in the application is Vietnamese.

#### 4.1.2 Software interface

SI-1: File Storage Service

SI-1.1: Web app that integrates with the Hosting Service and uses its File Upload API.

SI-2: Authentication Service

SI-2.1: Web app can integrate with JSON Web Token (JWT) for user authentication.

## 4.2 Quality Attributes

### 4.2.1 Usability

- The application should be friendly and easy for users to use after training.
- Administrators and Managers can use the web with less than a day's instruction.
- Renters can use the mobile applications with less than 2 days' practice.

### 4.2.2 Reliability

- System failure is less than 5% and system availability is above 80%.
- Several critical bugs less than 10 per year.

### 4.2.3 Performance

- All typical response times are under 20 seconds.
- Other response times are less than 15 seconds.

### 4.2.4 Security

- The system always checks authorization before executing a function.
- The system is divided into 4 roles of users: admins, supervisors, technicians and renters.
- Each role can only access its own set of functions.
- The username must be unique for each role.

## 5. Requirement Appendix

### 5.1 Business Rules

ID	Rule Definition
BR-1	If the account is correct, system redirect users to corresponding home page: <ul style="list-style-type: none"><li>- User logs in as role "admin", the system will display admin Dashboard page</li><li>- User logs in as role "supervisor", the system will display supervisor dashboard page</li><li>- User logs in as role "Renter", the system will display renter home screen</li><li>- User logs in as role "technician", the system will display technician problem list page</li></ul>
BR-2	Username and phone must be unique for user
BR-3	Phone number must be 10 digits
BR-4	Email must be in standard email format
BR-5	User can only update their profile
BR-6	User can only change their password
BR-7	Banned account cannot interact with the system until it is unbanned
BR-8	Account profile can be updated by only it's owner
BR-9	Only the admin role manages supervisor accounts

<b>BR-10</b>	Only the admin role manages technician accounts
<b>BR-11</b>	Only the admin role manages areas
<b>BR-12</b>	When adding a new area, name location and status field is required
<b>BR-13</b>	When an area has active buildings, it cannot be deactivated
<b>BR-14</b>	Only the admin role and supervisor role manage technician accounts
<b>BR-15</b>	All contracts the system manages were already physically signed by renters in person
<b>BR-16</b>	Only the supervisor role manages his/her building
<b>BR-17</b>	The supervisor role can manage contracts in the his/her building
<b>BR-18</b>	The number of contracts in a flat must not exceed the total number of slots a flat has
<b>BR-19</b>	The supervisor can add a contract only into a flat when that flat is not fully rented
<b>BR-20</b>	When adding a new contract, can choose either an existed renter who has signed another contract with VinFlat or a new renter
<b>BR-21</b>	Only the supervisor role manages flat types in his/her building
<b>BR-22</b>	A flat type rules the max number of rooms a flat can have
<b>BR-23</b>	When a flat type has active flats, it cannot be deactivated
<b>BR-24</b>	Only the supervisor can manage flats in the his/her building
<b>BR-25</b>	When a flat is rented, it cannot be deactivated
<b>BR-26</b>	When all rooms of a flat is fully rented, flat state change to "not available"
<b>BR-27</b>	Only the supervisor role manages room types in his/her building
<b>BR-28</b>	A room's type has the max number of renters can rent that room
<b>BR-29</b>	A room has only one room type
<b>BR-30</b>	When a room type has active rooms, it cannot be deactivated
<b>BR-31</b>	The supervisor can manage rooms in his/her building
<b>BR-32</b>	When the room is rented, it cannot be deactivated
<b>BR-33</b>	The supervisor role can manage service types in the his/her building
<b>BR-34</b>	When a service type has active services, it cannot be deactivated
<b>BR-35</b>	The supervisor manages services
<b>BR-36</b>	When the service is used, it cannot be deactivated
<b>BR-37</b>	The supervisor must create the monthly invoice at the first few days of the month, that invoice must be updated with all the fees at the end of the month
<b>BR-38</b>	Only the supervisor role can confirm an invoice is paid when the invoice has a transaction with picture is included and match the amount of the invoice
<b>BR-39</b>	Renter's monthly invoice must include rental fee, service fee, water fee and electricity fee
<b>BR-40</b>	Renter's electricity fee must be calculated based on the number of renter's present days, room's electricity usage coefficient, the electricity price in the contract and the total amount of electricity the whole flat use
<b>BR-41</b>	Renter's water fee is calculated based on the number of present days, room's water usage coefficient, the water price in the contract and the total amount of water the whole flat use
<b>BR-42</b>	A ticket can be of the following type:

	<p>- "Phàn nàn": To report other renters and employees' misbehaviour or behaviours that violate the term of service</p> <p>- "Sự cố": To report sudden failure of the rental appliances.</p> <p>- "Bảo trì": To report failure of rental appliances due to long usage time and to request repairment or replacment</p> <p>- "Khác": To report problem that does not belong to the types above</p>
<b>BR-43</b>	Only the renter role can send a new ticket
<b>BR-44</b>	If the problem in "Sự cố" ticket is caused by a renter, the repairment fee is added to that renter's invoice
<b>BR-45</b>	Only the supervisor role can approve a ticket in his/her building when the problem is real, and start negotiating price for "Sự cố" type
<b>BR-46</b>	Only the supervisor role can disapprove a ticket when the problem is fake
<b>BR-47</b>	The fee of "Sự cố" ticket must be negotiated by renter and supervisor via 3rd-party message app or in person
<b>BR-48</b>	"Phàn nàn" and "Bảo trì" and "Khác" tickets are solved for free
<b>BR-49</b>	For "Sự cố" and "Khác" problem, if the renter accept the fee, the fee will be added to that month monthly invoice
<b>BR-50</b>	For "Sự cố" and "Khác", if the renter opposes the fee, problem fee negotiation will restart.
<b>BR-51</b>	Only the supervisor assigns a technician to resolve a ticket for all ticket types or the technician can choose which ticket to resolve
<b>BR-52</b>	The technician can choose only "Sự cố" and "Bảo trì" ticket to resolve
<b>BR-53</b>	When there are contracts which is due in less than 1 month, the application and supervisor must notify the renter
<b>BR-54</b>	To extend the contract, the renter must sign a contract extension, if not the contract will close
<b>BR-55</b>	Only the supervisor role and technician role can record electricity meter of a flat
<b>BR-56</b>	When recording electricity meter, the supervisor or the technician must include at least 1 picture as proof
<b>BR-57</b>	Only the supervisor role and technician role can record water meter of a flat
<b>BR-58</b>	When recording water meter, the supervisor or the technician must include at least 1 picture as proof
<b>BR-59</b>	The maximum number of active contracts a renter has is 1
<b>BR-60</b>	The renter role must notify contract cancelation in advance of at least 1 month
<b>BR-61</b>	Contract extension/cancelation is managed by renter and supervisor via either 3rd-party message application or in person
<b>BR-62</b>	The renter's flat/room information must include provided utilities, flat mate list
<b>BR-63</b>	Only the renter role can request new service
<b>BR-64</b>	The renter role can request to be absent at most 1 month to reduce the amount he/she pay for electricity and water
<b>BR-65</b>	Renter's absent request must be approved by the building supervisor to be in effect
<b>BR-66</b>	The absent period can be updated or removed only when it hasn't started
<b>BR-67</b>	Renter confirms paying invoice with the supervisor via either 3rd-party message application or in person
<b>BR-68</b>	Renter must be reminded to pay the invoices which is due in less than 1 month



<b>BR-69</b>	Renter who fails to pay any invoice in time will be fined from 2% to 8% of that invoice's total amount with the increment of 1% each time
<b>BR-70</b>	The fine for late payment will be added to the next month invoice
<b>BR-71</b>	The renter role can confirm the invoice payment by sending an image of the transaction
<b>BR-72</b>	The renter must upload his/her citizenship card's front side image when renter login the first time
<b>BR-73</b>	The renter citizenship number is 10 or 12 digits
<b>BR-74</b>	Renter and employee must follow the terms of contract
<b>BR-75</b>	The room will have min slot at 1
<b>BR-76</b>	Supervisor can only update rent, water, electricity and service price in contracts
<b>BR-77</b>	Maximum tickets that a renter can have been 10
<b>BR-78</b>	Max room slot cannot exceed room capacity
<b>BR-79</b>	Max room slot can not exceed room capacity
<b>BR-80</b>	Available room slot will not exceed room capacity or max room slot
<b>BR-81</b>	Renter is unique with each contract
<b>BR-82</b>	Only supervisor can create renter
<b>BR-83</b>	Renter be created through contract
<b>BR-84</b>	When create renter, citizen number and citizen image card is required.
<b>BR-85</b>	When supervisor update renter account, just status field will be update.
<b>BR-86</b>	Invoice name is required.
<b>BR-87</b>	Renter must be in the building and has an active contract.
<b>BR-88</b>	Minimum days of contract is 180 days (6 months).
<b>BR-89</b>	Renter can only cancel the ticket if the ticket hasn't approved by supervisor or technician.
<b>BR-90</b>	
<b>BR-91</b>	

Table 22- Business Rules

## 5.2 Common Requirements

**CR-1** Login: User can log in to the system

**CR-2** Logout: User can sign out of system

**CR-3** View profile: User can view information of profile

**CR-4** Update profile: User can update the information of profile

**CR-5** Change password: User can change password

## 5.3 Application Messages List

#	Message code	Message Type	Context	Content
1	MSG01	Toast message	This manager is not managing any buildings	<i>Quản lí này hiện đang không quản lí tòa nhà nào</i>
2	MSG02	Toast message	This manager is managing more	<i>Quản lí này hiện đang quản</i>

			than 1 building	<i>lĩ hơn 1 toà nhà</i>
3	MSG03	Toast message	Unable to update management for the building	<i>Không thể cập nhật quản lý cho toà nhà</i>
4	MSG04	Toast message	This building does not exist in the system	<i>Toà nhà này không tồn tại trong hệ thống</i>
5	MSG05	Toast message	Staff not found in the building	<i>Nhân viên không tìm thấy trong toà nhà</i>
6	MSG06	Toast message	More than 1 manager is managing this building	<i>Nhiều hơn 1 quản lý đang quản lý toà nhà này</i>
7	MSG07	Toast message	Staff does not exist	<i>Nhân viên không tồn tại</i>
8	MSG08	Toast message	Staff creation failed	<i>Tạo nhân viên thất bại</i>
9	MSG09	Toast message	Staff list is empty	<i>Danh sách nhân viên trống</i>
10	MSG10	Toast message	The technician does not support any buildings	<i>Người kĩ thuật viên không hỗ trợ tòa nhà nào</i>
11	MSG11	Toast message	Invoice does not exist	<i>Hoá đơn không tồn tại</i>
12	MSG12	Toast message	Invoice list is empty	<i>Danh sách hoá đơn trống</i>
13	MSG13	Toast message	Invoice update failed	<i>Cập nhật hoá đơn thất bại</i>
14	MSG14	Toast message	Invoice creation failed	<i>Hoá đơn tạo thất bại</i>
15	MSG15	Toast message	Invoice type does not exist	<i>Loại hoá đơn không tồn tại</i>
16	MSG16	Toast message	There are no contracts	<i>Không có hợp đồng nào</i>
17	MSG17	Toast message	The contract does not exist in the system	<i>Hợp đồng không tồn tại trong hệ thống</i>
18	MSG18	Toast message	This user currently has no contract or is renting	<i>Người dùng này hiện tại không có hợp đồng nào hoặc đang thuê</i>
19	MSG19	Toast message	Area creation failed	<i>Tạo khu vực không thành công</i>
20	MSG20	Toast message	Area not found	<i>Khu vực không tìm thấy</i>
21	MSG21	Toast message	Area list is empty	<i>Danh sách khu vực trống</i>
22	MSG22	Toast message	List of available rooms	<i>Danh sách phòng trống</i>
23	MSG23	Toast message	Room not found	<i>Không tìm thấy phòng</i>
24	MSG24	Toast message	The room has been updated	<i>Phòng đã được cập nhật</i>
25	MSG25	Toast message	List of available room types	<i>Danh sách loại phòng trống</i>
26	MSG26	Toast message	Room type not found	<i>Không tìm thấy loại phòng</i>
27	MSG27	Toast message	Room type creation failed	<i>Tạo loại phòng thất bại</i>

28	MSG28	Toast message	List of apartments is empty	<i>Danh sách căn hộ hiện đang trống</i>
29	MSG29	Toast message	Flat created successful	<i>Tạo căn hộ thành công</i>
30	MSG30	Toast message	This flat does not exist	<i>Căn hộ này không tồn tại</i>
31	MSG31	Toast message	Flat update failed	<i>Cập nhật căn hộ thất bại</i>
32	MSG32	Toast message	List of flat type currently available	<i>Danh sách loại căn hộ hiện đang trống</i>
33	MSG33	Toast message	Flat type created successful	<i>Tạo loại căn hộ thành công</i>
34	MSG34	Toast message	This flat type does not exist	<i>Loại căn hộ này không tồn tại</i>
35	MSG35	Toast message	Flat type update failed	<i>Cập nhật loại căn hộ thất bại</i>
36	MSG36	Toast message	The total number of rooms cannot exceed the limit of the flat type	<i>Tổng số phòng không được quá giới hạn của loại căn hộ</i>
37	MSG37	Toast message	Incorrect username, phone number or password	<i>Tên đăng nhập, số điện thoại hoặc mật khẩu không đúng</i>
38	MSG38	Toast message	This account is invalid	<i>Tài khoản không hợp lệ</i>
39	MSG39	Toast message	User not found	<i>Không tìm thấy người dùng</i>
40	MSG40	Toast message	Ticket list is empty	<i>Danh sách yêu cầu trống</i>
41	MSG41	Toast message	This flat has no coupons	<i>Người thuê này hiện tại không có yêu cầu nào</i>
42	MSG42	Toast message	No ticket found	<i>Không có yêu cầu nào được tìm thấy</i>
43	MSG43	Toast message	Ticket does not exist	<i>Yêu cầu này không tồn tại</i>
44	MSG44	Toast message	Can only confirm when the status is processed	<i>Chỉ có thể xác nhận khi trạng thái là đã xử lý</i>
45	MSG45	Toast message	Can't find this ticket from user account	<i>Không tìm thấy yêu cầu này từ tài khoản người dùng</i>
46	MSG46	Toast message	No data found	<i>Không tìm thấy dữ liệu</i>
47	MSG47	Toast message	List found	<i>Hiển thị danh sách</i>
48	MSG48	Toast message	Requested detail found	<i>Đã tìm thấy</i>
49	MSG49	Toast message	Showing total water and electricity used in building	<i>Hiển thị tổng số điện nước</i>

50	MSG50	Toast message	Showing total renter in building	<i>Hiện thị tổng số khách thuê</i>
51	MSG51	Toast message	Create new employees successful	<i>Tạo mới nhân viên thành công</i>
52	MSG52	Toast message	Area created successfully	<i>Khu vực được tạo thành công</i>
53	MSG53	Toast message	This apartment is currently empty	<i>Căn hộ này hiện đang trống</i>
54	MSG54	Toast message	Flat type created successful	<i>Loại căn hộ tạo thành công</i>
55	MSG55	Toast message	Room type created successful	<i>Loại phòng tạo thành công</i>
56	MSG56	Toast message	Ticket created successful	<i>Phiếu yêu cầu tạo thành công</i>
57	MSG57	Toast message	Contract created successfully	<i>Hợp đồng được tạo thành công</i>
58	MSG58	Toast message	Updated successfully	<i>Đã cập nhật thành công</i>

*Table 23 - Application Messages List*

## IV. Software Design Description

### 1. System Design

#### 1.1 System Architecture

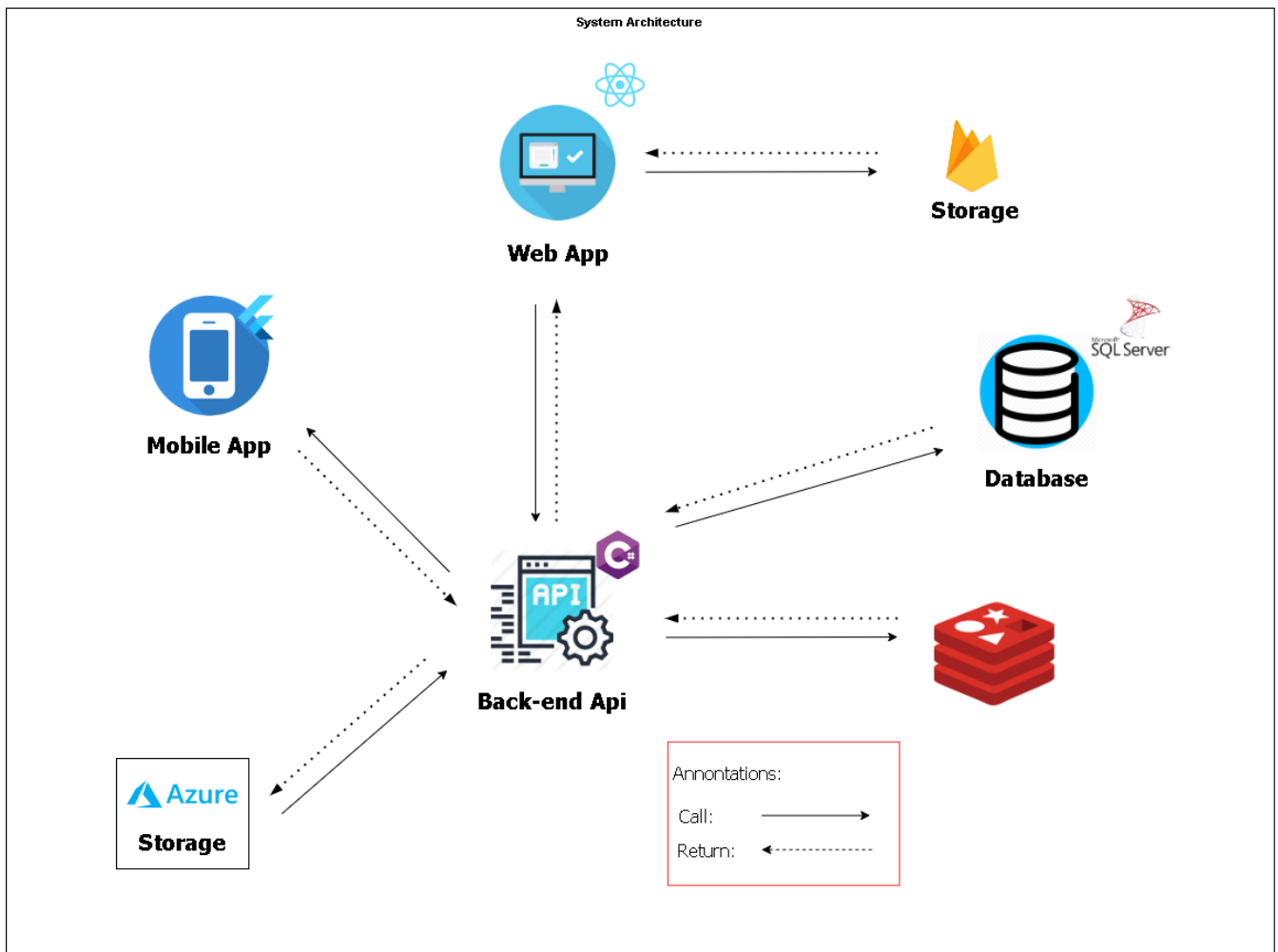


Figure 8 - System Design

## 1.2 Package Diagram

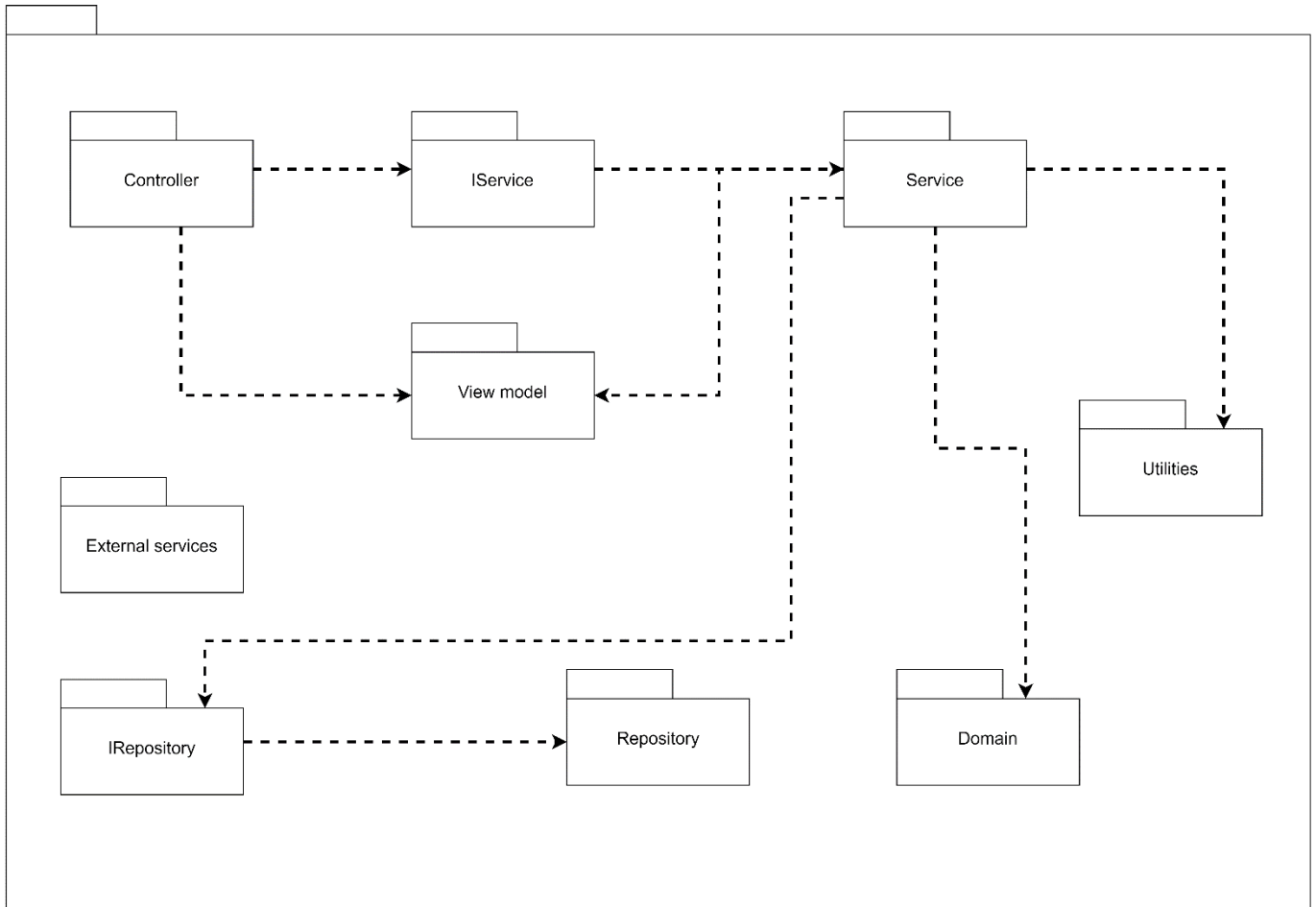


Figure 9 - Package Diagram

## 2. Database Design

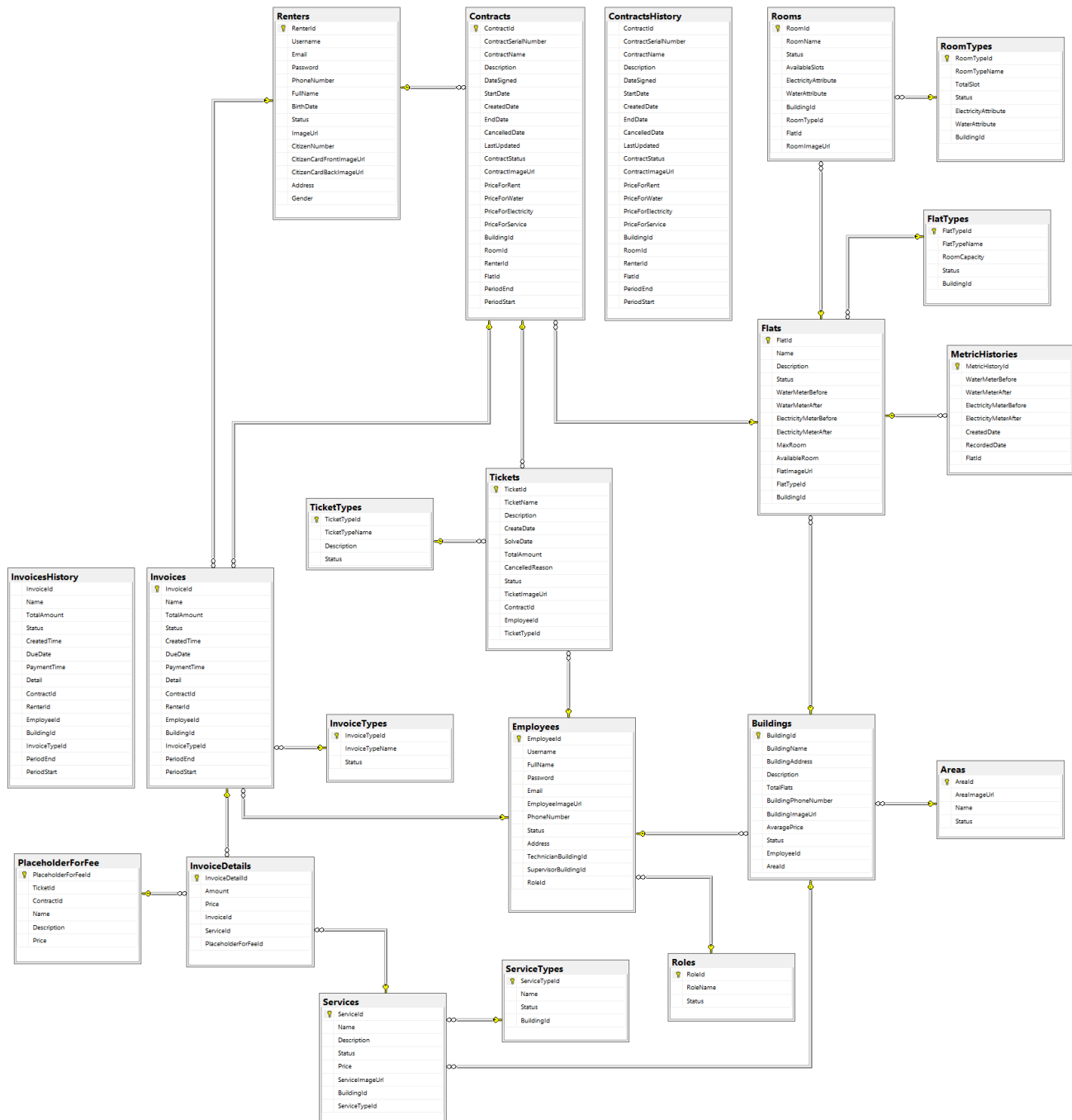


Figure 10 - Database Diagram

### **Table Descriptions**

No	Table	Description
01	Area	Describe area in the system
02	Building	Describe building in the system
03	Contract	Describe contract in the system
04	ContractHistory	Describe contract history in the system
05	Employee	Describe employee in the system
06	Flat	Describe flat in the system
07	Flat type	Describe flat type in the system
08	Invoice	Describe invoice in the system
09	InvoiceHistory	Describe invoice history in the system
10	Invoice detail	Describe invoice detail in the system
11	Invoice type	Describe invoice type in the system
12	Metric history	Describe metric history in the system
13	Placeholder for fee	Describe placeholder for fee in the system
14	Renter	Describe renter in the system
15	Role	Describe role in the system
16	Room	Describe room for building in the system
17	Room type	Describe room type for building in the system
18	Service entity	Describe service entity for building in the system
19	Service type	Describe service type for building in the system
20	Ticket	Describe ticket for renter in the system
21	Ticket type	Describe message between mod and translator or customer

Table 24 24 - Table Description

### **Attribute Data Dictionary**

Entity Name	Attributes	Description	Domain	Null
Area	AreaId{PK}	Unique identifier of area	int	No



	Name	Area name	nvarchar(100)	No
	Status	Area status	boolean	No
	AreaImageUrl	Area image Urls	byte[]	No
Building	BuildingIdPK}	Unique identifier of building	int	No
	BuildingName	Building name	nvarchar(100)	No
	BuildingDescription	Building description	nvarchar(100)	No
	TotalFlats	Total flats in building	int	No
	BuildingPhoneNumber	Building phone number	varchar(14)	No
	BuildingImageUrl	Building image urls	byte[]	No
	AveragePrice	Average price of building	decimal(18,2)	No
	Status	Building status	boolean	No
	EmployeeId	Foreign key, identifier of employee	int	Yes
	AreaId	Foreign key, identifier of area	int	No
Contract	ContractId{PK}	Unique identifier of contract	varchar(50)	No
	ContractSerialNumber	Contract serial	nvarchar(100)	No

		number		
	ContractName	Name of contract	nvarchar(100)	No
	Description	Description of contract	nvarchar(100)	Yes
	DateSigned	Signed date in contract	datetime	No
	StartDate	Started date in contract	datetime	No
	CreatedDate	Created date in contract	datetime	No
	EndDate	End date in contract	datetime	Yes
	CancelledDate	Cancel date in contract	datetime	Yes
	LastUpdated	Last update date in contract	datetime	No
	ContractImageUrl	Contract image urls	byte[]	No
	PriceForRent	Rent price in contract	decimal(18,2)	No
	PriceForWater	Water price per block in contract	decimal(18,2)	No
	PriceForElectricity	Electric price per watt in contract	decimal(18,2)	No
	PriceForService	Service price for building in contract	decimal(18,2)	No
	BuildingId	Unique identifier of building	int	No
	RoomId	Unique identifier of	int	No

			room		
		RenterId	Unique identifier of renter	int	No
		FlatId	Foreign key, identifier of flat	int	No
		PeriodEnd	Record date of end period	datetime	No
		PeriodStart	Record date of start period	datetime	No
ContractHistory		ContractId{PK}	Unique identifier of contract	int	No
	ContractSerialNumber		Contract serial number	nvarchar(100)	No
	ContractName		Name of contract	nvarchar(100)	No
	Description		Description of contract	nvarchar(100)	Yes
	DateSigned		Signed date in contract	datetime	No
	StartDate		Started date in contract	datetime	No
	CreatedDate		Created date in contract	datetime	No
	EndDate		End date in contract	datetime	Yes
	CancelledDate		Cancel date in contract	datetime	Yes

	LastUpdated			Last update date in contract	datetime	No
	ContractImageUrl			Contract image urls	byte[]	No
	PriceForRent			Rent price in contract	decimal(18,2)	No
	PriceForWater			Water price per block in contract	decimal(18,2)	No
	PriceForElectricity			Electric price per watt in contract	decimal(18,2)	No
	PriceForService			Service price for building in contract	decimal(18,2)	No
	BuildingId			Unique identifier of building	int	No
	RoomId			Unique identifier of room	int	No
	RenterId			Unique identifier of renter	int	No
	FlatId			Foreign key, identifier of flat	int	No
	PeriodEnd	Record date of end period	datetime	No		
	PeriodStart			Record date of start period	datetime	No
Employee	EmployeeId{PK}			Unique identifier of employee	int	No
	Username			Username of employee	nvarchar(100)	No

	Fullname	Fullname of employee	nvarchar(100)	No
	Password	Password of employee	nvarchar(100)	No
	Email	Email of employee	nvarchar(100)	No
	EmployeeImageUrl	Image url of employee	byte[]	No
	PhoneNumber	Phone number of employee	nvarchar(14)	No
	Status	Status of employee	boolean	No
	Address	Address of employee in the system	nvarchar(200)	No
	TechnicianBuildingId	Unique identifier of employee with role technician	int	Yes
	SupervisorBuildingId	Unique identifier of employee with role supervisor	int	Yes
	RoleId	Foreign key, identifier of role	int	No
Flat	FlatId{PK}	Unique identifier of flat	int	No
	Name	Name of flat	nvarchar(200)	No
	Description	Description of flat	nvarchar(200)	No
	Status	Status of flat	nvarchar(100)	No

	WaterMeterBefore	Water meter for the previous month	decimal	No
	WaterMeterAfter	Water meter of the current month	decimal	No
	ElectricityMeterBefore	Electricity meter of the previous month	decimal	No
	ElectricityMeterAfter	Electricity meter of the current month	decimal	No
	MaxRoom	Max room of the flat	int	No
	AvailableRoom	Total available room in flat	int	No
	FlatImageUrl	Image url of flats	byte[]	No
	FlatTypeId	Foreign key, identifier of flat type	int	No
	BuildingId	Foreign key, identifier of building	int	No
Flat Type	FlatTypeId{PK}	Unique identifier of flat type	int	No
	FlatTypeName	Name of flat type	nvarchar(100)	No
	RoomCapacity	Capacity of room	int	No
	Status	Status of room	nvarchar(100)	No

	BuildingId	Foreign key, identifier of building	int	No
Invoice	InvoiceId{PK}	Unique identifier of invoice	int	No
	Name	Name of invoice	nvarchar(100)	No
	TotalAmount	Total price of invoice	decimal(18,2)	No
	Status	Status of invoice	boolean	No
	CreatedTime	Created time of invoice	datetime	No
	DueDate	Due date for invoice	datetime	No
	PaymentTime	Payment date for invoice	datetime	No
	Detail	Detail of invoice	nvarchar(200)	No
	ContractId	Foreign key, identifier of contract	int	No
	RenterId	Foreign key, identifier of renter	int	No
	EmployeeId	Foreign key, Unique identifier of employee	int	No
	BuildingId	Unique identifier of building	int	No
	InvoiceTypeId	Foreign	int	No

		key, identifier of invoice type		
	PeriodEnd	Record date of end period	datetime	No
	PeriodStart	Record date of start period	datetime	No
InvoiceHistory	InvoiceId{PK}	Unique identifier of invoice history	int	No
	Name	Name of invoice	nvarchar( 100)	No
	TotalAmount	Total price of invoice	decimal( 18,2)	No
	Status	Status of invoice	boolean	No
	CreatedTime	Created time of invoice	datetime	No
	DueDate	Due date for invoice	datetime	No
	PaymentTime	Payment date for invoice	datetime	No
	Detail	Detail of invocie	nvarchar( 200)	No
	ContractId	Foreign key, identifier of contract	int	No
	RenterId	Foreign key, identifier of renter	int	No
	EmployeeId	Foreign key, Unique identifier of employee	int	No



	BuildingId	Unique identifier of building	int	No	
	InvoiceTypeId	Foreign key, identifier of invoice type	int	No	
	PeriodEnd	Record date of end period	datetime	No	
	PeriodStart	Record date of start period	datetime	No	
InvoiceDetail	InvoiceDetailId{PK}		Unique identifier of translator	varchar (50)	No
	Amount		Total amount of details	int	No
	Price		Price of invoice detail	decimal(12,2)	No
	InvoiceId		Foreign key, identifier of invoice	int	No
	ServiceId		Foreign key, identifier of service	int	Yes
	PlaceHolderForFeeId		Foreign key, identifier of place holder for fee	int	Yes
InvoiceType	InvoiceTypeId{PK}		Unique identifier of invoice type	int	No
	InvoiceTypeName		Name of invoice type	nvarchar(100)	No
	Status		Status of invoice type	boolean	No

MetricHistory	MetricHistory Id{PK}	Unique identifier of achievement	varchar (50)	No
	WaterMeterBefore	Water meter before of metric of history	decimal(18,2)	No
	WaterMeterAfter	Water meter after of metric history	decimal(18,2)	No
	ElectricityMeterBefore	Electricity meter before of metric history	decimal(18,2)	No
	ElectricityMeterAfter	Electricity meter after of metric history	decimal(18,2)	No
	CreatedDate	Created date of metric history	datetime	No
	RecordedDate	Recorded date of metric history	datetime	No
	FlatId	Foreign key, identifier of flat	int	No
PlaceHolderForFee	PlaceHolderForFeeId{PK}	Unique identifier of place holder of fee	int	No
	TicketId	identifier of ticket	int	Yes
	ContractId	identifier of contract	int	Yes
	Name	Name of	nvarchar	No

		place holder	ar(100)	
	Description	Description of place holder	nvarchar(100)	Yes
	Price	Price of place holder	decimal(18,2)	No
Renter	RenterId{PK}	Unique identifier of invitation	varchar(50)	No
	Username	Foreign key, project identifier	varchar(50)	No
	Email	Foreign key, translator identifier	varchar(50)	No
	Password	Invitation creation date	datetime	No
	PhoneNumber	Status of invitation	nvarchar(1)	No
	Fullname	Fullname of renter	nvarchar(100)	No
	BirthDate	Birthdate of renter	datetime	No
	Status	Status of renter	boolean	No
	ImageUrl	Image url of renter profile	byte[]	No
	CitizenNumber	Citizen number of renter	nvarchar(20)	No
	CitizenCardFrontImageUrl	Image url for the front side of citizen image	byte[]	No
	CitizenCardBackImageUrl	Image url for the back	byte[]	No

		side of citizen image		
	Address	Address of renter	nvarchar(100)	No
	Gender	Gender of renter	nvarchar(100)	No
Role	RoleId{PK}	Unique identifier of role	int	No
	RoleName	Name of role	nvarchar(50)	No
	Status	Status of role	boolean	No
Room	RoomId{PK}	Unique identifier of room	int	No
	RoomName	Name of room	nvarchar(100)	No
	Status	Status of room	nvarchar(100)	No
	AvailableSlots	Available slot of room	int	No
	ElectricityAttribute	Electricity attribute of room type	decimal(18,2)	No
	WaterAttribute	Water attribute of room type	decimal(18,2)	No
	BuildingId	Unique identifier of building	int	No
	RoomTypeId	Unique identifier of room type	int	No
	FlatId	Unique identifier of flat	int	Yes
	RoomImageUrls	Image url	byte[]	No

		of rooms		
RoomType	RoomTypeId{PK}	Unique identifier of room type	int	No
	RoomTypeName	Name of room type	nvarchar(100)	No
	TotalSlot	Total slot in room type	nvarchar(100)	No
	Status	Status of room type	nvarchar(100)	No
	ElectricityAttribute	Electricity attribute of room type	decimal(18,2)	No
	WaterAttribute	Water attribute of room type	decimal(18,2)	No
	BuildingId	Unique identifier of building	int	No
ServiceEntity	ServiceId{PK}	Unique identifier of service	int	No
	Name	Foreign key, account identifier	nvarchar(100)	No
	Description	Description of service entity	nvarchar(200)	No
	Status	Status of service entity	boolean	No
	Price	Price of service entity	decimal(18,2)	No
	ServiceImageUrl	Image url of service	byte[]	No

	BuildingId	Foreign key, Unique identifier of translator's language	int	No
	ServiceTypeId	Foreign key, Unique identifier of service type	int	No
ServiceType	TicketTypeId{PK}	Unique identifier of ticket type	int	No
	Name	Name of service type	nvarchar(100)	No
	Status	Status of service type	nvarchar(100)	No
	BuildingId	Unique identifier of building	int	No
Ticket	TicketId{PK}	Unique identifier of ticketp	int	No
	TicketName	Foreign key, account identifier	nvarchar(100)	No
	Description	Description of ticket	nvarchar(200)	No
	CreatedDate	Type of transaction	datetime	No
	SolveDate	Transaction creation date	datetime	No
	TotalAmount	Total price of ticket	decimal(18,2)	Yes
	CancelledReason	Cancelled reason of ticket	nvarchar(100)	Yes
	Status	Status of ticket	nvarchar(100)	No

	TicketImageUrl	Image url of ticket	byte[]	No
	ContractId	Foreign key, Unique identifier of contract	int	No
	EmployeeId	Foreign key, Unique identifier of employee	int	No
	TicketTypeId	Foreign key, Unique identifier of ticket type	int	No
TicketType	TicketTypeId{PK}	Unique identifier of ticket type	int	No
	TicketTypeName	Name of ticket type	nvarchar(100)	No
	Description	Description of ticket type	nvarchar(200)	No
	Status	Status of ticket type	boolean	No

Table 25 25 - Attribute Data Dictionary

### 3. Detailed Design

#### 3.1 Admin Login Feature

##### 3.1.1 Class diagram

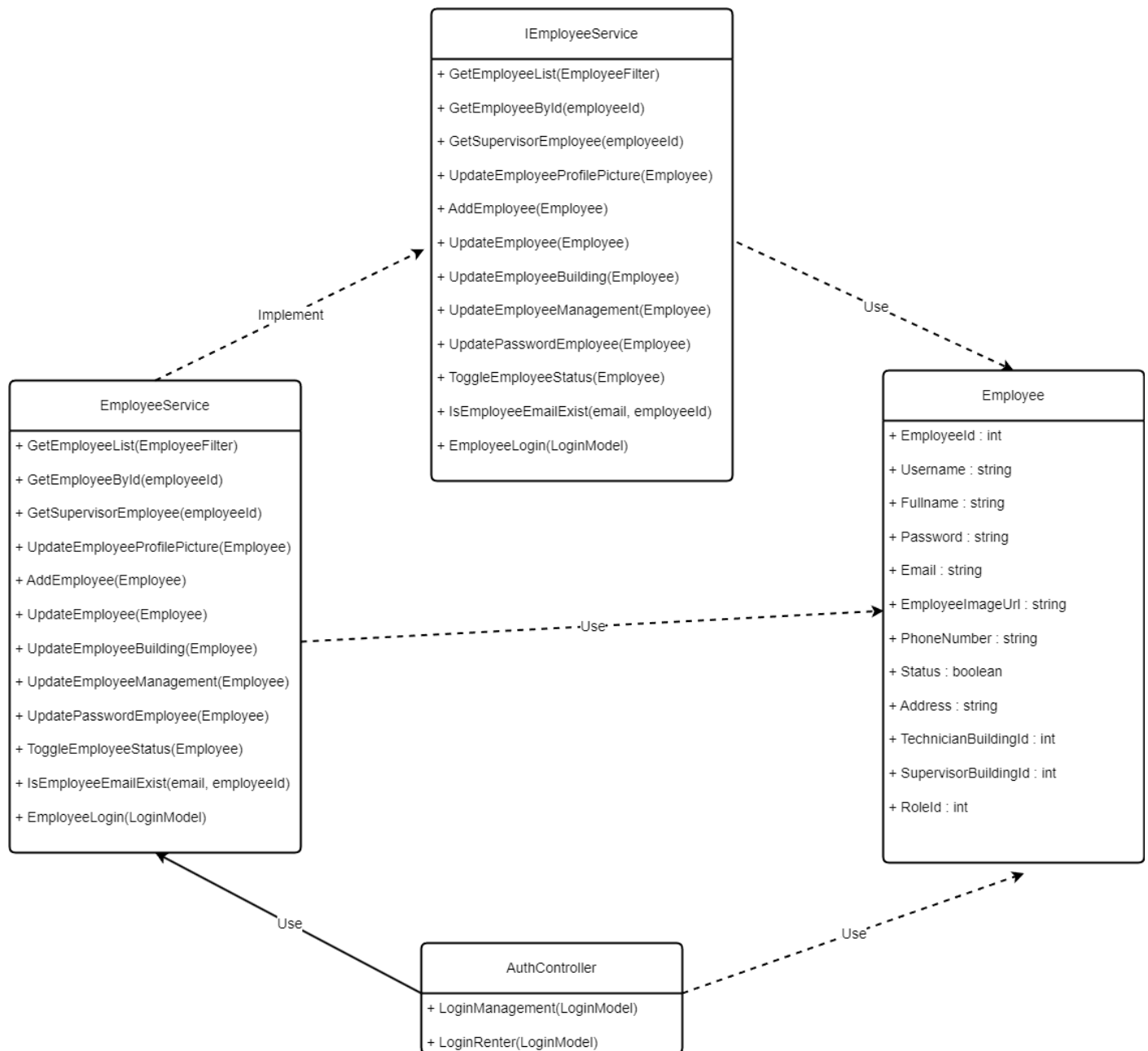


Figure 11 - Admin Login Feature Class Diagram

##### 3.1.2 Class specification

No	Method	Description
1	loginManagement(LoginModel)	Login as management
2	loginEmployee(LoginModel)	Login as employee



### 3.1.3 Login Activity Diagram

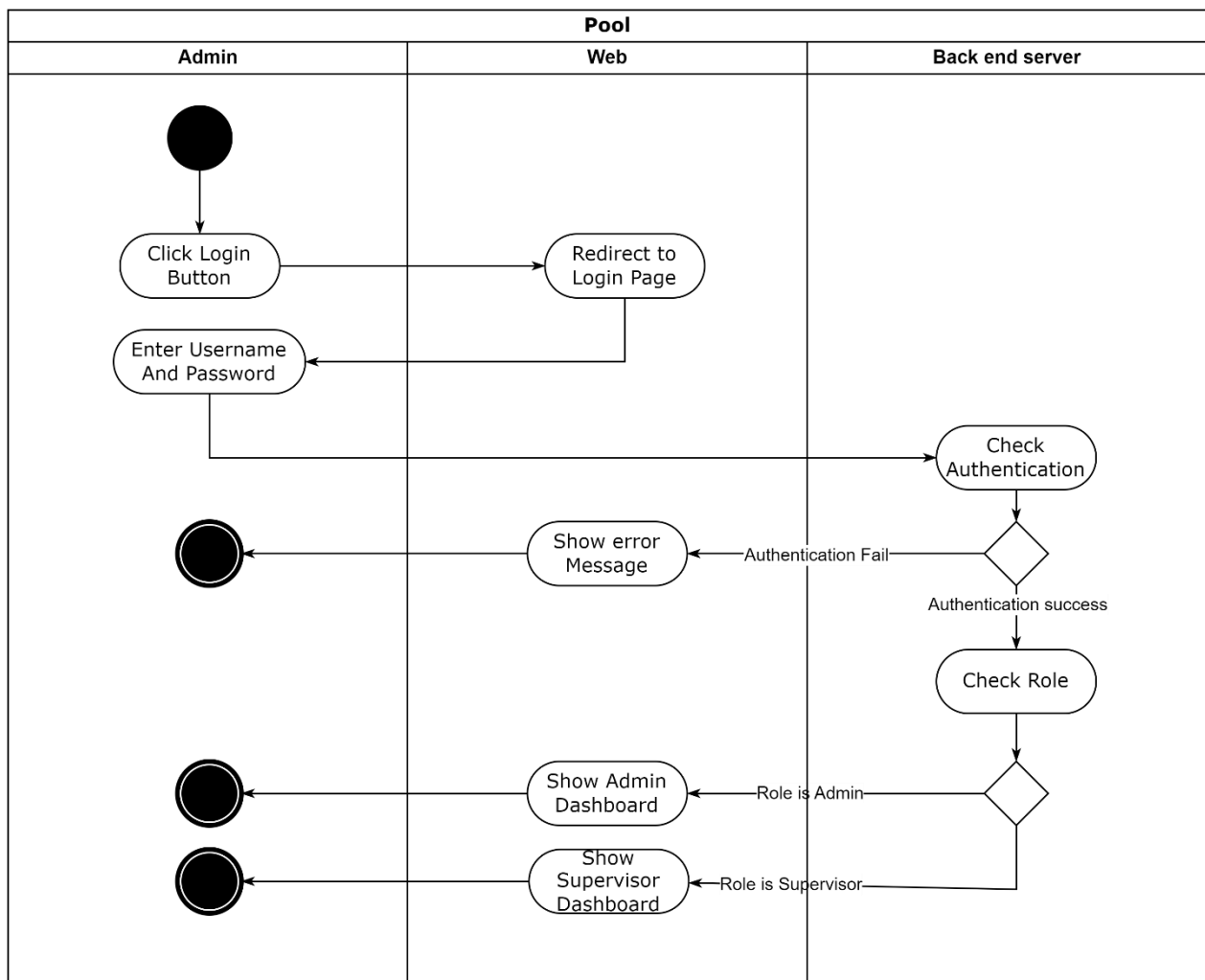


Figure 12 - Login Activity Diagram

### 3.1.4 Login Sequence Diagram

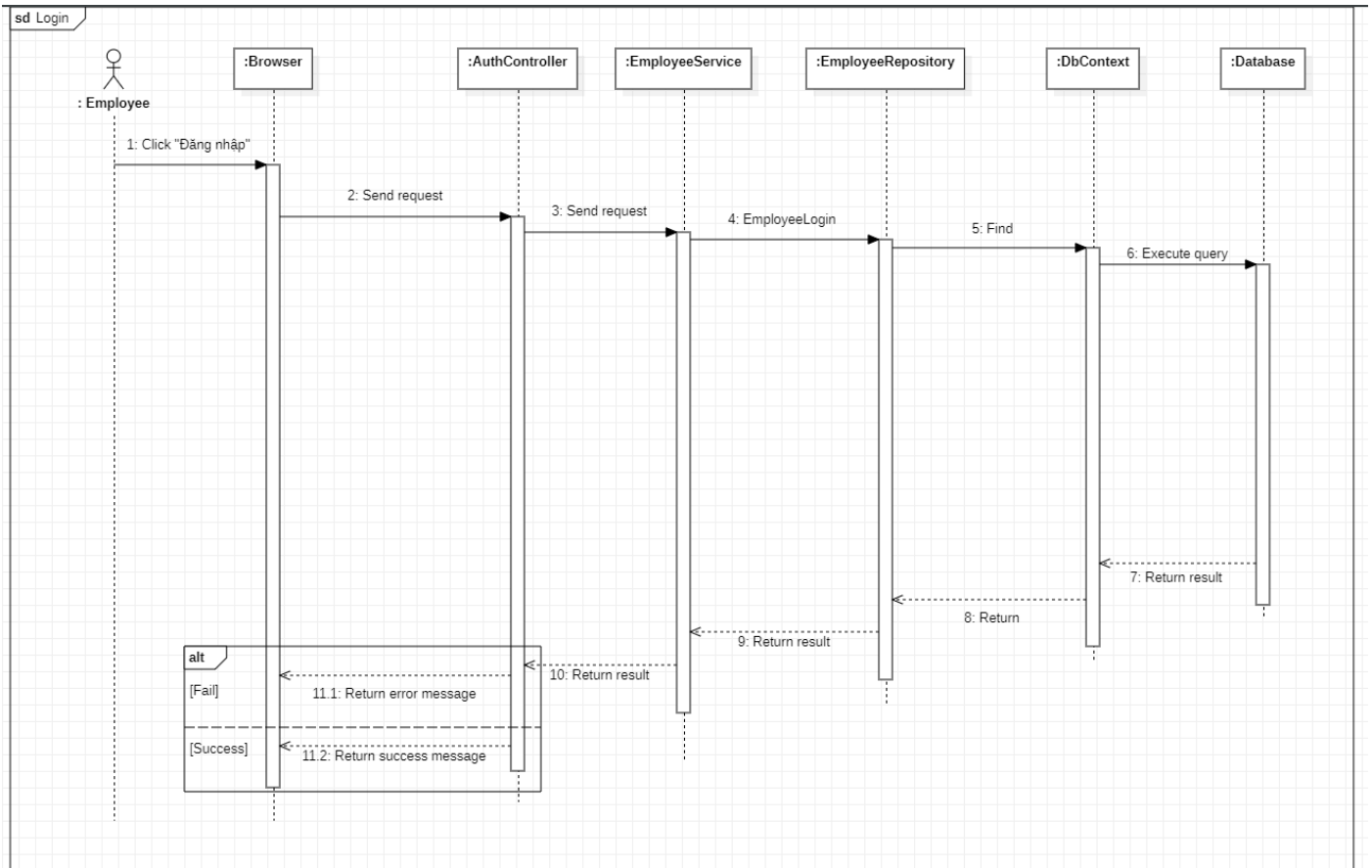


Figure 13 - Login Sequence Diagram

## 3.2 Supervisor Manage Building Feature

### 3.2.1 Class Diagram

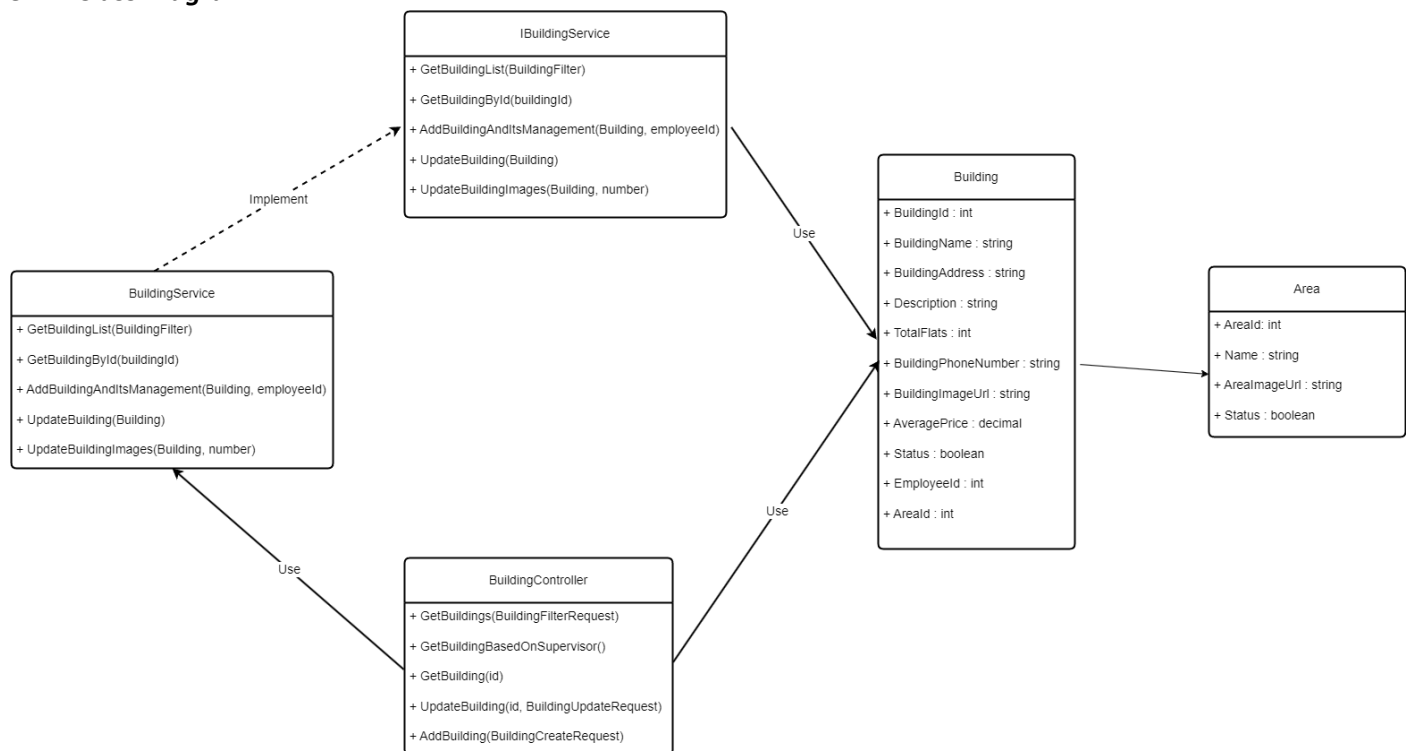


Figure 14 - Supervisor Manage Building Class Diagram

No	Method	Description
1	GetBuilding(BuildingFilterRequest)	Get building list
2	GetBuildingBasedOnSupervisor()	Get building based on logged in supervisor
3	GetBuilding(id)	Get building based on id
4	UpdateBuilding(id, BuildingUpdateRequest)	Update building
5	AddBuilding(BuildingCreateRequest)	Add building

### 3.2.2 Class Diagram Specification

### 3.2.3 Supervisor Create Building Activity Diagram

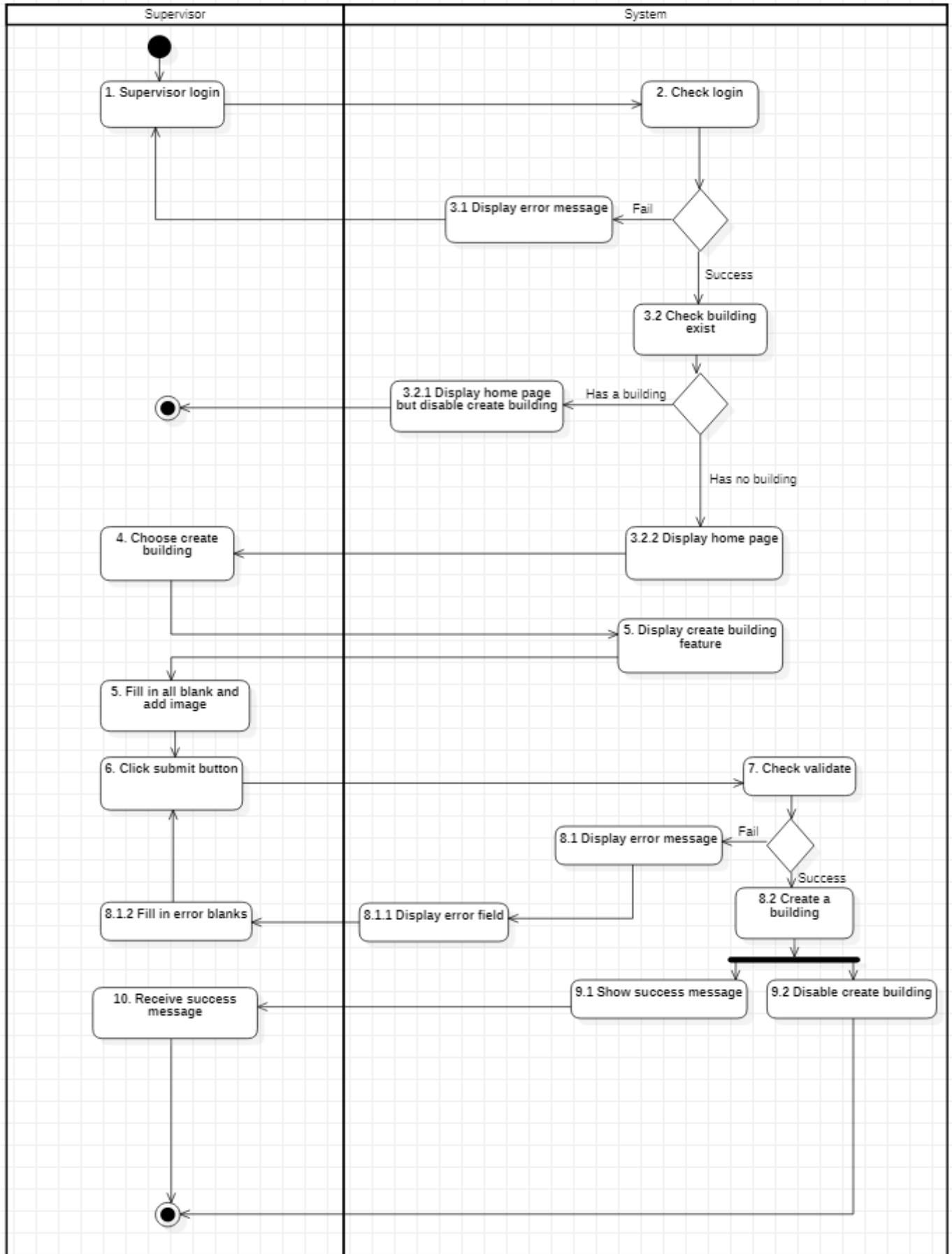


Figure 15 - Supervisor Create Building Activity Diagram

### 3.2.4 Supervisor Create Building Sequence Diagram

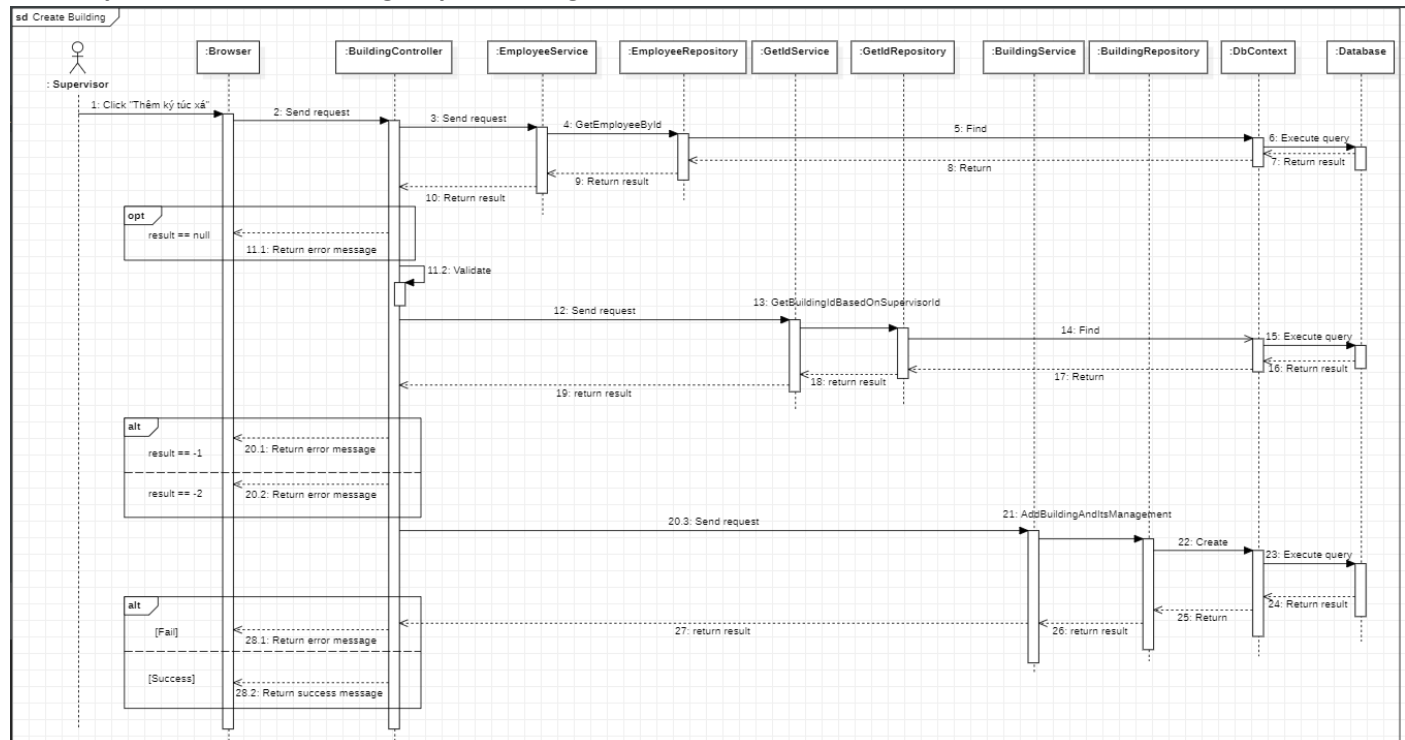


Figure 16 - Supervisor Create Building Sequence Diagram

### 3.3 Supervisor Manage Flat / Room

#### 3.3.1 Class Diagram

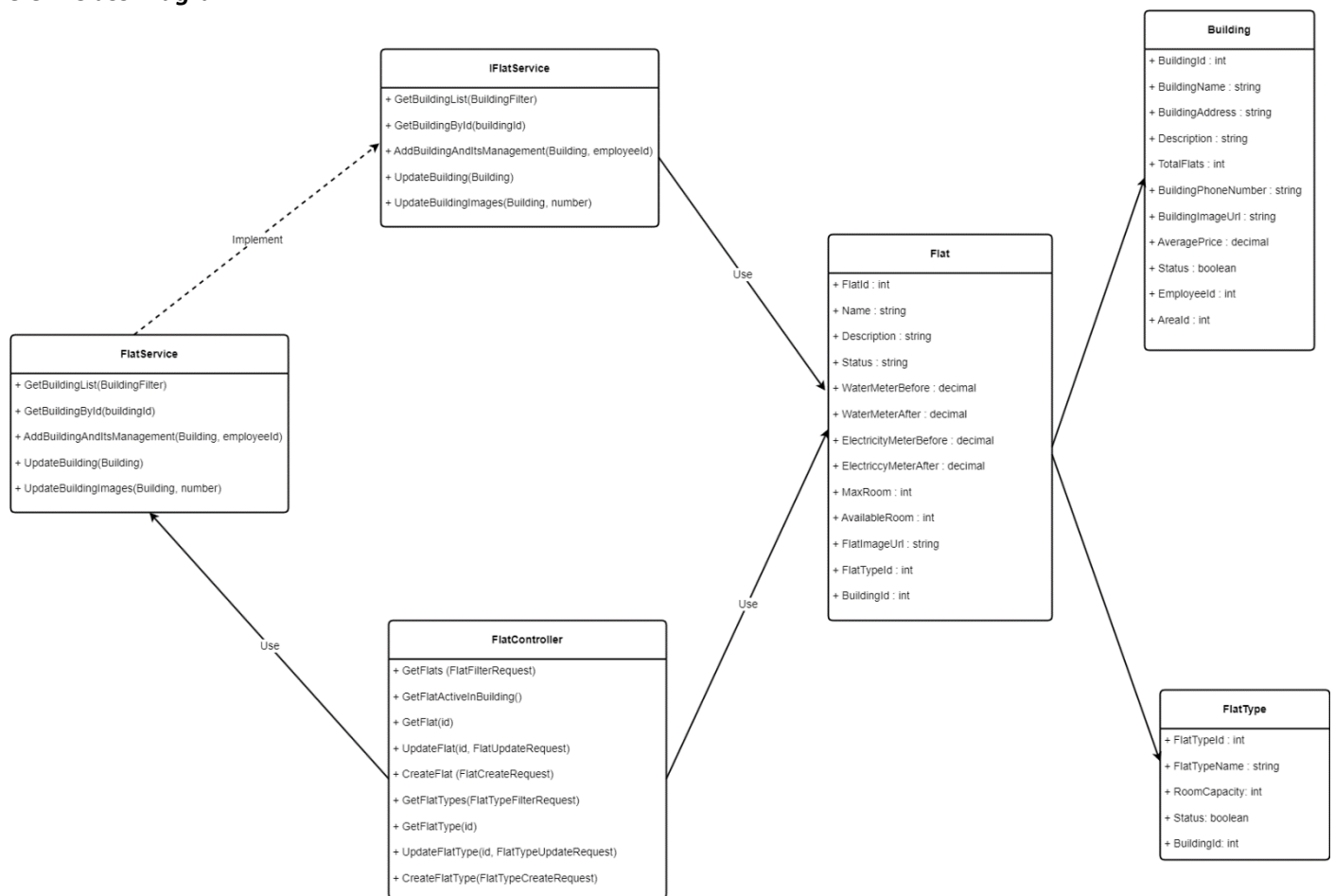


Figure 17 - Supervisor Manage Flat / Room Class Diagram

#### 3.3.2 Class Diagram Specification

No	Method	Description
1	GetFlat(FlatFilterRequest)	Get flat list
2	GetFlatActiveInBuilding()	Get active flats in building
3	GetFlat(id)	Get flat id
4	UpdateFlat(id, FlatUpdateRequest)	Update flat
5	CreateFlat(FlatCreateRequest)	CreateFlat
6	GetFlatTypes(FlatTypeFilterRequest)	Get flat types
7	GetFlatType(id)	Get flat type by id
8	UpdateFlatType(id, FlatTypeUpdateRequest)	Update flat type
9	CreateFlatType(FlatTypeCreateRequest)	Create falt type

### 3.3.3 Supervisor Create Flat Activity Diagram

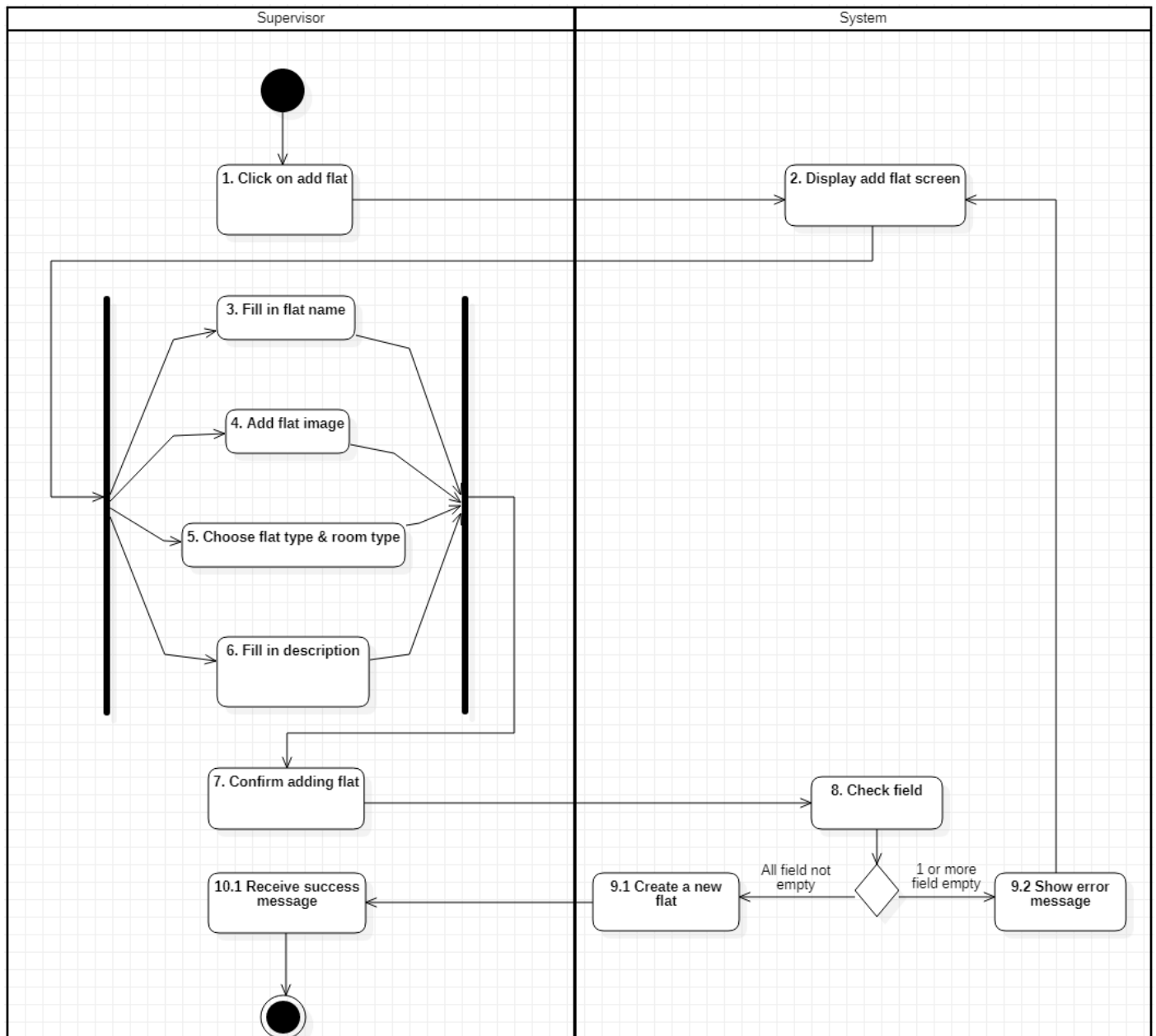


Figure 18 - Supervisor Create Flat Activity Diagram

### 3.3.4 Sequence Create Flat Activity Diagram

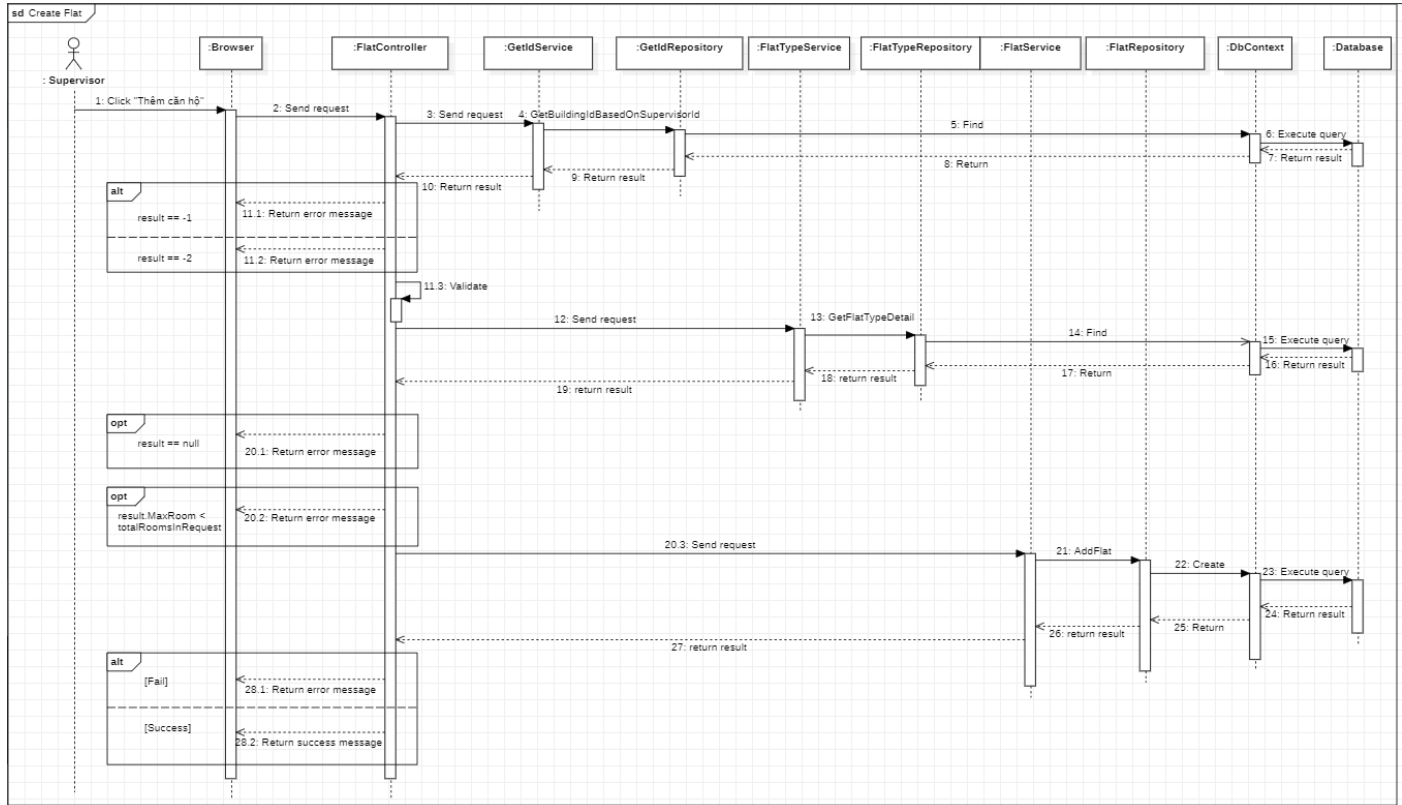


Figure 19 - Sequence Create Flat Activity Diagram



### 3.4 Supervisor Manage Contract

#### 3.4.1 Class Diagram

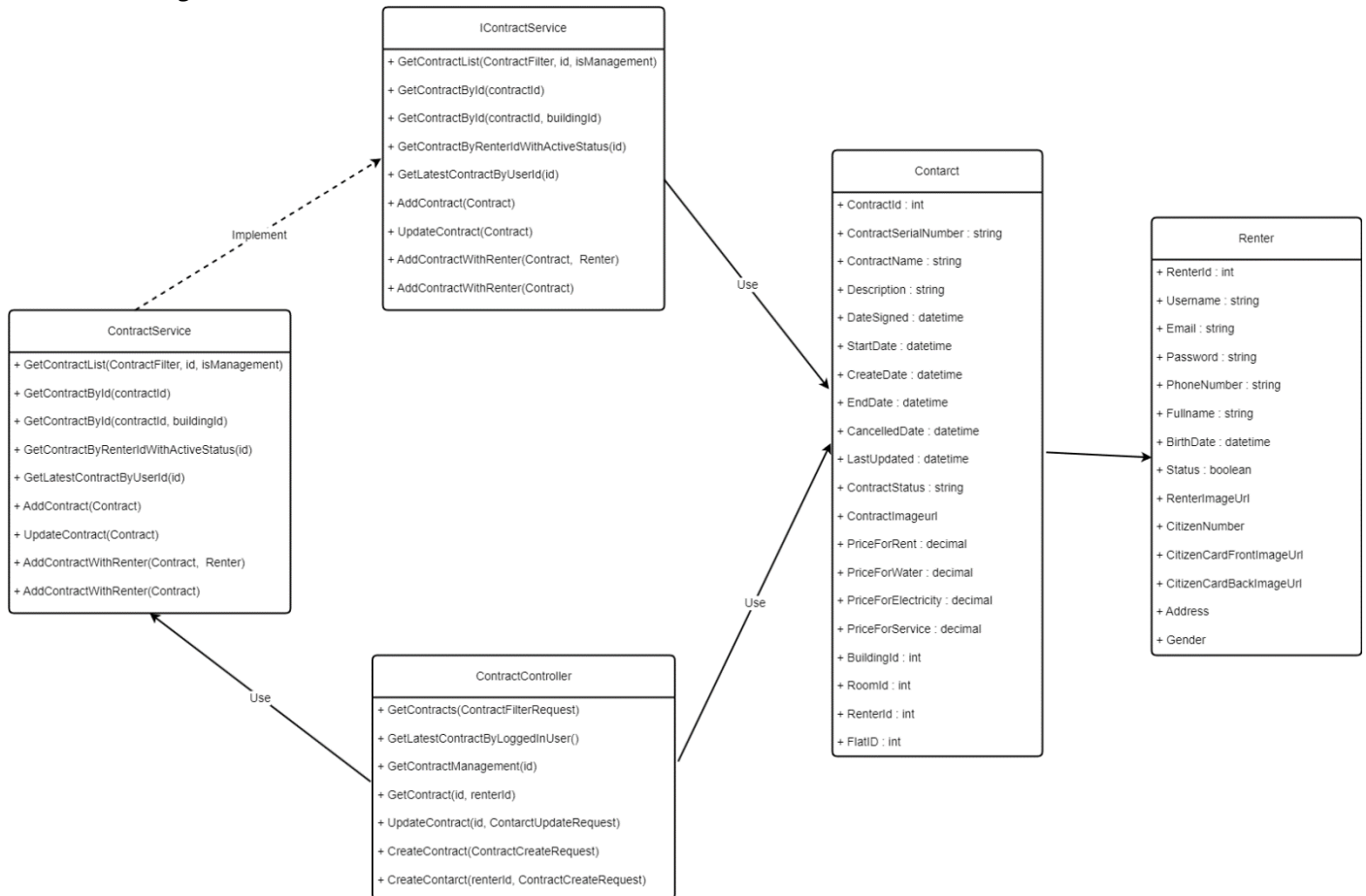


Figure 20 - Supervisor Manage Contract Class Diagram

#### 3.4.2 Class Diagram Specification

No	Method	Description
1	GetContracts(FlatFilterRequest)	Get contract list
2	GetLatestContractByLoggedInUser()	Get latest contract by user
3	GetContractByManagement(id)	Get contract by management
4	UpdateContract(id, ContractUpdateRequest)	Update contract
5	GetContract(id, renterId)	Get contract by renter and its id
6	CreateContract(ContractCreateRequest)	Create contract with new renter
7	CreateContract(renterid, ContractCreateRequest)	Create contract with existing renter

### 3.4.3 Supervisor Create Renter by Contract Activity Diagram

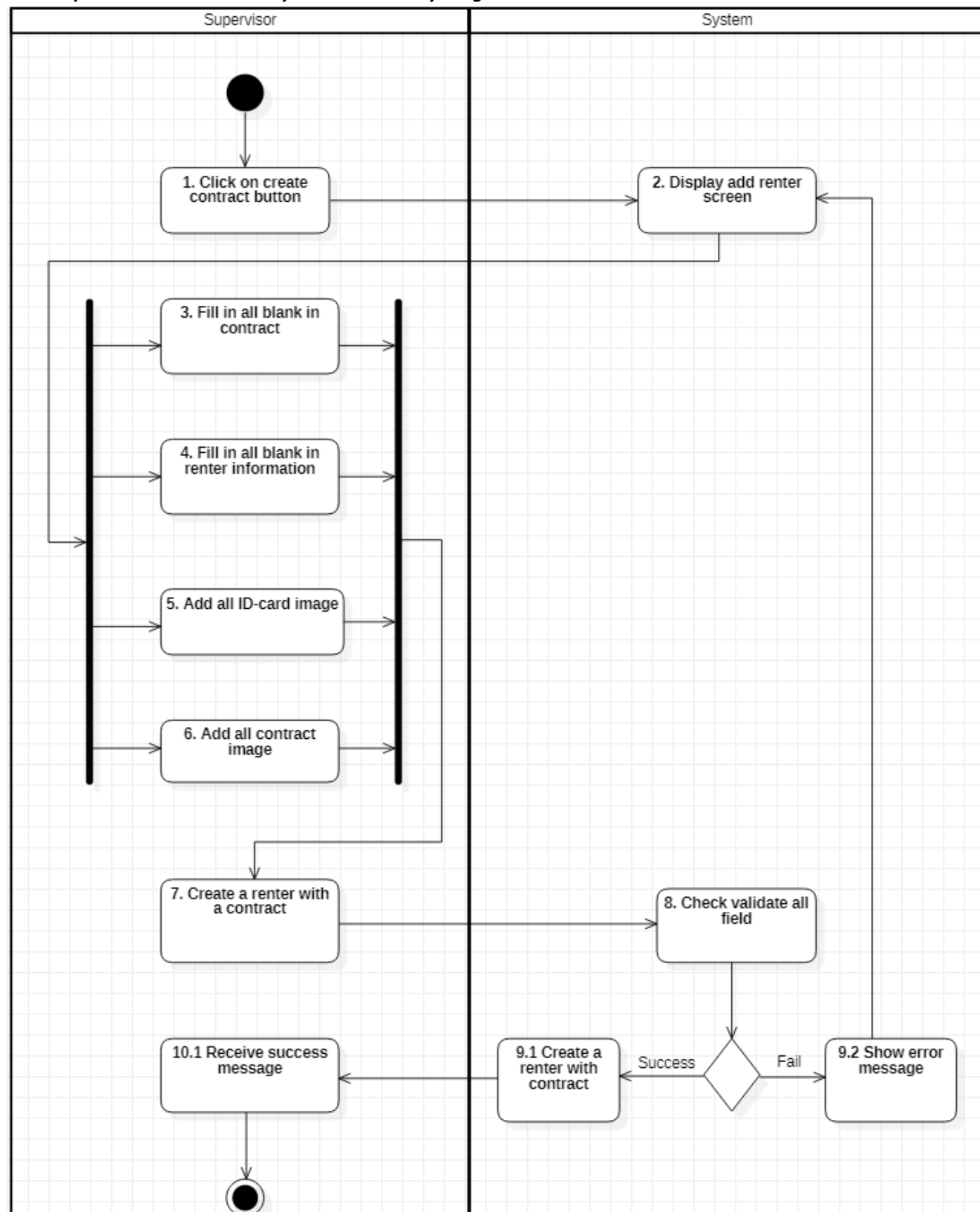


Figure 21 - Supervisor Create Renter by Contract Activity Diagram

### 3.4.4 Sequence Create Renter by Contract Activity Diagram

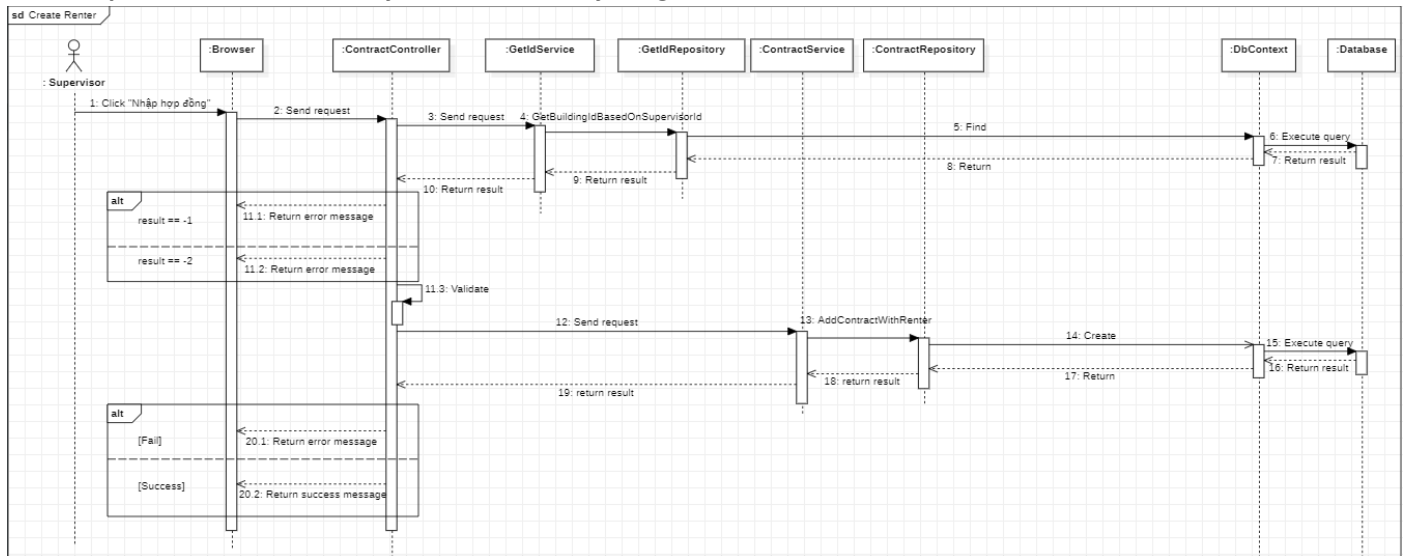


Figure 22 - Sequence Create Renter by Contract Activity Diagram

## 3.5 Supervisor Renew Contract

### 3.5.1 Class Diagram

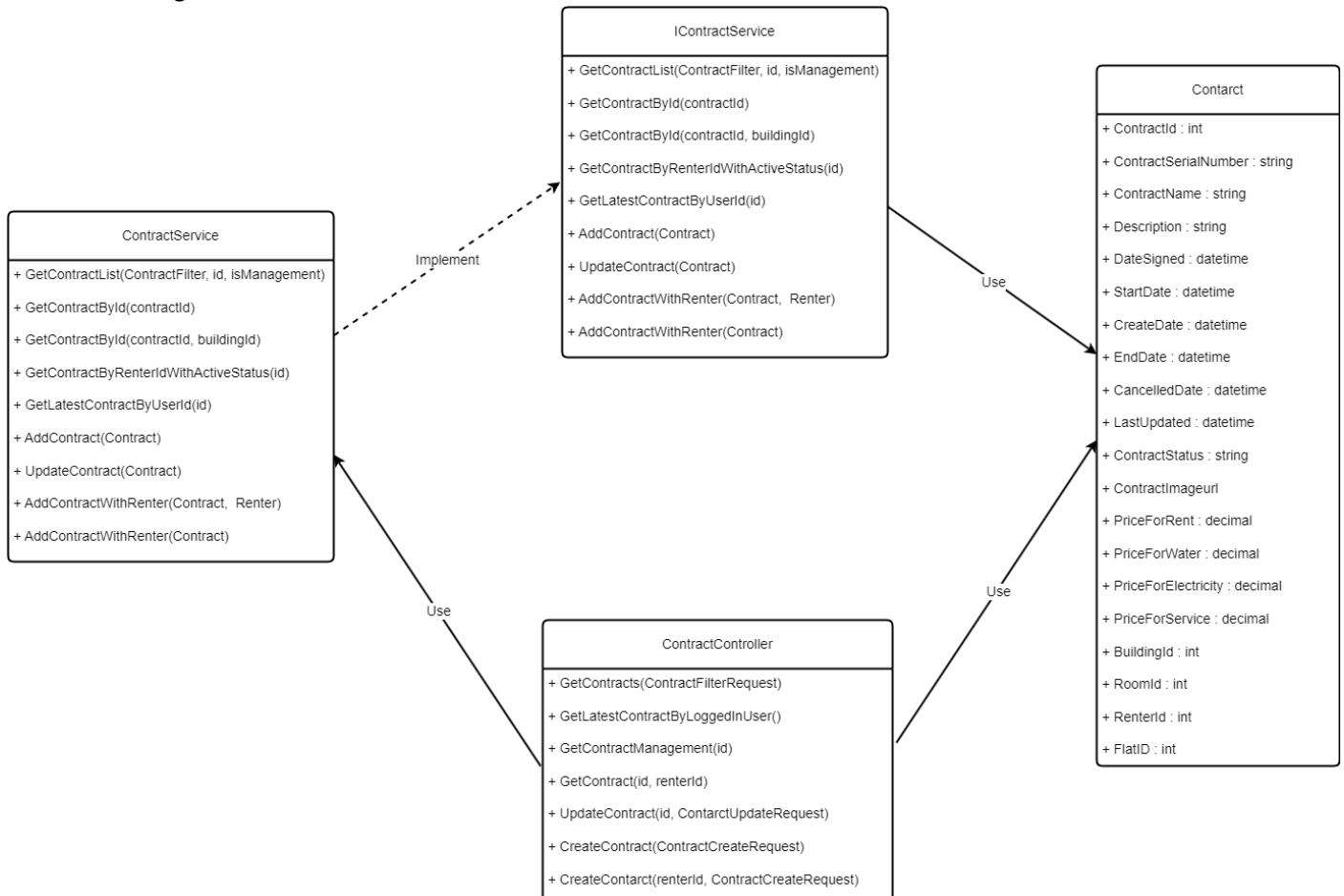


Figure 23 - Supervisor Renew Contract Class Diagram

### 3.5.2 Class Diagram Specification

No	Method	Description
1	GetContracts(FlatFilterRequest)	Get contract list
2	GetLatestContractByLoggedInUser()	Get latest contract by user
3	GetContractByManagement(id)	Get contract by management
4	UpdateContract(id, ContractUpdateRequest)	Update contract
5	GetContract(id, renterId)	Get contract by renter and its id
6	CreateContract(ContractCreateRequest)	Create contract with new renter
7	CreateContract(renterid, ContractCreateRequest)	Create contract with existing renter

### 3.5.3 Supervisor Renew Contract Activity Diagram

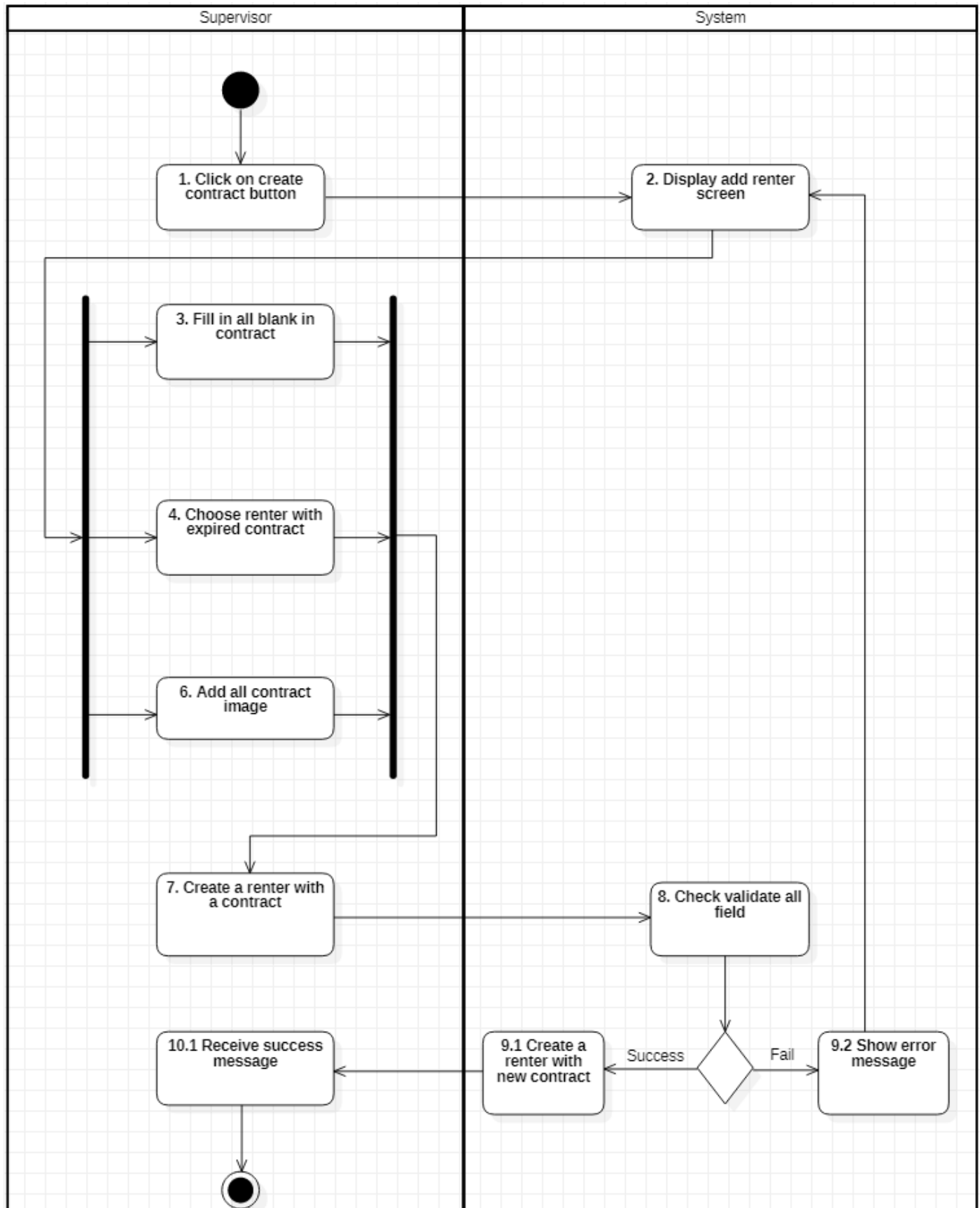


Figure 24 - Supervisor Renew Contract Activity Diagram

### 3.5.4 Sequence Renew Contract Activity Diagram

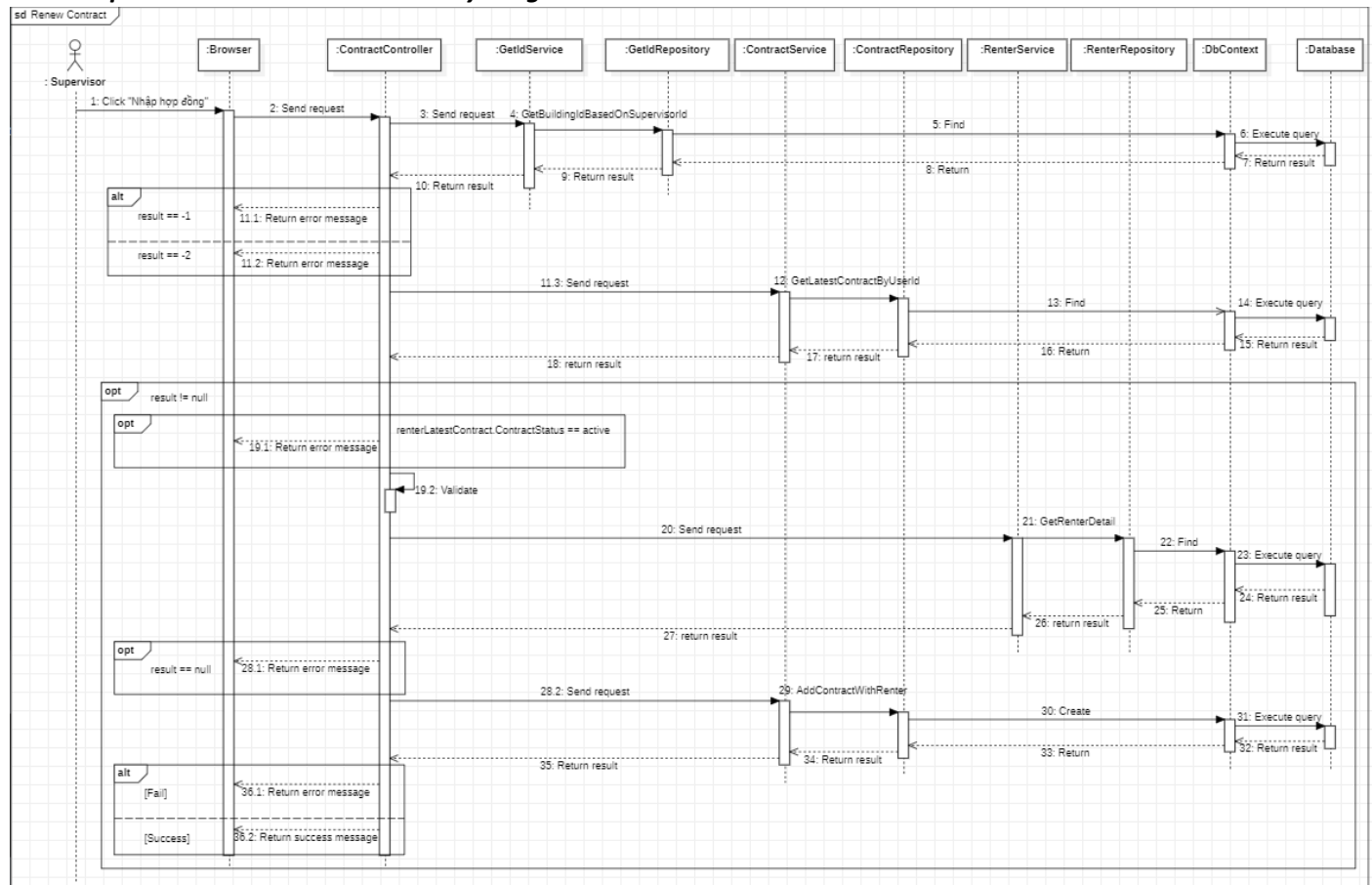


Figure 25 - Sequence Renew Contract Activity Diagram

### 3.6 Supervisor View Contract

#### 3.6.1 Class Diagram

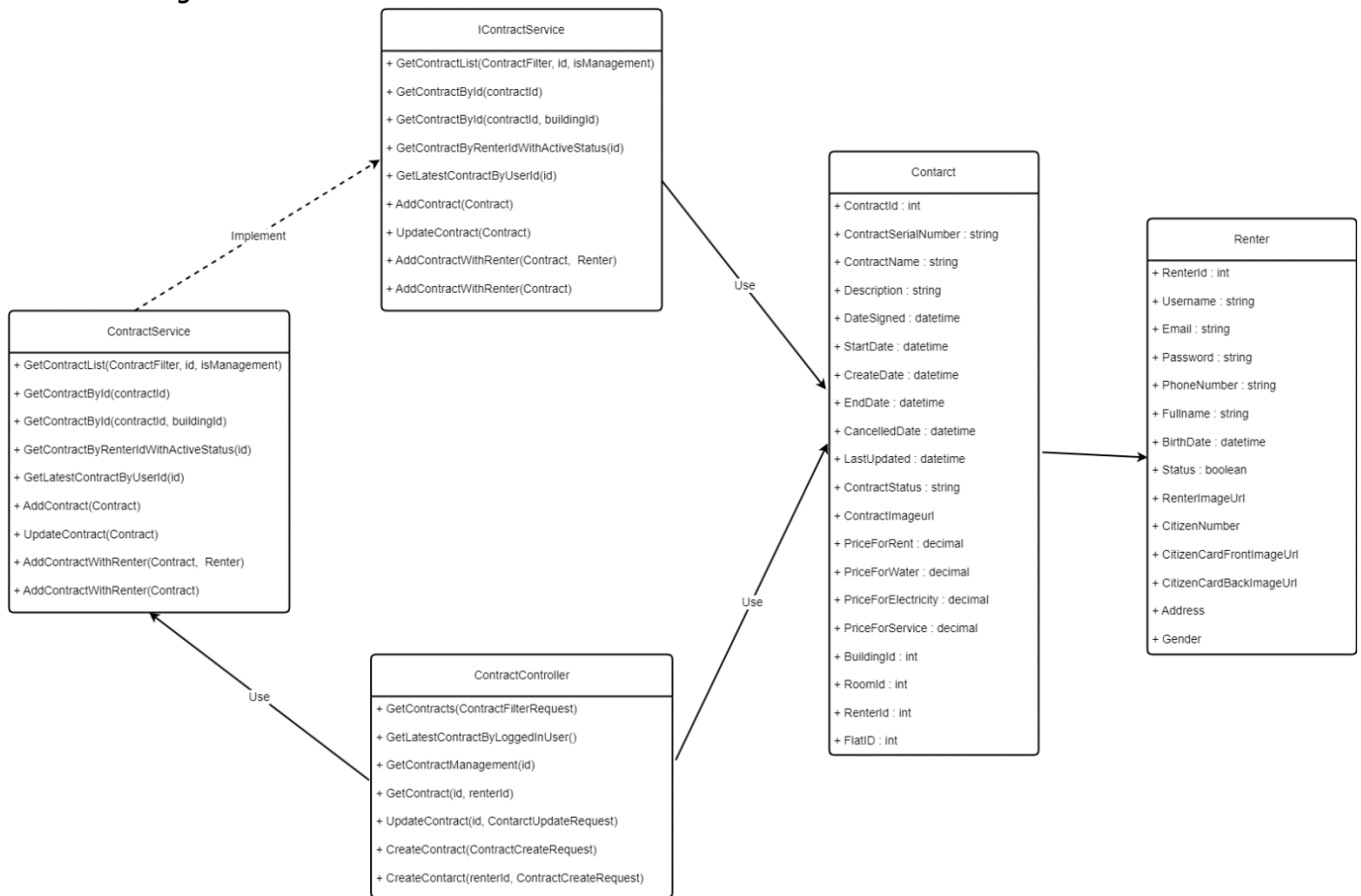


Figure 26 - Supervisor View Contract Class Diagram

#### 3.6.2 Class Diagram Specification

No	Method	Description
1	<code>GetContracts(FlatFilterRequest)</code>	Get contract list
2	<code>GetLatestContractByLoggedInUser()</code>	Get latest contract by user
3	<code>GetContractByManagement(id)</code>	Get contract by management
4	<code>UpdateContract(id, ContractUpdateRequest)</code>	Update contract
5	<code>GetContract(id, renterId)</code>	Get contract by renter and its id
6	<code>CreateContract(ContractCreateRequest)</code>	Create contract with new renter
7	<code>CreateContract(renterid, ContractCreateRequest)</code>	Create contract with existing renter

### 3.6.3 Supervisor View Contract Activity Diagram

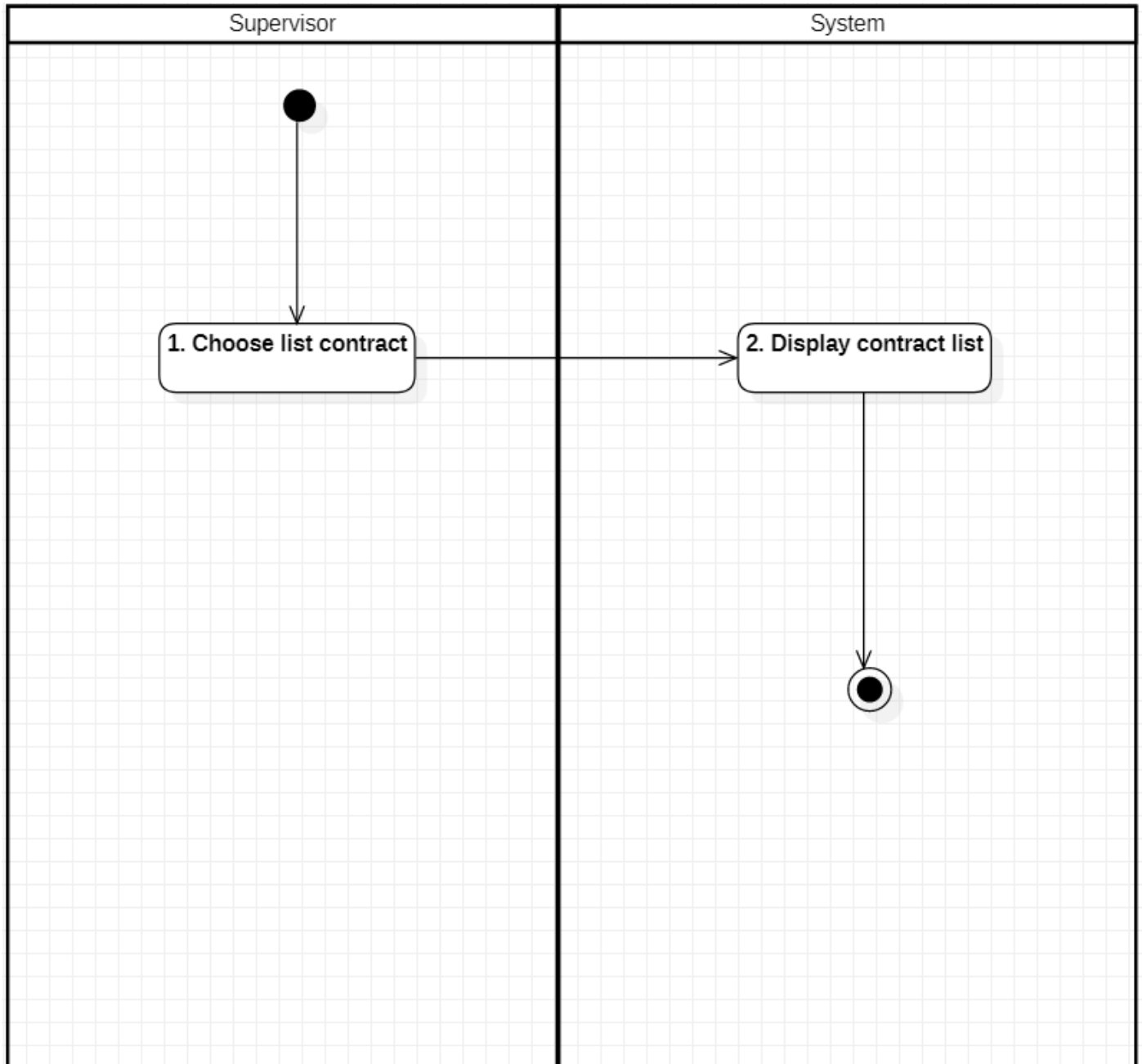


Figure 27 - Supervisor View Contract Activity Diagram



### 3.6.4 Supervisor View Contract Sequence Diagram

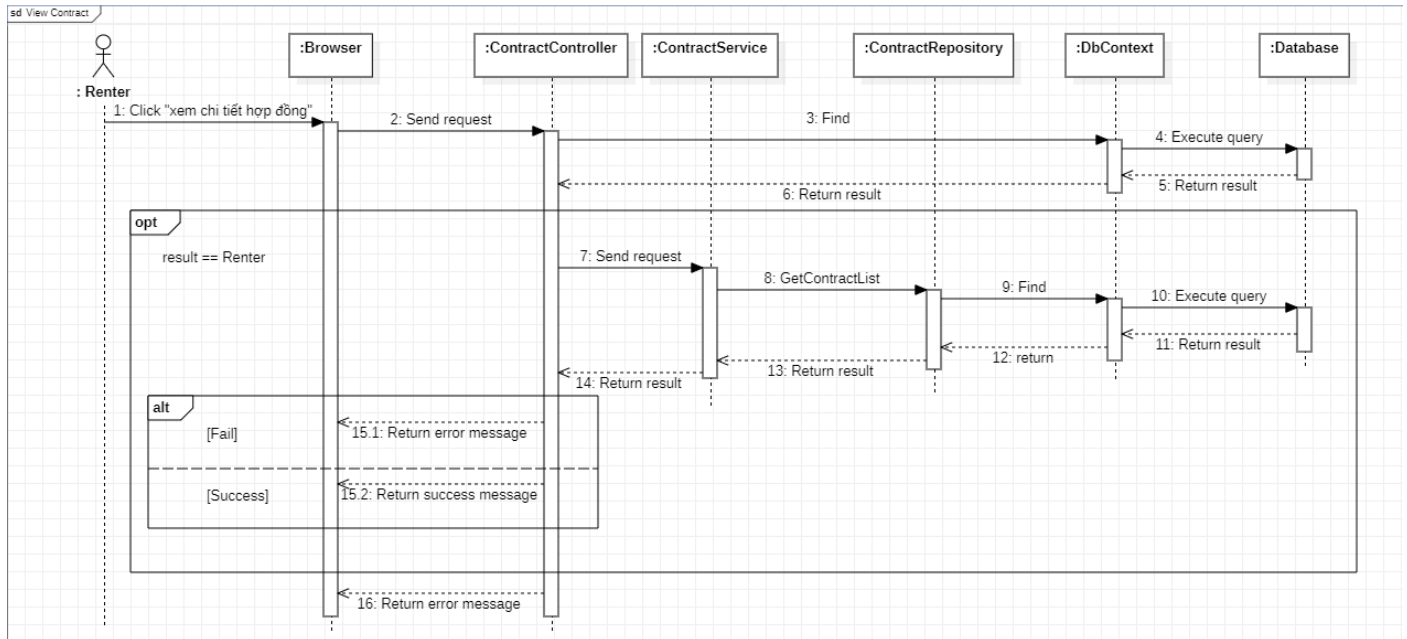


Figure 28 - Supervisor View Contract Sequence Diagram

## 3.7 Supervisor Manage Ticket

### 3.7.1 Class Diagram

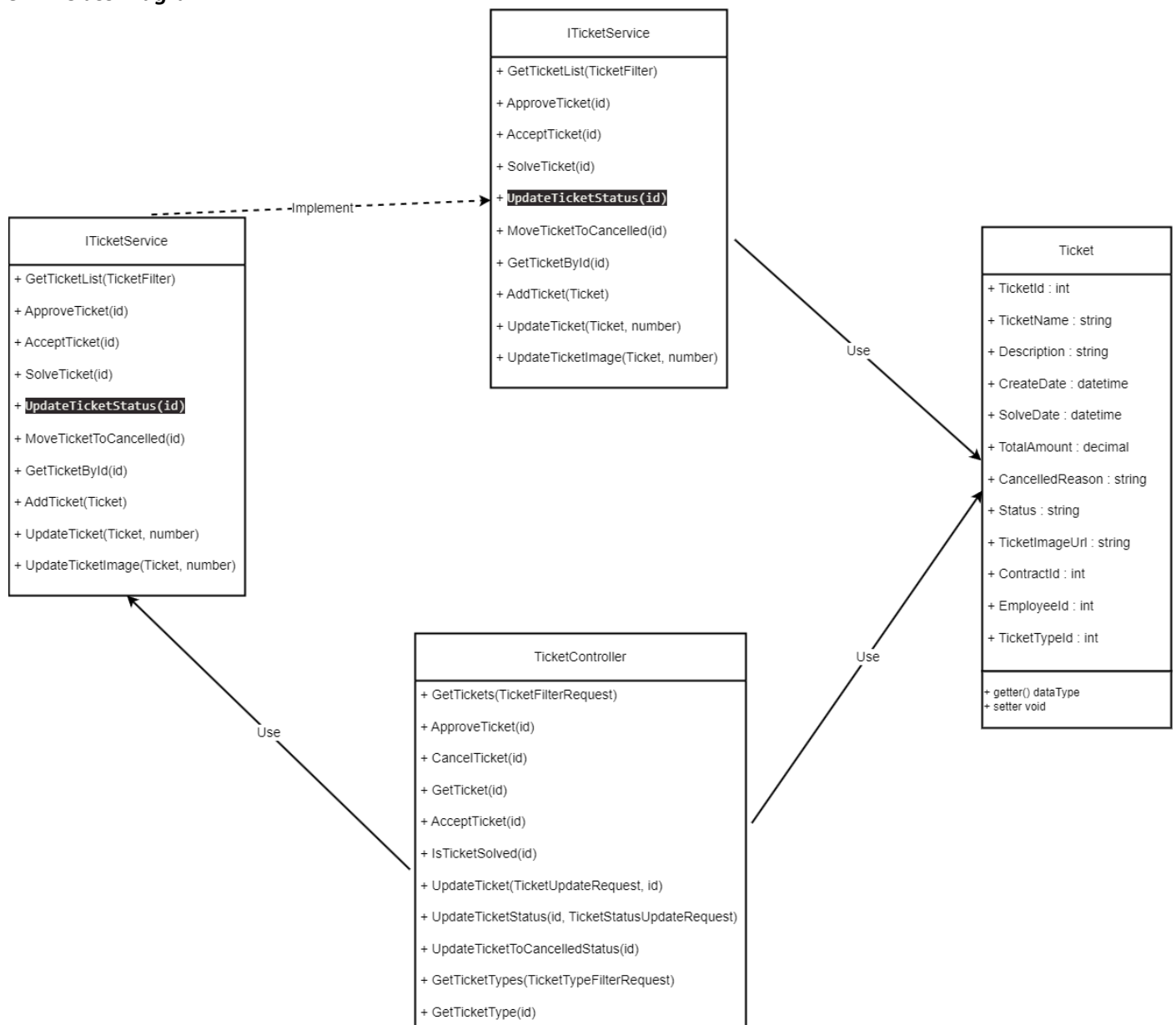


Figure 29 - Supervisor Manage Ticket Class Diagram

### 3.7.2 Class Diagram Specification

No	Method	Description
1	GetTickets(TicketFilterRequest)	Get ticket list
2	ApproveTicket(id)	Get latest contract by user
3	CancelTicket(id)	Cancel ticket by id
4	GetTicket(id)	Get ticket by id
5	AcceptTicket(id)	Accept ticket to solve
6	IsTicketSolved(id)	Change ticket status to confirming
7	UpdateTicket(id, TicketStatusUpdateRequest)	Update ticket status

8	UpdateTicketToCancelledStatus(id)	Renter choose to cancel theirs active ticket
9	GetTicketTypes(TicketTypeFilterRequest)	Get ticket types
10	GetTicketType(id)	Get ticket type by id

### 3.7.3 Supervisor Edit Ticket Activity Diagram

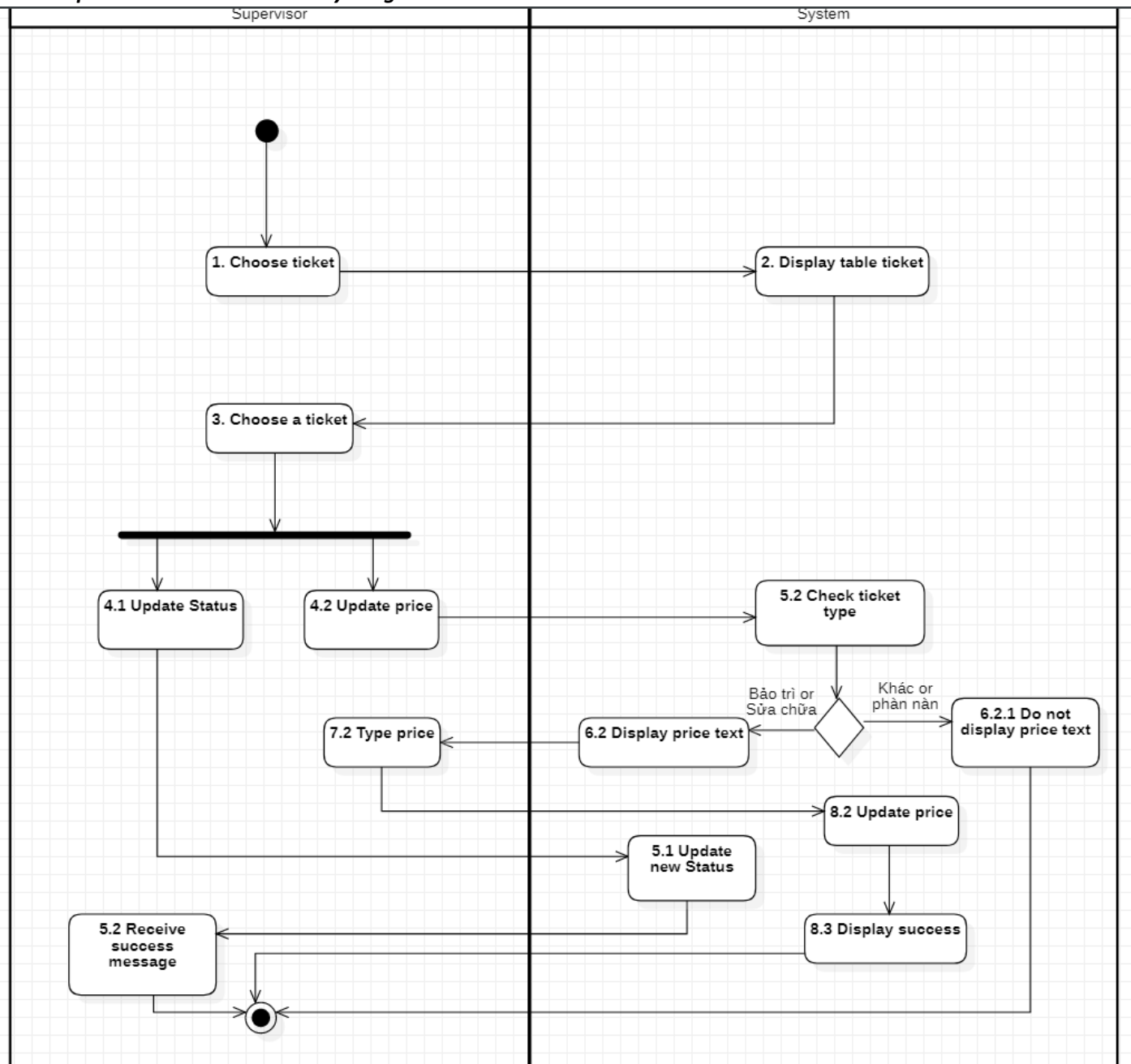


Figure 30 - Supervisor Edit Ticket Activity Diagram

### 3.7.4 Supervisor Edit Ticket Sequence Diagram

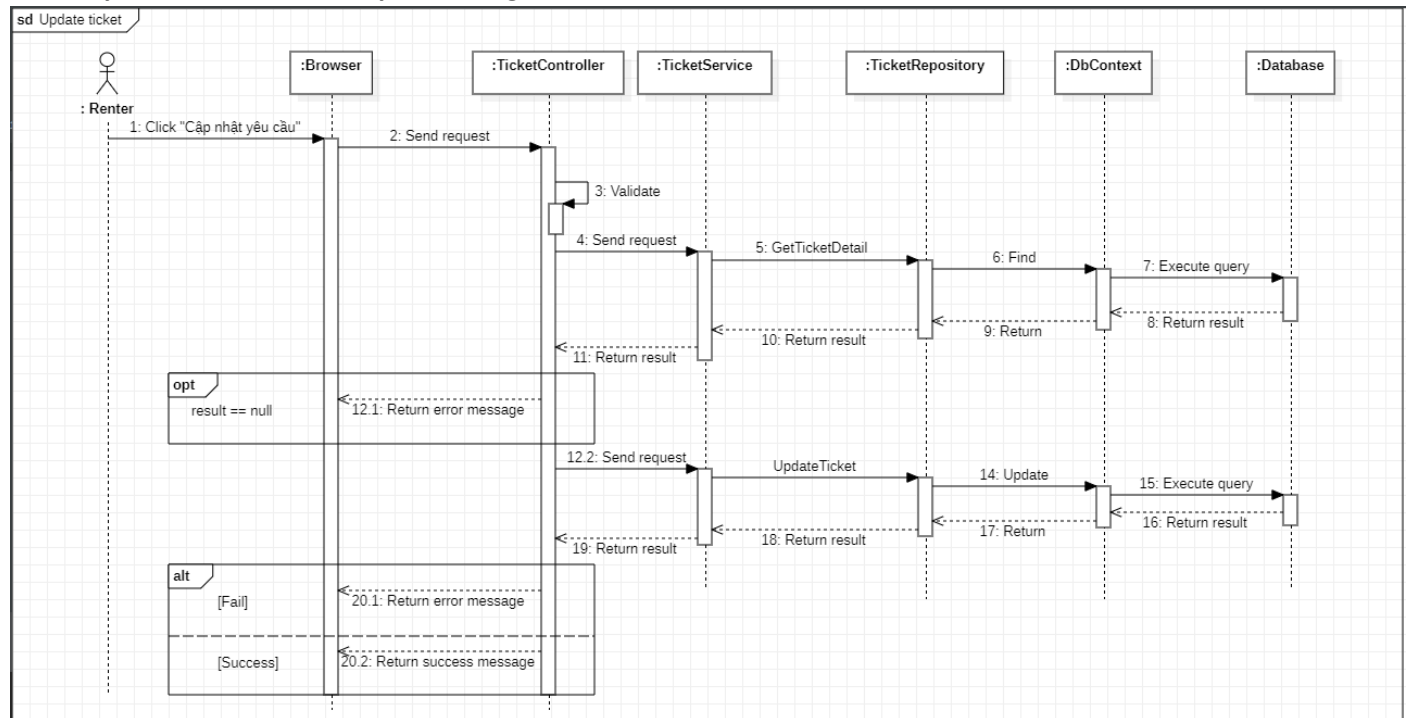
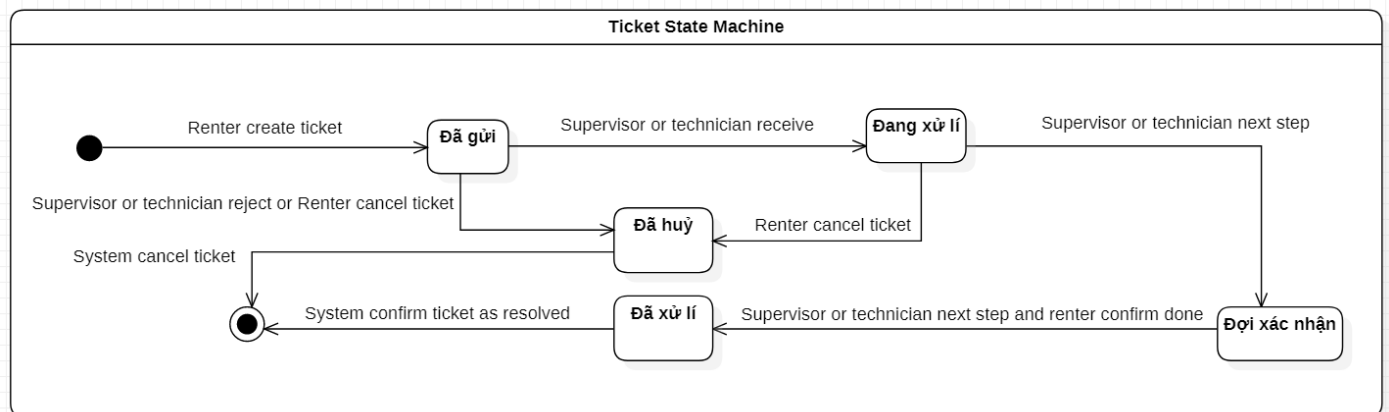


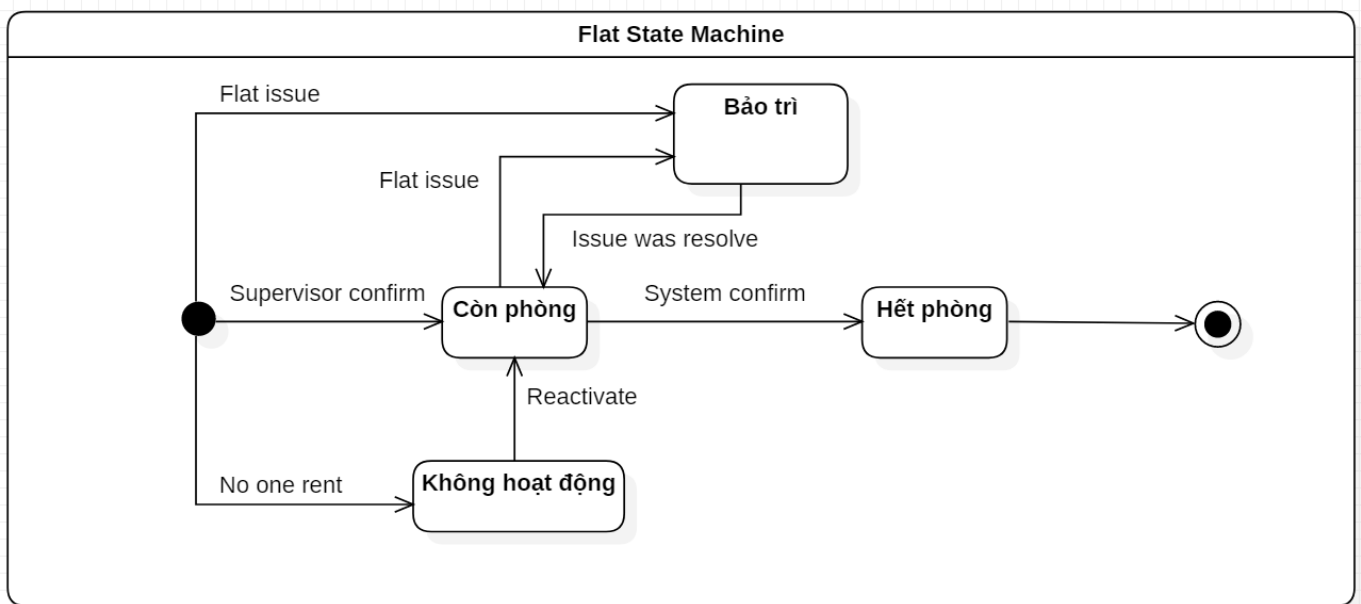
Figure 31 - Supervisor Edit Ticket Sequence Diagram

## 3.9 State Machine Diagram

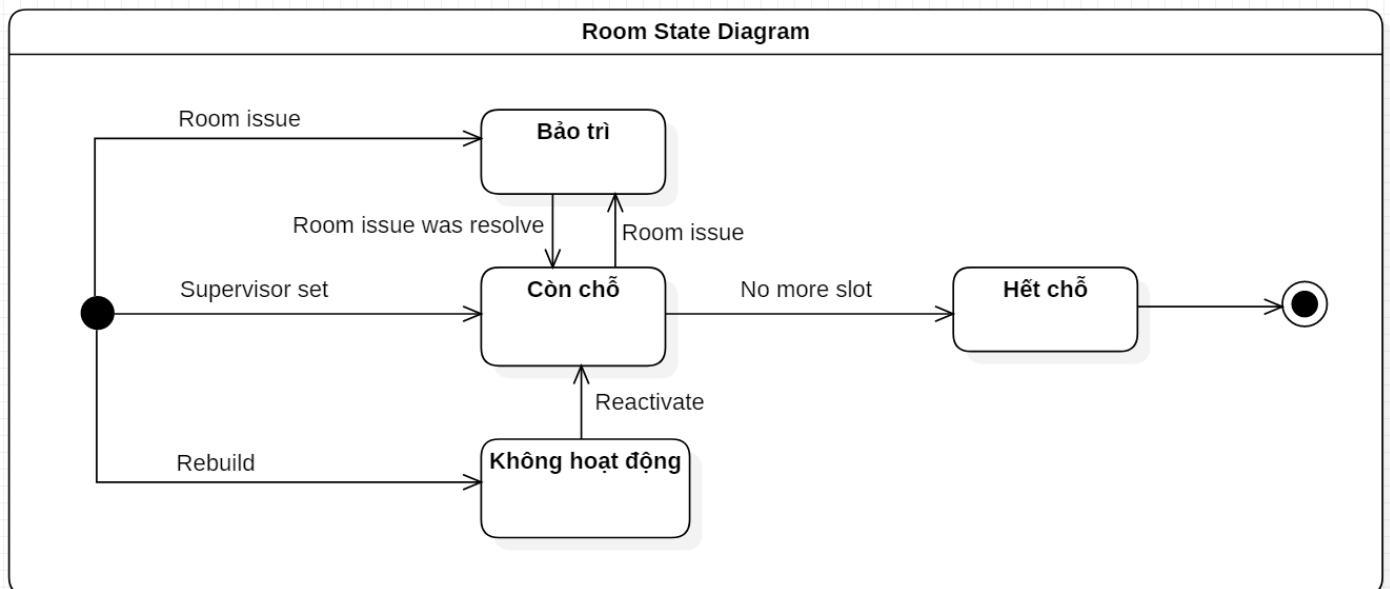
### 3.9.1 Ticket State Diagram



### 3.9.2 Flat State Diagram



### 3.9.3 State Room Diagram



## V. Software Testing Documentation

### 1. Scope of Testing

#### 1.1 Testing model

We chose Agile Testing strategy, as:

- Its adaptable nature to rapid changes.
- It helps developers engage deeper with the software's quality, performance, as well as user friendliness.
- It encourages frequent interaction between developers and end users.

It helps producing a working software earlier.

### 1.2 Testing level

- Unit testing
- Integration testing
- System testing
- Acceptance testing

## 2. Test Strategy

### 2.1 Testing Types

- Testing types: Function test, User Interface testing, user acceptant test
- Function testing
- User Interface testing

### 2.2 Test Levels

- Tesing levels: System testing, manual testing

Type of Tests	Test Level			
	Unit	Integration	System	Acceptance
Function test		X	X	X
User Interface test			X	

Test Levels

### 2.3 Supporting Tools

Purpose	Tool	Vendor/In-house	Version
Test Management	TestRail	TestRail	4.2

Supporting Tools

## 3. Test Plan

### 3.1 Human Resources

Worker/Doer	Role	Specific Responsibilities/Comments
Dao Bao Tram	Leader	Planning, verifying test deliverables, do interface testing as planned
Pham Minh Duc	Member	Planning, verifying test deliverables, do interface testing as planned

Human Resources

### 3.2 Test Environment

Purpose	Tool	Provider	Version
Browser Testing	Google Chrome	Google Chrome	108.0.5359.73
Browser Testing	Internet Explorer	Internet Explorer	107.0.1418.62
Mobile Testing			

Test Enviroment

### 3.3 Test Milestones

Milestone Task	Start Date	End Date
Create test plan	1/3/2023	3/3/2023
Create integration test case for Web App	3/3/2023	27/3/2023
Create integration test case for Mobile App	3/3/2023	27/3/2023
Create system test case for Web App	1/4/2023	10//2023
Create system test case for Mobile App	1/4/2023	10//2023
Create report for testing	12/4/2022	20/4/2022

*Test Milestones*

### 4. Test Cases

- Test Cases (IT, ST, AT): [SP23SE35\\_Test\\_Report\\_v1.0](#)

### 5. Test Reports

**SP23SE35\_Test\_Report\_v1.0** .xlsx

Tệp Chỉnh sửa Xem Chèn Định dạng Dữ liệu Công cụ Trợ giúp

Find & Replace | Undo | Redo | Print | Copy | Paste | Bold | Italic | Underline | Font Color | Background Color | Paragraph | Styles | Spelling | Grammar | Comments | Track Changes | View | Zoom | Window | Help

	A	B	C	D	E	F	G	H
J1	f_x							
1								
2	<b>Module Code</b>		Manage Building					
3	<b>Test requirement</b>		Manage Building: Create, update					
4	<b>Reference Document</b>							
5	<b>Pass</b>	<b>Fail</b>	<b>Untested</b>	<b>N/A</b>	<b>Number of Test cases</b>			
6	7	0	0	0	7			
7								
8	ID	Test Case Description	Test Case Procedure	Expected Results	Inter-test case Dependence	Result	Test date	Tester
9	Create building by Supervisor							
10	[Manage Building -1]	Create a building when the manager has not managed any buldings yet	Only when the manager has not managed any buildings yet 1. Login into Admin Role 2. Click "Tòa nhà KTX" tab in the sidebar 3. Fill all the form "Toà nhà KTX" with all valid data 4. Click "Thêm ký túc xá" button	The system will display message: "Thêm ký túc xá thành công"	After that, the screen rebads and goes to the update page, the create page is disabled	Pass	5/15	Phạm Minh Đức
11	[Manage Building -2]	Validation don't input any data for "Tên ký túc xá"	Only when the manager has not managed any buildings yet 1. Login into Admin Role 2. Click "Tòa nhà KTX" tab in the sidebar 3. Fill all the form "Toà nhà KTX" with "Tên ký túc xá"	The system displays message: "Tên đang trống"		Pass	5/15	Phạm Minh Đức

Project Name	VinFlat	Creator	Đào Bảo Trâm
Project Code	SP23SE35	Reviewer / Approver	Phạm Minh Đức
Document Code	SP23SE35_Test Report_v1.0	Issue Date	05/15/2023
Notes			



## VI. Release Package & User Guides

### 1. Deliverable Package

No	Deliverable Equipment	Description
01	Project Schedule/Tracking	Project progress tracking
02	Project Backlog	It is often a complete list that breaks down work that needs to be completed.
03	Source Codes	VinFlat_Backend.zip VinFlat_Frontend.zip VinFlat_Mobile.zip
04	Database Script(s)	Script_VinFlat
05	Final Report Document	VinFlat_FinalProjectReport.docx
06	Test Cases Document	SP23SE35_Test_Report.xlsx

### 2. Installation Guides

#### 2.1 System Requirements

##### 2.1.1 Hardware requirements

##### 2.1.1.1 Web Application

PC	Minimum	Recommended
Internet connection	Cable, Wi-fi (4 Mbps)	Cable, Wi-fi (8 Mbps)
Processor	Intel Core i3 1.4Ghz	Intel Core i7 2.5Ghz
Memory	4GB RAM	8GB RAM
Storage	HDD 100GB	SSD 200GB
Web Browser	Chromes (v69) Microsoft Edge (v109)	Chromes latest stable version Microsoft Edge latest stable version

##### 2.1.1.2 Mobile Application

<b>Operating System</b>	Android 20 or higher
<b>Processor</b>	Intel(R) Atom(TM) CPU Z3580 @ 1.33GHz, or faster processor
<b>Storage</b>	Minimum 256 MB
<b>RAM</b>	Minimum of 1GB, 2GB is recommend

### 2.1.2 Software requirements

Component	Name and Version	Description
Operating System	Windows 7 SP1/8.1/10/Window Server 2016 or above (Web application) Android 10 AndroidQ or above (Mobile application)	Operating system for building production
DBMS		Used to manage database
.Net	.NET SDK 6.0, .NET Core 6.0 Runtime	Used to run backend server
ReactJS(Javascript)	7.0.2	Used to run frontend application
Flutter	>3.0.2	Used to run mobile application
IDE	Visual Studio, Visual Studio Code, Android Studio	Used to edit and run code

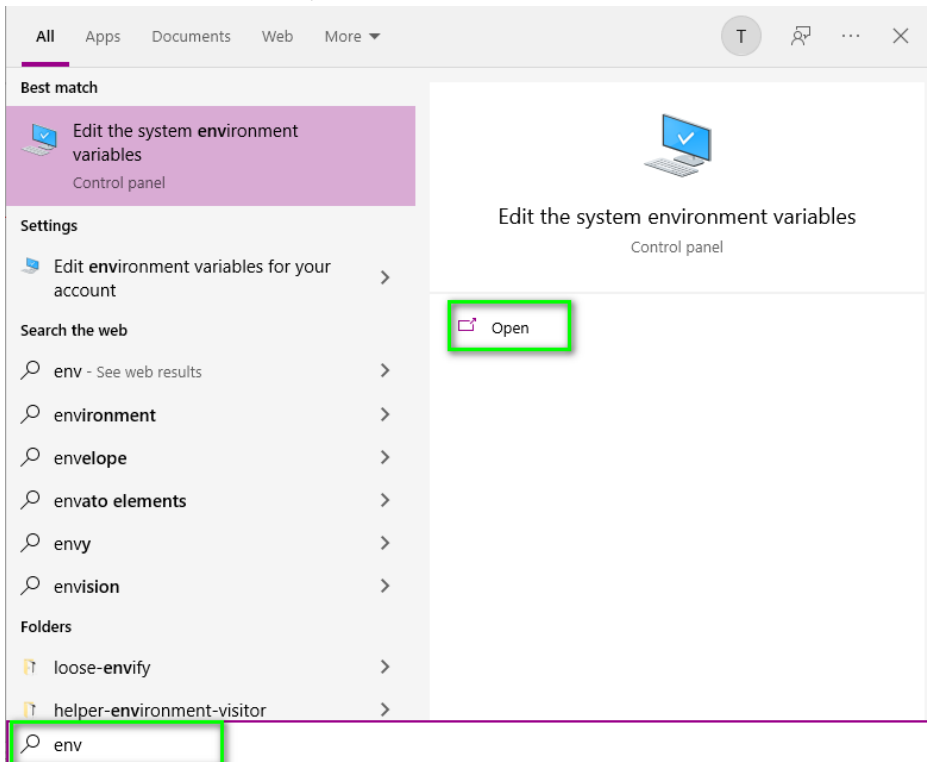
### 2.3 Installation Instruction

Setup and start a Windows OS (Windows 10 or Windows Server 2016):

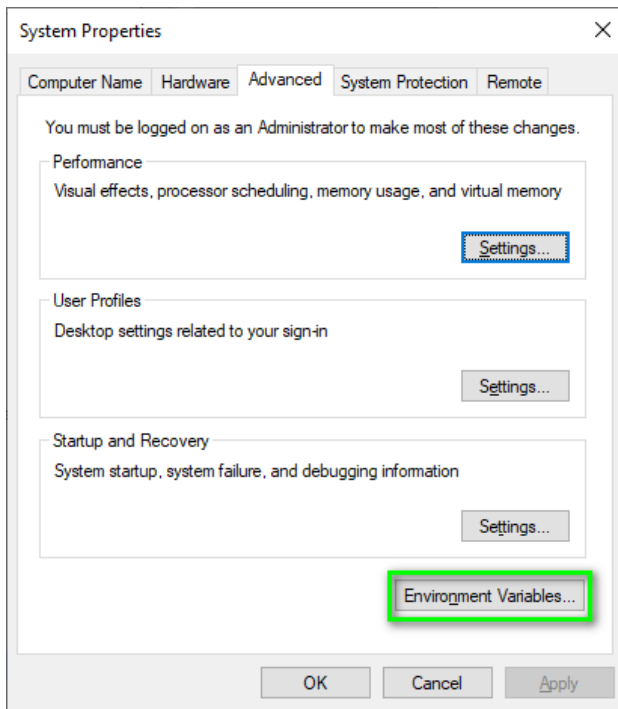
## 2.3.1 Setup Environments

### 2.3.1.1 Flutter

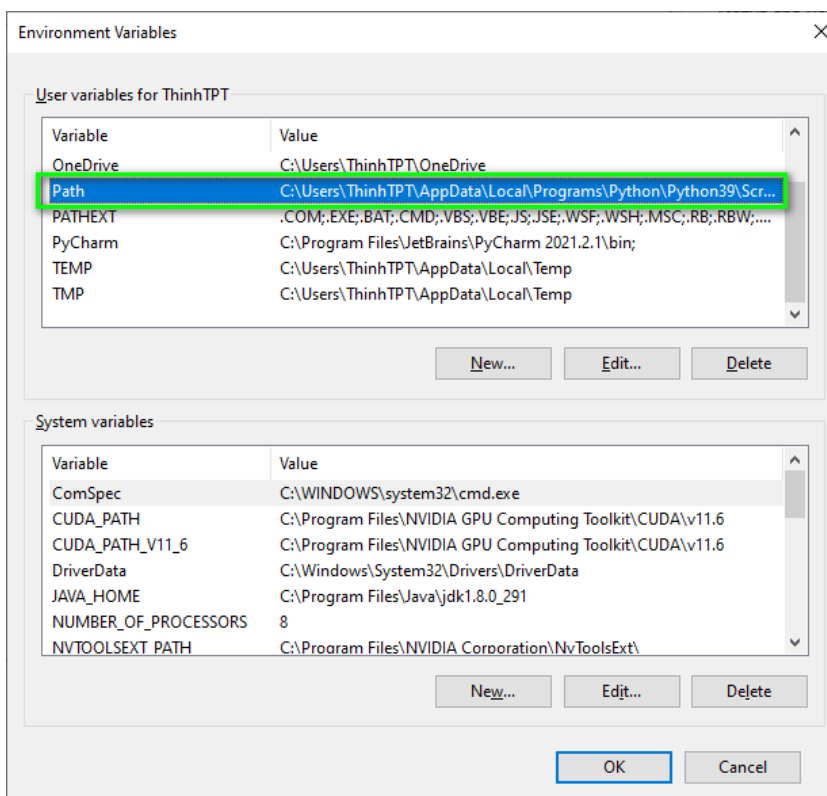
1. Download the following installation bundle to get the latest stable release of the Flutter SDK from [here](#).
2. Extract the zip file and place the contained flutter in the desired installation location for the Flutter SDK (for example, C:\src\flutter)
3. Update the environment path:
  - 3.1. From the Start search bar, enter 'env' and select **Edit environment variables for your account**.



- 3.2. Click **Environment Variable** button.



3.3. Under User variables check if there is an entry called **Path**

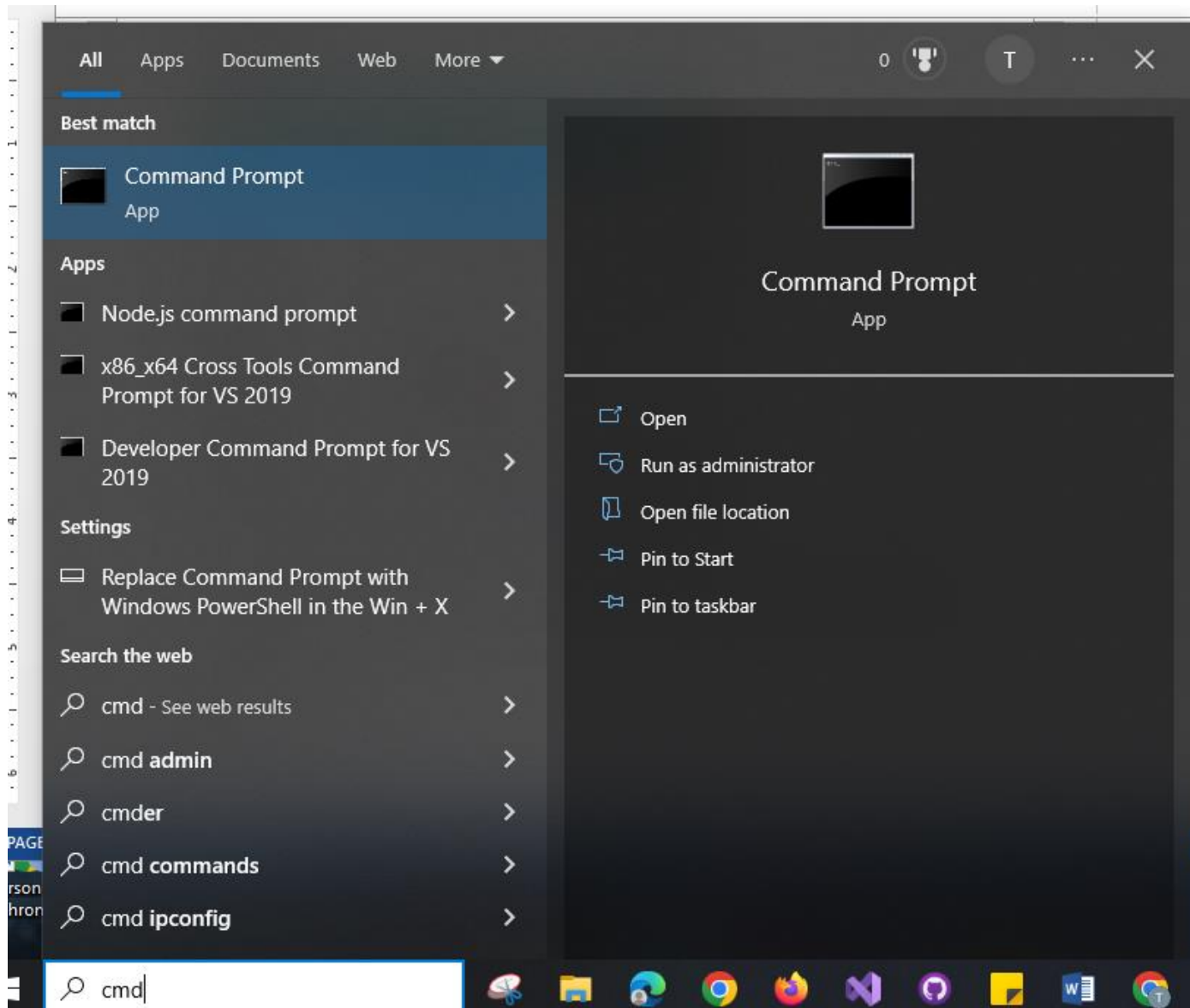


3.3.1.If the entry exists, append the full path to flutter\bin using ";" as a separator from existing values.

3.3.2.If the entry does not exist, create a new user variable named Path with the full path to flutter\bin as its value

4. Run command line to check

#### 4.1. Open command line



#### 4.2. Run “where flutter dart” and “flutter doctor”

```

C:\Users\DELL>where flutter dart
E:\otherProgram\Library\Flutter\flutter_win_3.3.10\flutter\bin\flutter
E:\otherProgram\Library\Flutter\flutter_win_3.3.10\flutter\bin\flutter.bat
E:\otherProgram\Library\Flutter\flutter_win_3.3.10\flutter\bin\dart
E:\otherProgram\Library\Flutter\flutter_win_3.3.10\flutter\bin\dart.bat

C:\Users\DELL>flutter doctor

A new version of Flutter is available!
To update to the latest version, run "flutter upgrade".

Doctor summary (to see all details, run flutter doctor -v):
[✓] Flutter (Channel stable, 3.3.10, on Microsoft Windows [Version 10.0.19044.2728], locale en-US)
[✓] Android toolchain - develop for Android devices (Android SDK version 32.0.0)
[✓] Chrome - develop for the web
[!] Visual Studio - develop for Windows (Visual Studio Community 2022 17.4.2)
    X Visual Studio is missing necessary components. Please re-run the Visual Studio installer for the "Desktop
      development with C++" workload, and include these components:
        MSVC v142 - VS 2019 C++ x64/x86 build tools
          - If there are multiple build tool versions available, install the latest
        C++ CMake tools for Windows
        Windows 10 SDK
[✓] Android Studio (version 2021.1)
[✓] Connected device (3 available)
[✓] HTTP Host Availability

! Doctor found issues in 1 category.
C:\Users\DELL>

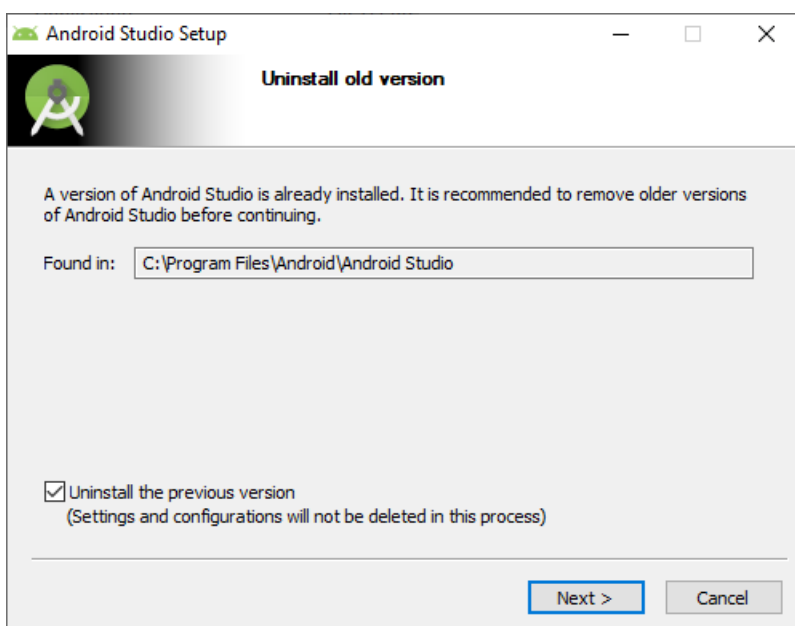
```

4.2.1.If “where flutter dart” return the location of 4 files like the first zone and “flutter doctor” return good flutter status like the second zone, the flutter enviroment is installed correctly. If not, check and redo the missing steps of installing flutter

4.2.2.If “flutter doctor” show both Visual Studio and Android Studio are missing, proceed to download either Visual Studio or Android Studio (recommended).

### 2.3.1.2 Android Studio

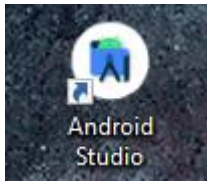
1. Download and install [Android Studio setup file](#).
2. Start Android Studio and go through the “Android Studio Setup Wizard”. This installs the latest Android SDK, Android SDK Command-line Tools, and Android SDK Build-Tools, which are required by Flutter when developing for Android.



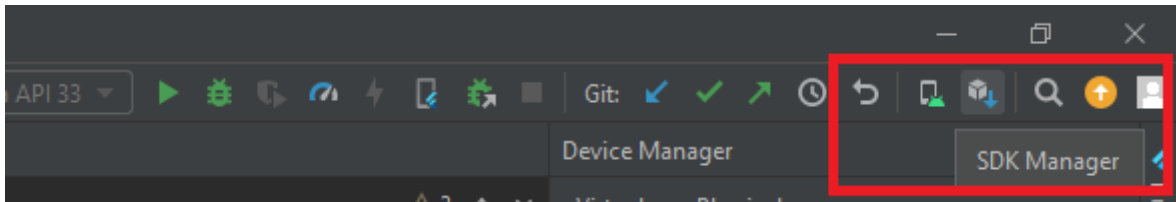
For more detail, see [here](#).

### 3. Download Android SDK

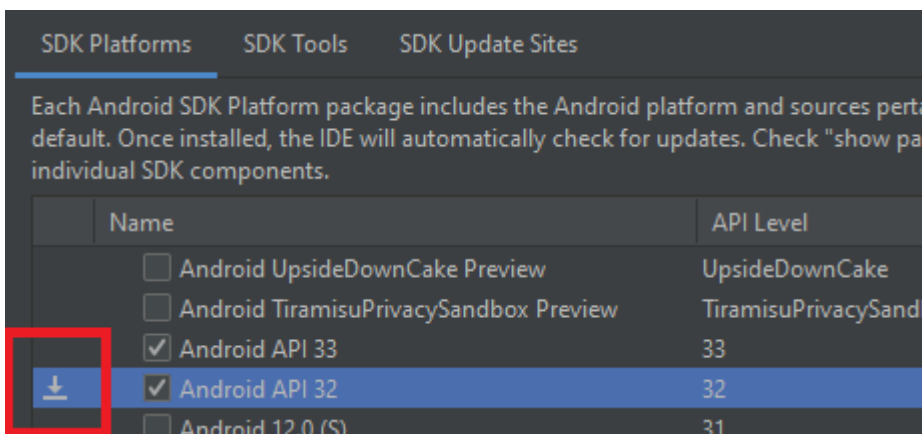
#### 3.1. Open Android Studio



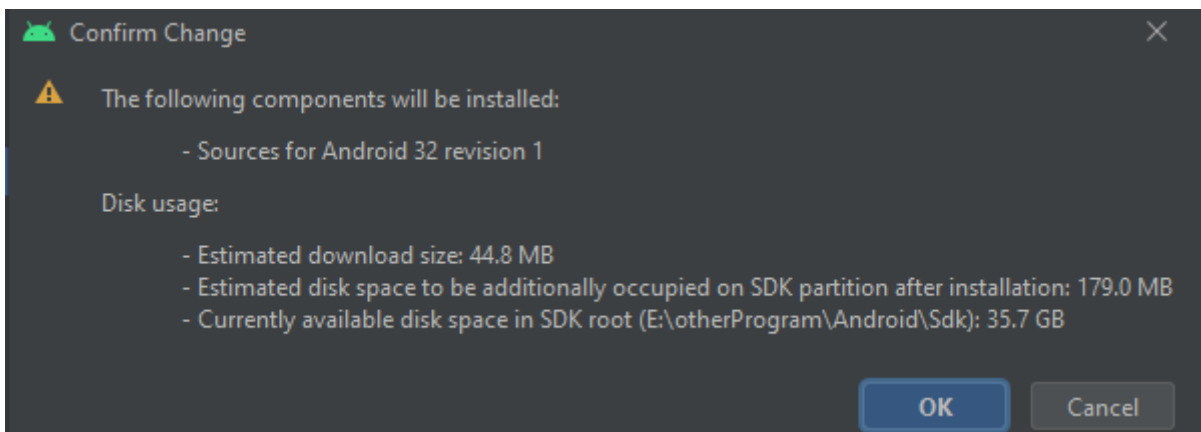
#### 3.2. Open window



#### 3.3. Choose SDK version (minimum 20) and click the download button

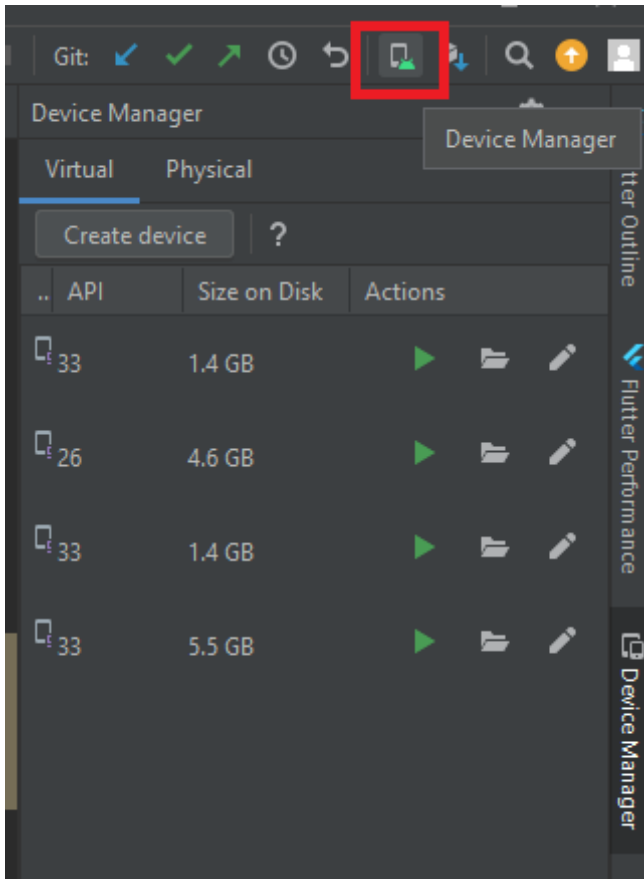


#### 3.4. Click Ok and wait for the download to complete

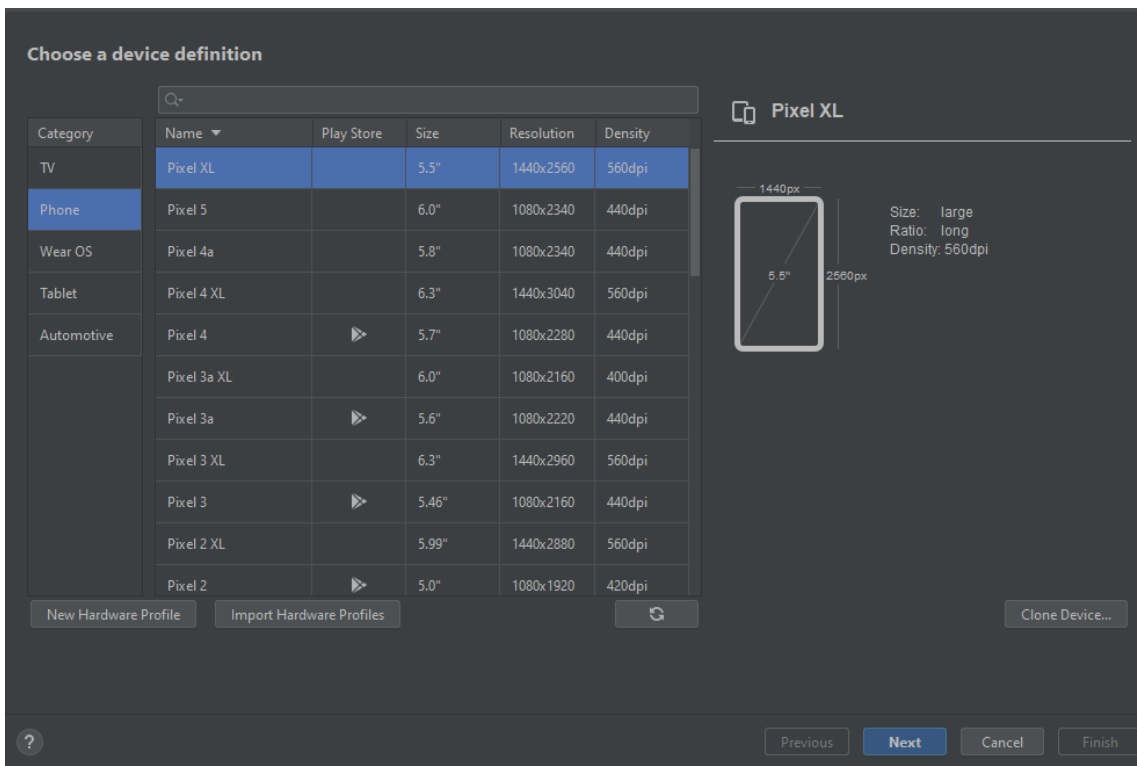


### 4. Download Virtual Device

#### 4.1. Open Device Manager and click Create New Device

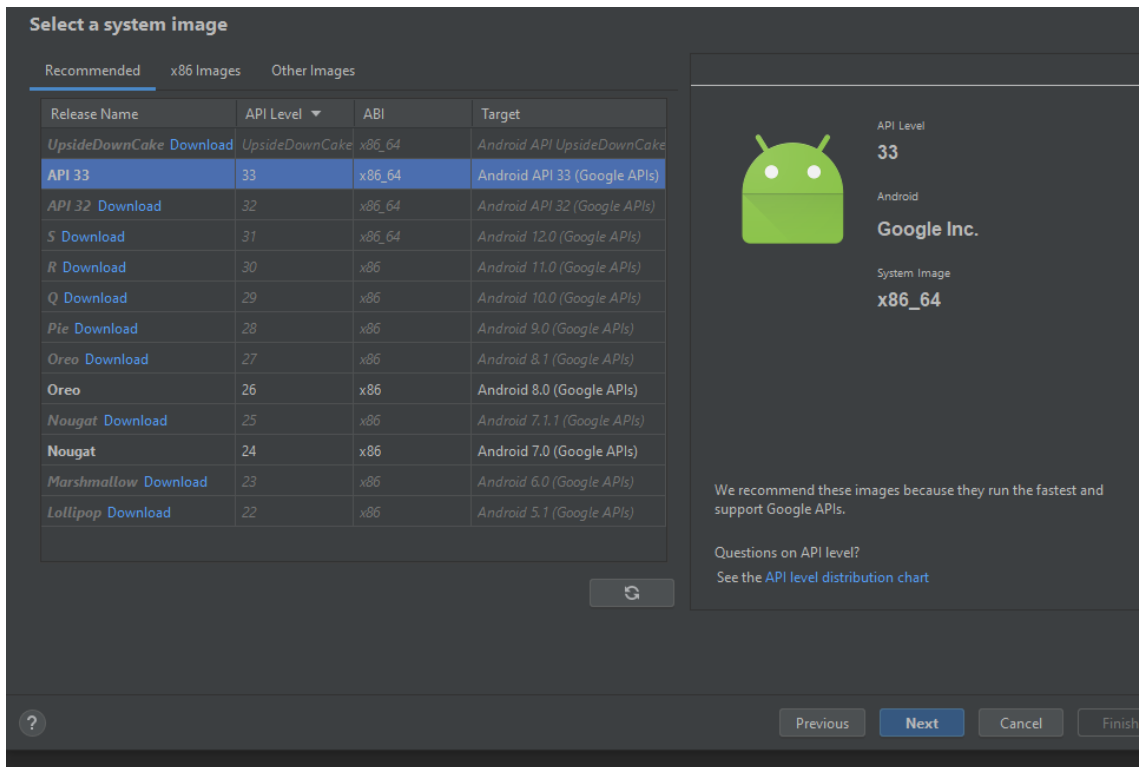


4.2. Choose Phone Device and click next



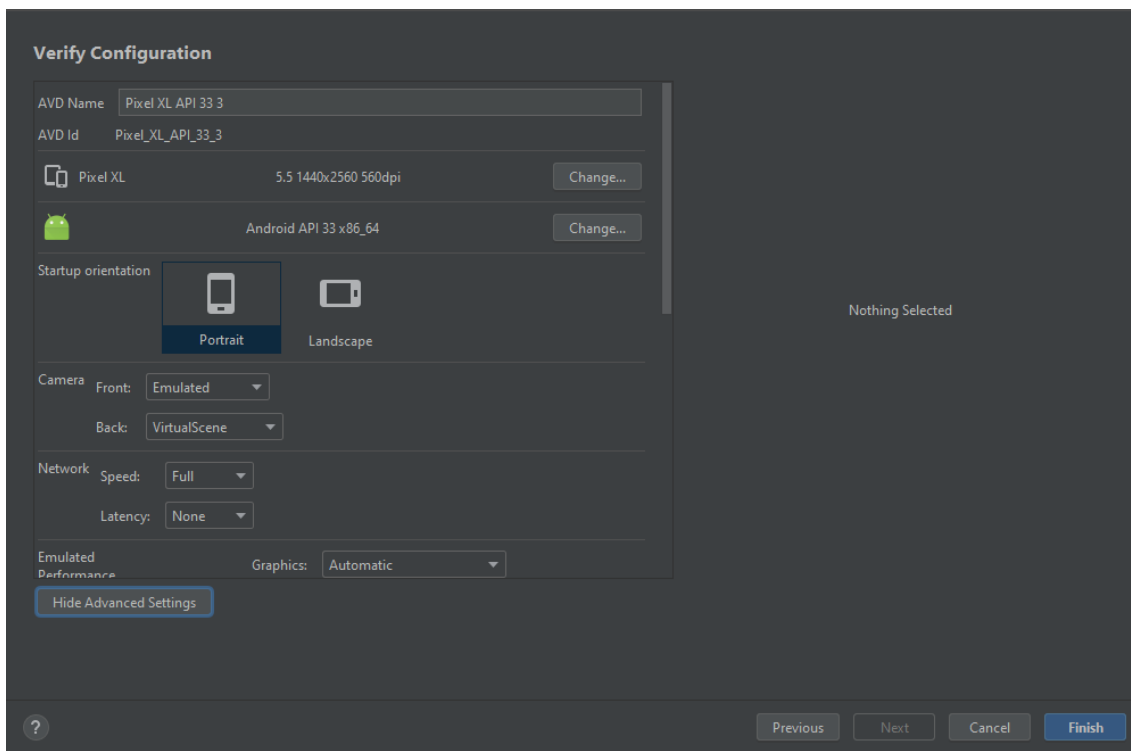
4.3. Choose a system image and click next





4.3.1.If there is no available system image, click download and wait till the download is finnnished

4.4. Custom some configuration if needed and click Finnish



b. Setup a Windows Server 2016

### 2.3.1.2 Backend

#### 2.3.1.2.1 Setup Database

Install Sql Server 2019 by following guidelines link:

<https://www.thegioioidong.com/game-app/huong-dan-cai-dat-sql-server-2019-cuc-don-gian-chi-1312926>

#### 2.3.2.2.2 Setup Backend API

1. Install Visual Studio by following guidelines link:  
<https://thuthuatphanmem.vn/huong-dan-tai-va-cai-dat-visual-studio/>
2. Extracting: VinFlat\_Backend.zip

Name	Date modified	Type	Size
.idea	5/16/2023 10:08 PM	File folder	
API	5/16/2023 10:08 PM	File folder	
Application	5/16/2023 10:08 PM	File folder	
Domain	5/16/2023 10:08 PM	File folder	
Infrastructure	5/16/2023 10:08 PM	File folder	
Service	5/16/2023 10:08 PM	File folder	
Utilities	5/16/2023 10:08 PM	File folder	
VinFlatSchedule	5/16/2023 10:08 PM	File folder	
.gitignore	10/12/2022 1:30 AM	Text Document	19 KB
ManagementBackEndTestBranch	11/23/2022 8:54 PM	SLN File	4 KB
query_drop_table_with_dbo	11/10/2022 5:57 AM	Microsoft SQL Ser...	3 KB
README	10/8/2022 2:21 PM	MD File	1 KB
script	10/16/2022 3:52 PM	Microsoft SQL Ser...	24 KB

Figure 61 - Setup Backend Api

3. Open ManagementBackend.sln file with Visual Studio
4. Config your database connection

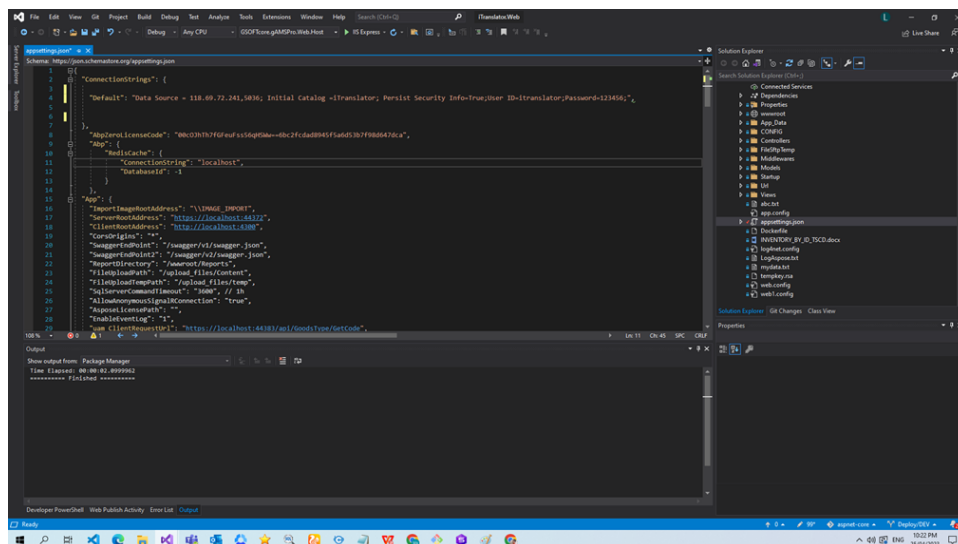


Figure 62 - Setup Backend Api

5. Run project

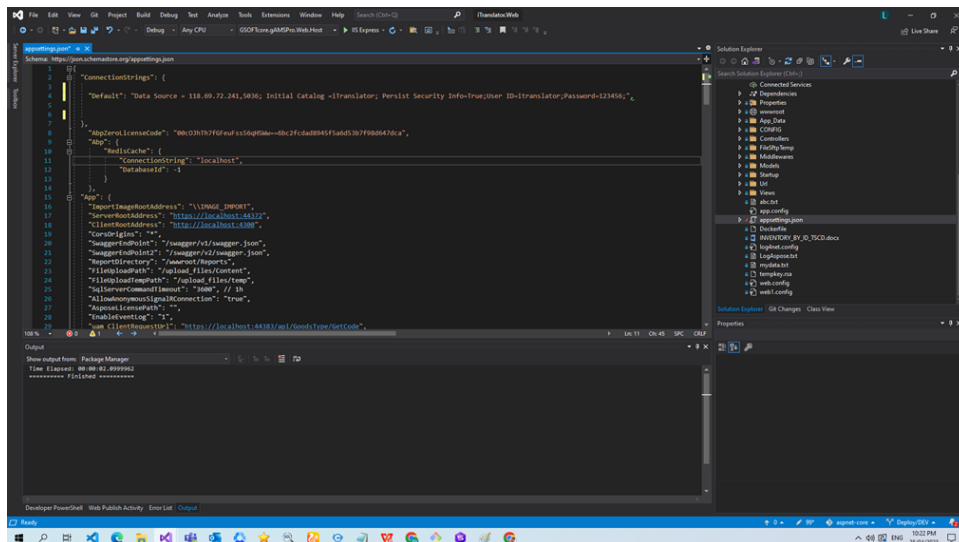
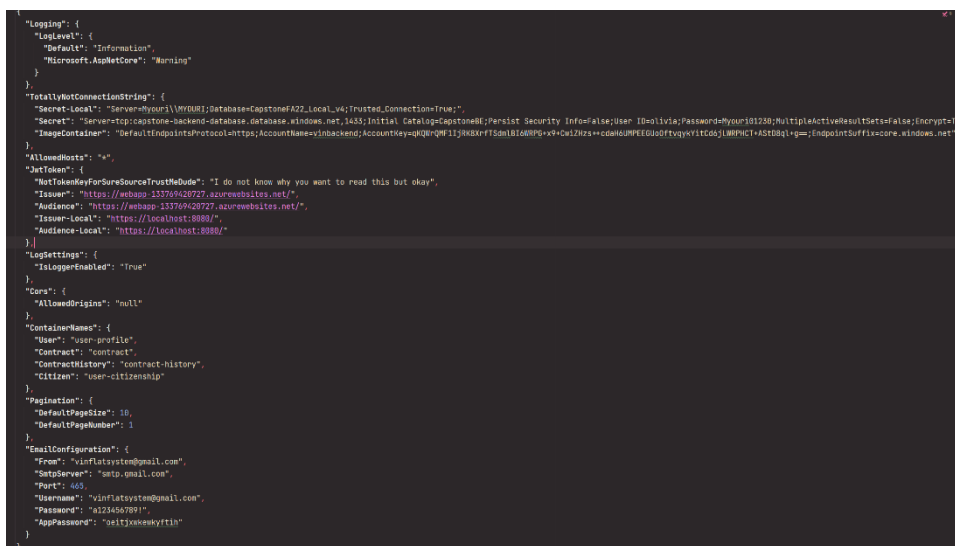


Figure 63 - Setup Backend Api



6. Test project in browser by go to url: <https://localhost:8080>

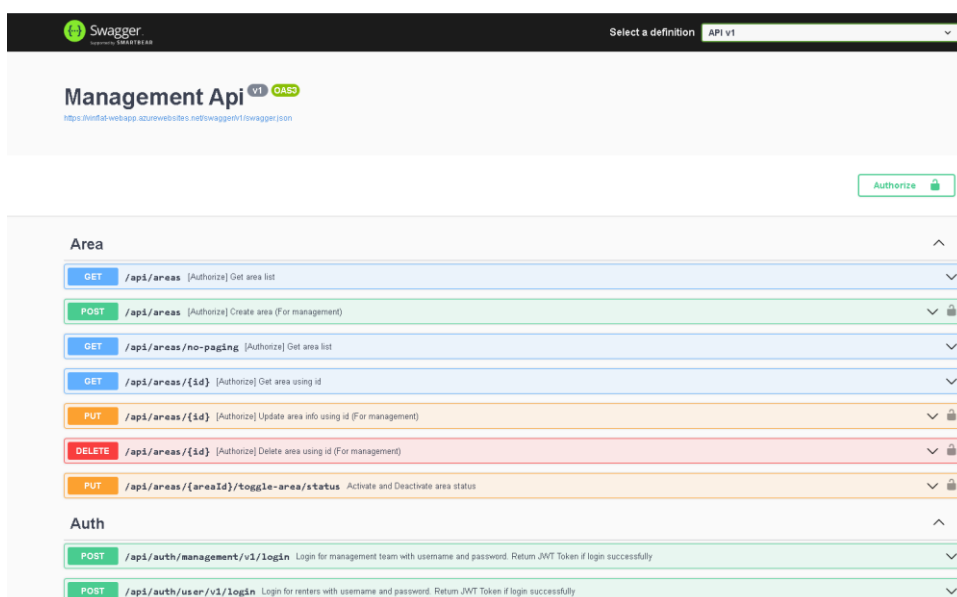


Figure 65 - Setup Backend Api

### 2.3.1.3 Frontend

#### 2.3.1.3.1 Setup Management Web

1. Install Visual Studio Code by following guidelines link:  
<https://thaynhuom.edu.vn/cai-visual-studio-code-lap-trinh-c-c/>
2. Install npm through Visual Studio Code or you can install npm via NodeJS <https://nodejs.org/en/>
3. Extract VinFlat\_Frontend.zip and open folder with Visual Studio Code
4. Open terminal and enter npm i:



```
i\project\Front-end> npm install
```

Figure 66 - Setup Management Web

5. Read more about tutorial

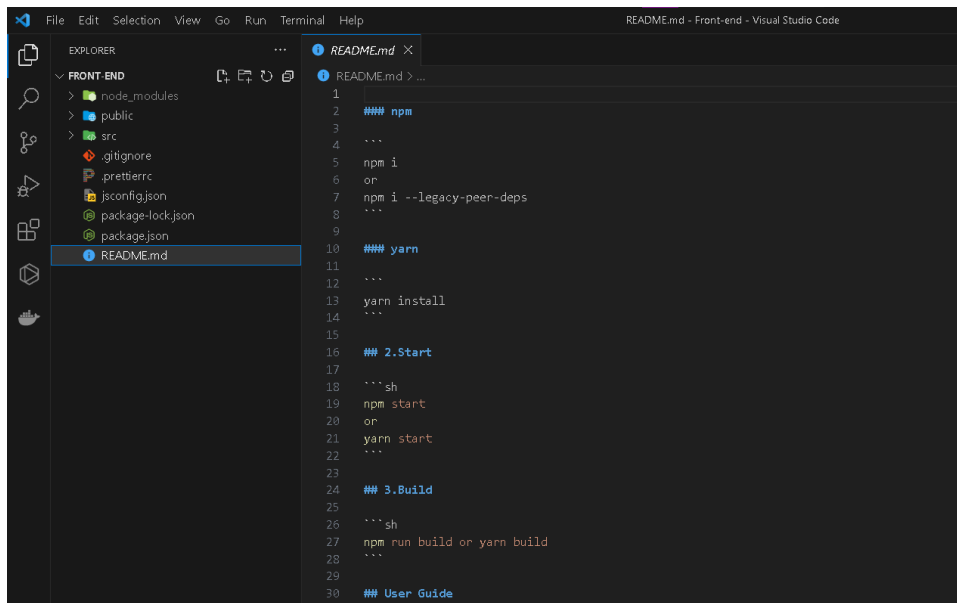
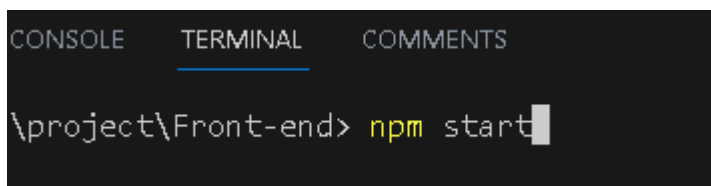


Figure 67 - Setup Management Web

6. In terminal, enter npm start



```
\project\Front-end> npm start
```

Figure 68 - Setup Management Web

+ Project start with <http://localhost:3000>

