MINISTRY OF EDUCATION AND TRAINING

FPT UNIVERSITY

Capstone Project Document

Rheinol Quick Service

Group 15 - IS	
Group members	Lê Đình Thiện Vũ – SE62590 (Leader) Phan Ánh Phúc – SE62543 Đặng Hữu Phương Nam – SE62167
Supervisor	Nguyễn Thị Cẩm Hương
Ext. Supervisor	N/A
Project Code	RQS

– Ho Chi Minh City, 20th August, 2019 –

Table of content

List Image

List Table

A. Proposal

1. Capstone Project name:

English: Rheinol Quick Service

Vietnamese: Hệ thống dịch vụ Rheinol

Abbreviation: ---

Building an application to manage Rheinol quick service. The purpose of Rheinol selling their products and services regarding luxury cars such as selling accessories, interior decoration, outside decoration, quickly maintenance and basic services. The system includes below main features:

- Buying and selling accessories
- Supply repairing, maintenance
- Launch new products or services for customers
- Notify maintenance schedule for customer's car
- Statistic report regarding all of services.

2. Main proposal content (including result and product)

a) Theory and practice (document):

- The student should apply the software development process and the UML
- Software artifacts include User Requirement, Software Requirement Specification, Architecture Design, Detail Design, System Implementation and Testing Document, Installation Guide, sources code, and deployable software packages.
- MVC should be applied
- Server-side technique:
  - Database design, OOA, OOD, OOP, MVC, Java or .Net technology, …
- Client-side technique
  - HTML5, CSS, JavaScript, jQuery, Ajax, Androids …
- Communication technique
  - Exchange information and transfer data in effective in networks, communicating protocol between mobile devices, ...
- Research
  - Algorithms, NFC …

b) Program:

- Web application
  - Director: could view all of reports
  - Admin: import, export accessories & products, add services, manage members and launch promotion programs.
  - Sales clerk: could manage order, manage account.
- Mobile application:

+ Used for customers

- Register members
- Register car’s information (1 or more). Adding information of car transfer.
- Select services, view history of used services or maintenance
- Book service basing on available slot
- Receive notification for the next services/maintenance
- View information about promotion, members and balance.
- Buy member card or save promotion.

+ Used for technical: check service, confirm to export invoice

- All of the management functions of the system must be implemented to support the operating system in the best.

B. Introduction

Swd is a medium-sized independent company, appearing from former FINA Schmierstoffwerk Duisburg. Swd has been developing, manufacturing and distributing high-quality lubricants and specialty for over 50 years. The Swd-Rheinol® brand has existed since 1998. Do not stop there, they are trying to extend business marketing by providing services, maintenance, repair and so on.

With the development of the technology revolution, we introduce two tools: first is a website administrator system for managing the business and the second is a mobile application. Customers can book any services by touching on their smartphone, hence implicitly making convenient and proactive.

1. Current Situation

When customers visit the Rheinol’s website, they can only see some product’s information or introduction to the company. Customers can’t buy products or book services through the website. Moreover, they do not know the available time of the garage to reserve. And with the current system, customers are not informed about the time to return to maintain replacement or used equipment or services.

2. Problem Definition

Below are the disadvantages of Rheinol website:

- The website just displays Rheinol’s information and products.
- Customers cannot buy products or use Rheinol's services through the website.
- The system has no control over the services or products they are trading.
- The system does not manage loyal customers to offer promotions.

3. Proposed Solution

We provide a solution about building web and mobile application to manage Rheinol quick service. The purpose of Rheinol is selling their products and services regarding luxury cars such as selling accessories, interior decoration, outside decoration, quickly maintenance and basic services. The system should be the following features:

3.1. Main features

- Buying and selling accessories.
- Supply repairing, maintenance.
- Launch new products, services, and promotions for customers.
- Notify a maintenance schedule for a customer's car.
- Statistic report regarding all of the services.
- Customer can book services base on the available slot.

3.2. Advantages and disadvantages

3.2.1 Advantages

- Help customer improve user experiment in using Rheinol quick service.
- Admin manages product, service, and the system more easily.
- Resolve the issue of warranty scheduling for Rheinol.
- Optimize time for the amount of space in the garage
- Manage customer accounts and offer specific promotions exactly.

3.2.2 Disadvantages

- Only support the Vietnamese language for web and mobile application.

4. Functional Requirements

Function requirements of the system are listed below:

4.1. Buying products and accessories:

- Customer must be authorized by logging in.
- The customer could buy multiple products or accessories.
- The customer could view all products and accessories if they are available.
- The customer could add products and accessories into the cart.
- The customer could pay the order by using the wallet in the application as long as the amount in the wallet is greater than or equal to the amount in the bill.
- If payment is successful, the system shall save purchase history into a customer's account.

4.2. Booking the services:

- Customer must be authorized by logging in.
- The customer could book multiple services.
- The customer could view all services if they are available.
- The customer could add services into the cart.
- The customer could select a time that they want to use services.
- If the customer selects an inappropriate time, the system shall notify.

4.3. Launch new product, service and promotion:

- Admin could add new product and service.
- Admin could create a new promotion for the customers.

4.4. Notifying maintenance schedule:

- The system shall inform customers on the mobile application about the warranty schedule for the services used before 7 days.

4.5. Statistic report regarding all of the services:

- The system shall generate a report about revenue services are used.
- Director could find report follow week, month, year.

5. Role and Responsibility

No	Full Name	Role	Position	Contact
1	Nguyen Thi Cam Huong	Project Manager	Supervisor	huongntc2@fe.edu.vn
2	Le Dinh Thien Vu	Developer	Leader	vuldtse62590@fpt.edu.vn
3	Phan Anh Phuc	Developer	Member	phucpase62543@fpt.edu.vn
4	Dang Huu Phuong Nam	Developer	Member	namdhpse62167@fpt.edu.vn

Table 1 - Role and responsibility

C. Software Process Model

This project is developed by using the waterfall model. We apply customized waterfall model to capable with the current situation in our team and our customer. We choose this model because of the following reasons:

- Based on the reality surveys, research, software Lucky Gara, the requirements of this project are stable, clear, fixed and well understood by all members.
- This project does not have anything to change the requirement from the customer.
- We do not meet the customer regularly.

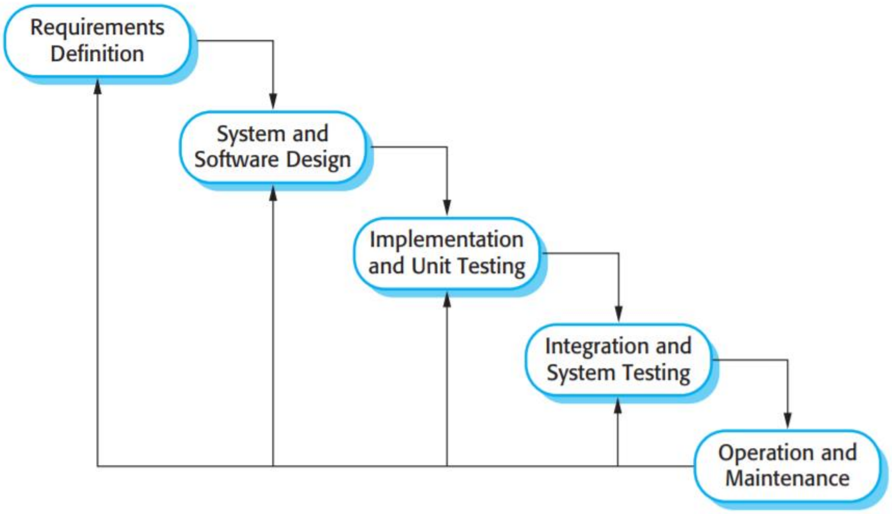
Requirements Definition
System and Software Design
Implementation and Unit Testing
Integration and System Testing
Operation and Maintenance

Figure 1 - Waterfall model

D. Conceptual Diagram

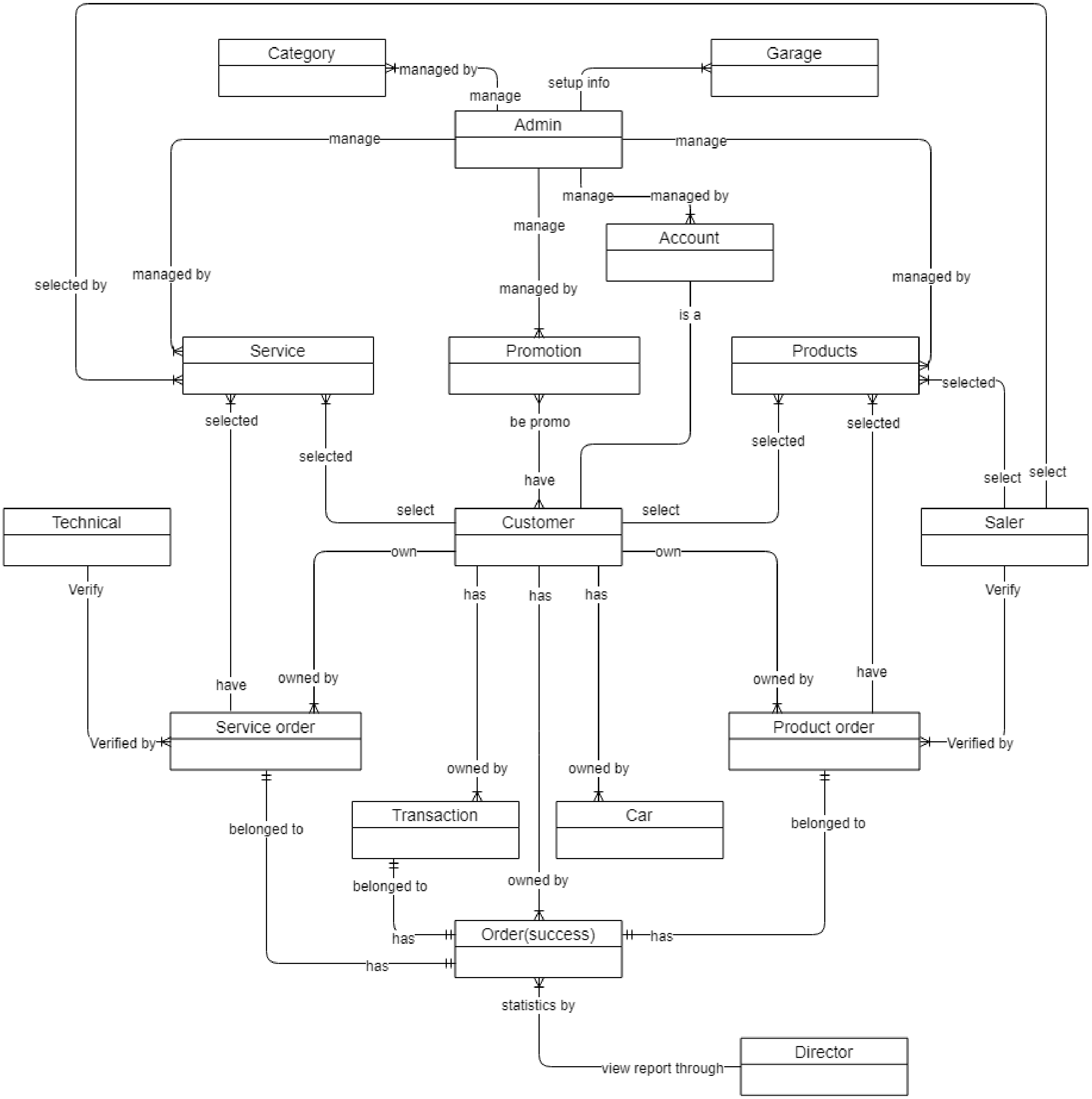

Figure 2 - Conceptual Diagram

E. [1]Use case Diagram

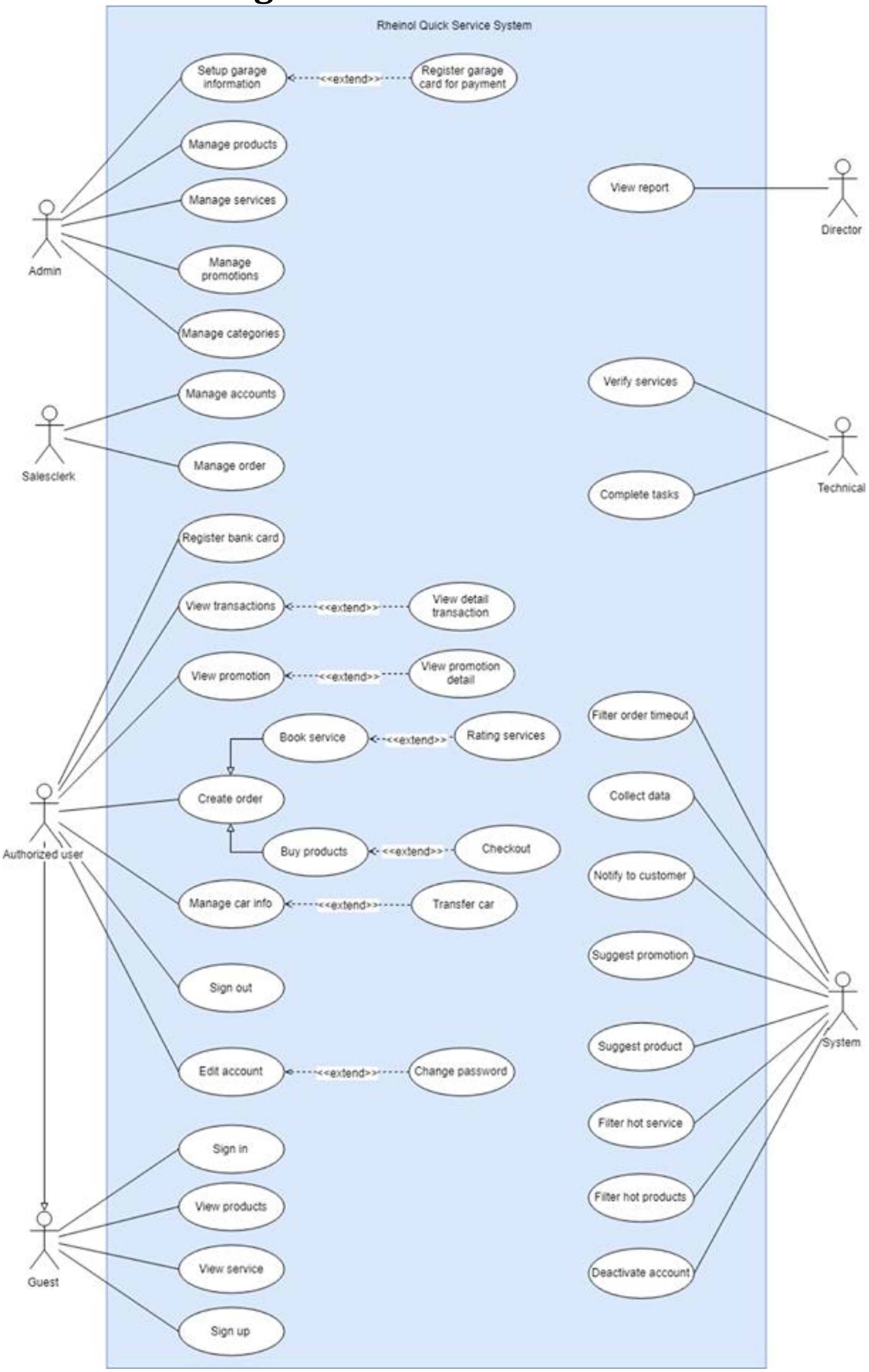

Figure 3 - Use case diagram

[1] These actors: admin, sales clerk, director can do all of functions of authorized user actor.

F. Use-case specification

1. <Guest> Overview use case

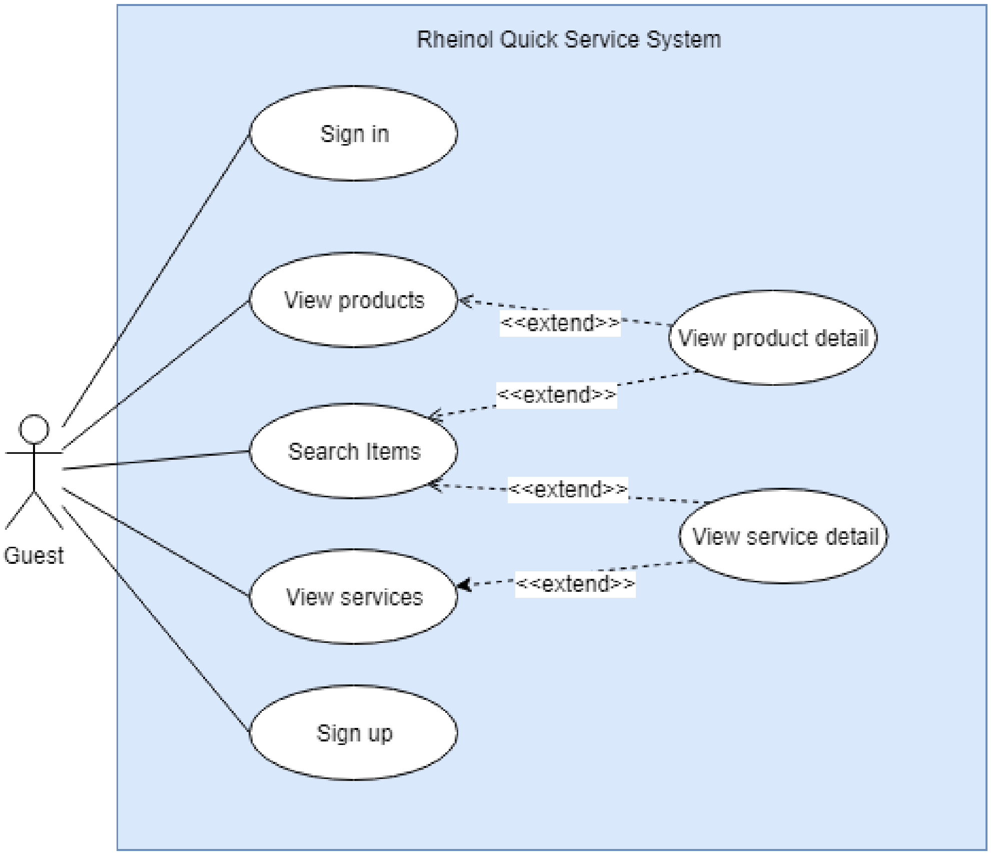

Figure 4 - <Guest> Overview use case

1.1. Sign in

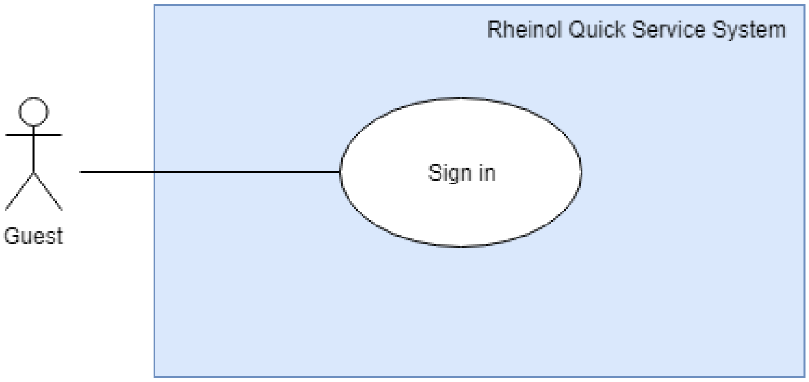

Figure 5 - <Use case detail> Sign in

USE CASE – RQS_UC_1			
Use Case No.	**RQS_UC_1**	**Use Case Version**	**2.0**
Use Case Name	**Sign in**		
Author	**NamDHP**		
Date	**02/06/2019**	**Priority**	**High**

Actor:

Guest

Summary:

This use case allows guest could sign in the system.

Goal:

Guest could sign in the system with a specific role.

Triggers:

The guest wants to sign in the system.

Preconditions:

Guest signed up the account before.

Postconditions:

Success: the guest is authorized with a specific role in the system.

Fail: the guest cannot sign in the system.

Main Success Scenario:

Step	Actor Action	System Response
1	The guest goes to "Sign in" view.	The system requires information from a guest: - Username: input text type. - Password: input text type.
2	The guest inputs information	
3.1	The guest sends the command to sign in the system.	The system will display an error message if the information does not match. [Exception 1]
3.2		Create

Exceptions:

No	Actor Action	System Response
1	The guest sends the command to sign in the system.	The system shows an error message to ask user input correct data.

Alternative:

No	Actor Action	System Response
1	The guest chooses “Sign in with Facebook”	The system shows a successful message and redirects to “User” view.

Business rules:

Password is encrypted before being sent to the server.

After login to the system, the guest will be redirected to a specific view based on their role.

On the mobile application system: an authorized user.

- The role is “Authorized user”, the system will display to the home view.

On the web system: a director, a sales clerk, an admin.

- If the role is “Director”, the system will display to Director view.
- If the role is “Sales clerk”, the system will display to Sales clerk Dashboard view.
- If the role is “Admin”, the system will display to Admin Dashboard view.

Table 2 - <Use case detail> Sign in

2. <Authorized user> Over view use case

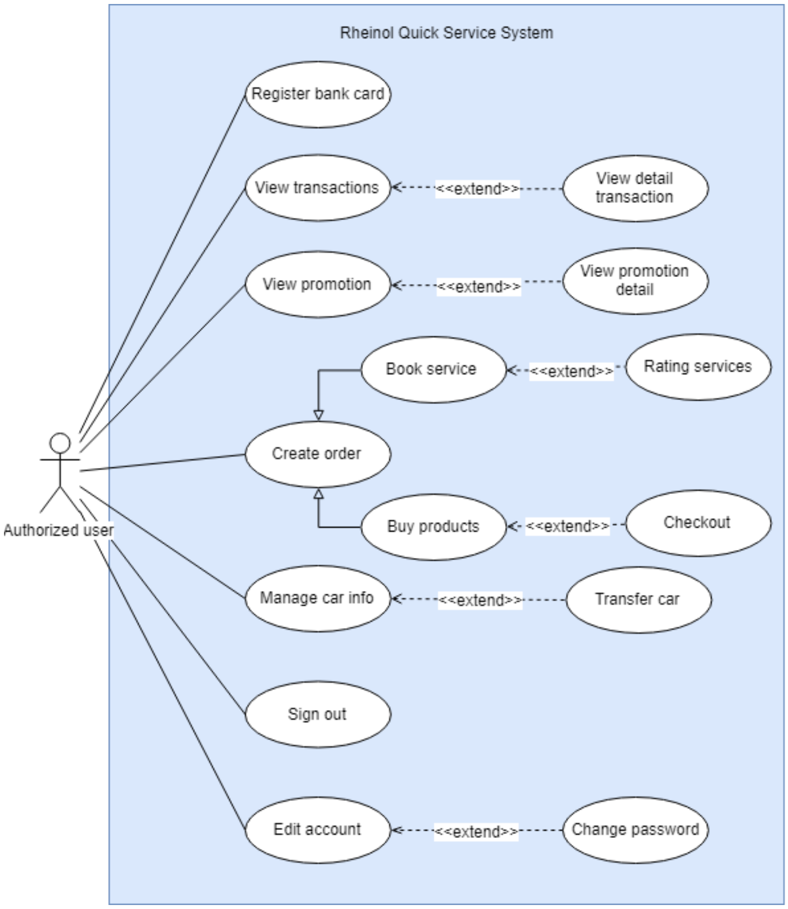

Figure 6 - <Authorized user> Overview use case

2.1. Register bank card

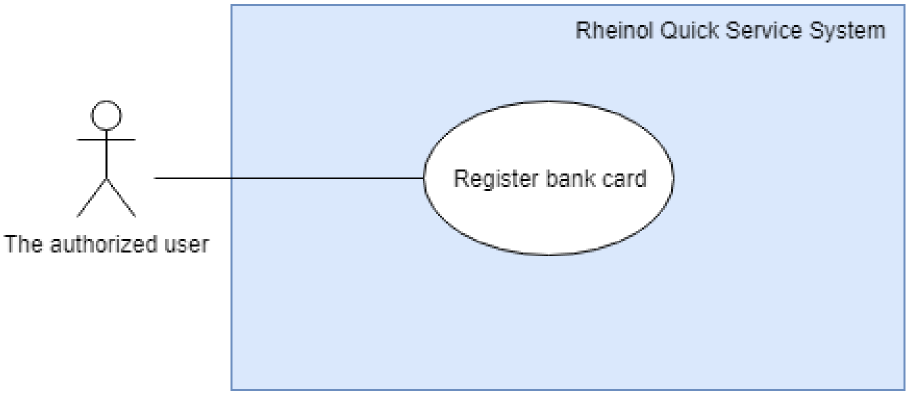

Figure 7 - <Use case detail> Register bank card

USE CASE – RQS_UC_2			
Use Case No.	**RQS_UC_2**	**Use Case Version**	**2.0**
Use Case Name	**Register bank card**		
Author	**VuLTD**		
Date	**02/06/2019**	**Priority**	**High**

Actor:

Authorized user

Summary:

This use case allows authorized user could register bank card in the mobile application.

Goal:

When the authorized user register bank card successfully, the authorized user can manipulate with wallet in the application.

Triggers:

The Authorized user wants to use the wallet in the application.

Preconditions:

Guest must log in into the mobile application with the authorized user role.

Postconditions:

Success: the bank card is registered successfully.

Fail: the bank card is not registered successfully.

Main Success Scenario:

Step	**Actor Action**	**System Response**
1	The authorized user goes to manage bank card view.	The system requires information: - Card number: a sequence of digits with 16 numbers. - Expiry: input text, type number. - CVC/CCV: input text, type number. - Cardholder's name: textbox, length more than 6 characters.

2	The authorized user input information	The system validates the information.
3.1	The authorized user sends the command to register bank card.	The system will display a message if the information is not matched. [Exception 1]
3.2		The system will display a successful message if the bank card is valid.

Exceptions:

No	Actor Action	System Response
1	The authorized user sends the command to register bank card.	The system shows an error message to ask user input wrong information.

Business rules:

The authorized user could register only one bank card in application.

The bank card accepted by the system is the visa card and master card.

The bank card is managed by a third party.

Table 3 - <Use case detail> Register bank card

2.2. Create the order

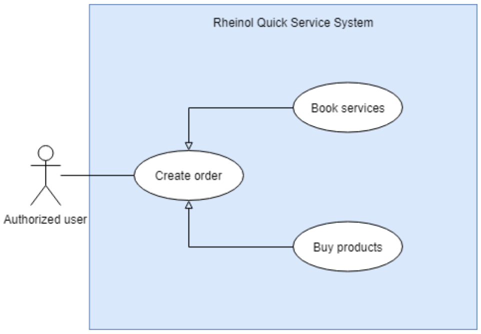

Figure 8 - <Use case detail> Create the order

USE CASE – RQS _UC_3			
Use Case No.	**RQS_UC_3**	**Use Case Version**	**2.0**
Use Case Name	**Create Order**		

Author	**PhucPA**		
Date	**02/06/2019**	**Priority**	**High**

Actor:

Authorized User

Summary:

The authorized user create order by themselves.

Goal:

When an authorized user creates order successfully, the order is created in system and system will assign a technical employee to take care of this order if the order includes doing service at the garage.

Triggers:

The user has an account and wants to buy some product or do some service or both at the garage.

Preconditions:

The user has an account and logged in on the mobile app.

If the order includes service, service must be booked on the free time of garage.

Postconditions:

Success: order is created in the system. If the order includes service, the system will send a notification to the technical employee who assigns for this service.

Fail: order is not created and the system sends fail notification to the client.

Main Success Scenario:

Step	Actor Action	System Response
1	User choose the product and add to cart	Cart is updated with the amount of item in cart and display in screen mobile.
2	User choose the service and add to cart [Alternative 1]	Request choose Date and choose the free time slot
3	User can choose the promotion that the system suggests for them	Display information of promotion in cart

	[Alternative 2]	
4	The user chooses to submit to create order	System process request create order [Exception 1,2]

Alternative Flow:

No	Actor Action	System Response
1	User don't choose service	
2	User don't choose promotion	

Exceptions:

No	Actor Action	System Response
1	Request create order is successfully	Order is created successfully and show detail order on screen
2	Request create order is failed	Network error message

Business rules:

Book service must be including date and time choice.

Table 4 - <Use case detail> Create order

2.3. Check out order

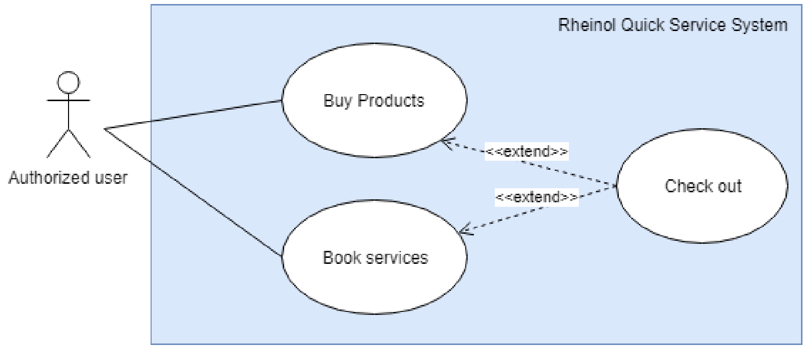

Figure 9 - <Use case detail> Check out order

USE CASE – RQS_UC_4			
Use Case No.	**RQS_UC_4**	**Use Case Version**	**2.0**

Use Case Name	**Check out order**		
Author	**NamDHP**		
Date	**02/06/2019**	**Priority**	**High**

Actor:

Authorized user

Summary:

This use case allows authorized user could check out the order in the mobile application.

Goal:

When the authorized user checks out the order successfully, the order's status is changed.

Triggers:

There is an order to be checked out.

Preconditions:

Guest must log in into the mobile application with the authorized user role.

The amount in the wallet must be greater than the amount of the bill.

Postconditions:

Success: the order's status is changed successfully.

Fail: the order's status is not changed successfully.

Main Success Scenario:

Step	**Actor Action**	**System Response**
1	The authorized user goes to "not payment order" view.	The system shows the completed order.
2	The authorized user sends a command to pay the order through the button.	The system shows "confirm" pop up.
3.1	The authorized user confirms to pay the order.	The system will display a message if the order cannot pay. [Exception 1]

3.2		The system will notify a success message and redirect to “order’s detail” view.

Alternative:

Step	Actor Action	System Response
1	The authorized user goes to “cart” view.	The system shows the list of product that is selected.
2	The authorized user sends a command to pay the order through the button.	The system shows “confirm” pop up.

Exceptions:

No	Actor Action	System Response
1	The authorized user confirms to pay the order.	The system shows an error message to ask user cash in the wallet.

Business rules:

If there are only products in the cart, the authorized user can pay at “cart” view. [Alternative]

If there are services in the order, the confirmation of service is “complete”, the authorized user can pay the order.

Table 5 - <Use case detail> Check out order

3. <Sales clerk> Overview use case

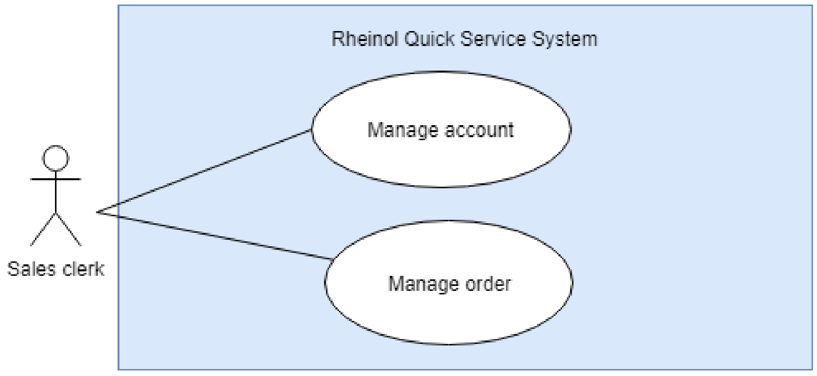

Figure 10 - <Sales clerk> Overview use case

3.1. Confirm to do order

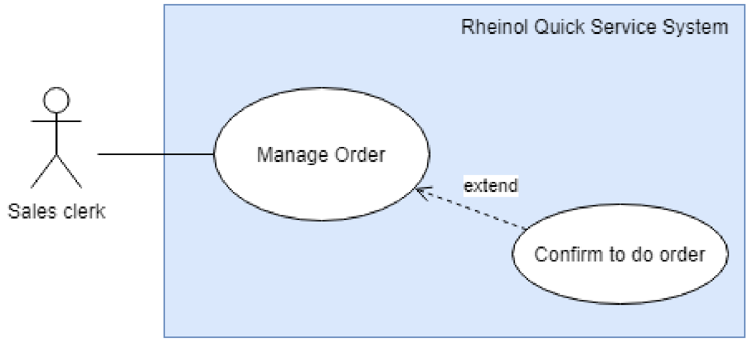

Figure 11 - <Use case detail> Confirm to do order

USE CASE – RQS_UC_5			
Use Case No.	**WCS_UC_5**	**Use Case Version**	**2.0**
Use Case Name	**Confirm to do order**		
Author	**NamDHP**		
Date	**02/06/2019**	**Priority**	**Normal**

Actor:

Sales clerk

Summary:

This use case allows the sales clerk could change order's status to confirm to do order.

Goal:

The sales clerk could change order's status to confirm successfully, and the notification is sent to technical to do order.

Triggers:

After the technical checks order and change the order's status to waiting.

Preconditions:

Guest must log in into the web system with the role Sales clerk.

Postconditions:

Success: the order is changed status, and the system sends the notification to technical to do order.

Fail: the order cannot be changed status, and the system shows the error message.

Main Success Scenario:

Step	Actor Action	System Response
1	The sales clerk goes to “Order” view.	The system shows the list order with “change status” button in each order.
2	The sales clerk sends a command to change the order’s status request through the button.	The system shows confirm pop up.
3.1	The sales clerk confirms to change the order’s status.	The system will display an error message if the order’s status cannot change. [Exception 1]
3.2		The order’s status is changed to confirm doing the order.

Exceptions:

No	Actor Action	System Response
1	The sales clerk confirms to change the order’s status.	The system will display an error message to notify the error.

Business rules:

The orders with a description and previously confirmed by the technical before, the sales clerk will be able to change the order’s status to confirm doing the order.

The order’s status is in progress.

Table 6 - <Use case detail> Confirm to do order

4. <Technical> Overview use case

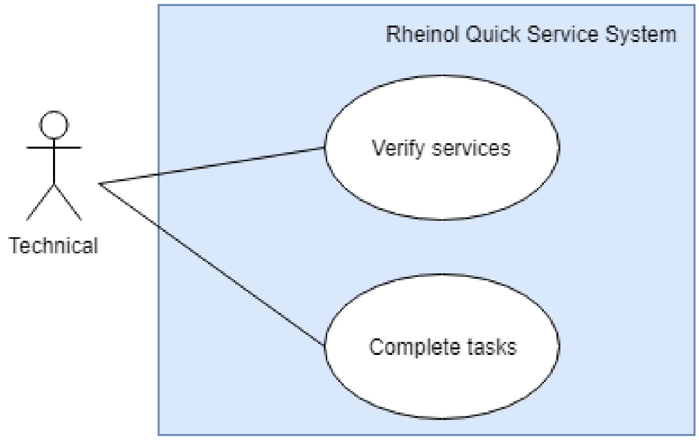

Figure 12 - <Technical> Overview use case

4.1. Verify service

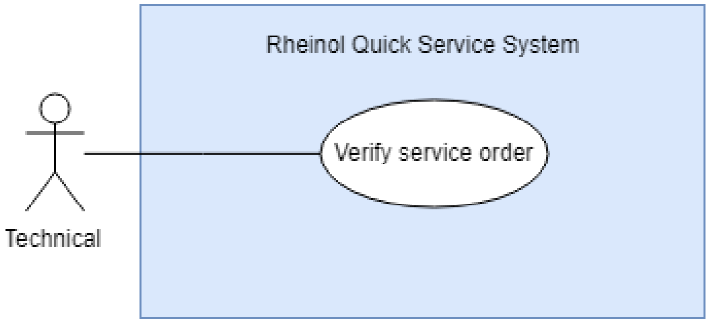

Figure 13 - <Use case detail> Verify service

USE CASE – RQS_UC_6			
Use Case No.	**RQS_UC_6**	**Use Case Version**	**2.0**
Use Case Name	**Verify service**		
Author	**NamDHP**		
Date	**02/06/2019**	**Priority**	**Normal**

Actor:

Technical

Summary:

This use case allows the technical could change order's status to verify services.

Goal:

The technical could change order's status to verify services successfully, and the notification is sent to the sales clerk.

Triggers:

There is the order has service which was booked time before.

Preconditions:

Guest must log in into the web system with the role Technical.

Postconditions:

Success: the order is changed status, and the system sends the notification to the sales clerk.

Fail: the order cannot be changed status, and the system shows the error message.

Main Success Scenario:

Step	Actor Action	System Response
1	The technical go to “Verify service” view	The system shows the order which is waited to verify.
2	The technical select the order	The system shows the order’s detail and displays field data for technical input: Description: input type text.
3	The technical inputs description	
4.1	The technical send command to change status.	The system will display an error message if the order’s status cannot change. [Exception 1]
4.2		The order’s status is changed to verify the service.

Exceptions:

No	Actor Action	System Response
1	The technical confirms to change the order’s status.	The system will display an error message to notify the error.

Business rules:

- Only technical can verify services.

Table 7 - <Use case detail> Verify service

5. <Admin> Overview use case

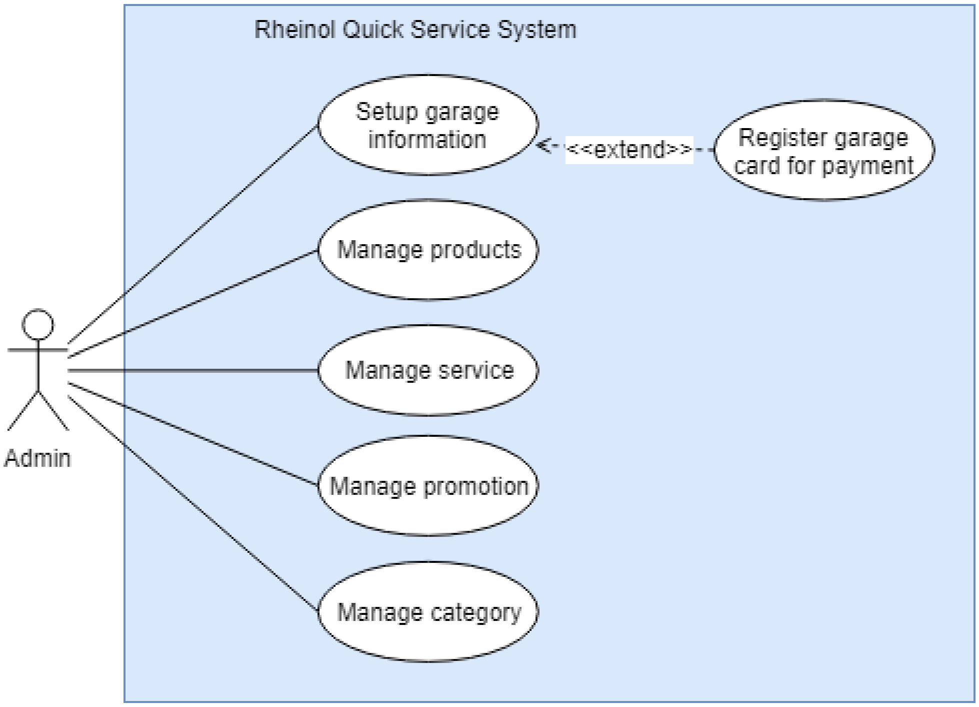

Figure 14 - <Admin>Overview use case

5.1. Add product

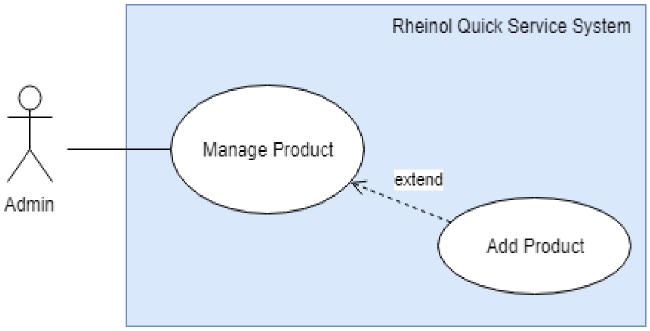

Figure 15 - <Use case detail> Add product

USE CASE – RQS _UC_7			
Use Case No.	**RQS _UC_7**	**Use Case Version**	**2.0**
Use Case Name	**Add Product**		
Author	**VuLTD**		
Date	**22/05/2019**	**Priority**	**High**
Actor: Admin			

Summary:

This use case allows admin could add a new product.

Goal:

When admin added product successfully, this product was saved in the system.

Triggers:

There is a new product that needs to be added to the system.

Preconditions:

Guest must log in into the web application with role admin.

Postconditions:

Success: A product is saved in the system.

Fail: The product is not added to the system and shows a notification.

Main Success Scenario:

Step	**Actor Action**	**System Response**
1	Admin goes to new product view.	The system requires information from Admin: - Product's name: free input text, required, length 6-200. - Product's price: free input text, required, number. - Product's image: required, select at least one picture. - Product's capacity: free input text, number. - Product's description: free input text, length 3- 250 - Category: required, select one category.
2	Admin inputs information	
3.1	Admin sends a command to add a new product request.	The system validates information, the system will display a message if the information is not matched. [Exception 1,2]
3.2		The product is added to the database and system will show message successfully. All input text will reset.

Exceptions:

No	Actor Action	System Response
1	Admin sends a command to add a new product request.	The system shows an error message to ask user input missing required fields or condition fields.
2	The product name has existed in the system	The message will appear to notify admin that product's name has existed.

Alternative:

No	Actor Action	System Response
1	Admin chooses import Excel.	The system shows device data that an admin can choose the excel file.
2.1	Admin sends a command to add a new product request.	The system validates information, the system will display a message if the information in the excel file is not matched.
2.2		The products are added to the database and system will show message successfully.

Business rules:

The product belongs to only one category.

The new product is added that must have the approval of the superior.

When the new product is added to the system, the product's status is always enabled.

Only admin can add a new product to the system.

Table 8 - <Use case detail> Add product

5.2. Add service

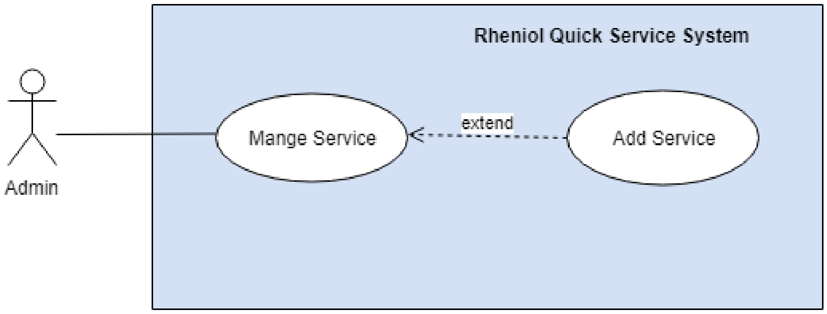

Figure 16 - <Use case detail> Add service

USE CASE – RQS_UC_8			
Use Case No.	**RQS_UC_8**	**Use Case Version**	**2.0**
Use Case Name	**Add Service**		
Author	**PhucPA**		
Date	**02/06/2019**	**Priority**	**High**

Actor:

Admin

Summary:

This use case allows Admin could add new service.

Goal:

When admin added service successfully, service was saved in System.

Triggers:

There is a new service that needs to be added into System.

Preconditions:

Guest must log in into the web application with role Admin.

Postconditions:

Success: service is saved in System.

Fail: service is not added to the System and show notification.

Main Success Scenario:

Step	**Actor Action**	**System Response**
1	Admin goes to new service view.	Service information screen display fields data for user input: - Name: textbox, required, length more than 6 characters.

		- Price: textbox, required, type integer. - Duration: textbox, required, type double. - Time maintenance: textbox, number. - Description: textbox, length more than 6 characters. - Image: select type, select one image. - Category: combo box, required, select a category
2	Admin inputs information	
3.1	Admin sends a command to add a new service request.	The system validates information, the system will display a message if the information is not matched. [Exception 1,2]
3.2		The service is added to the database and system will show message successfully. All input text will reset.

Exceptions:

No	Actor Action	System Response
1	Admin sends a command to add a new service request.	The system shows an error message to ask user input missing required fields or condition fields.
2	The name has existed in the system	The message will appear to notify admin that username has existed.

Business rules:

The service belongs to only one category.

The new service is added that must have the approval of the superior

When the new service is added to the system, the service's status is always enabled.

Only admin can add a new service to the system.

Table 9 - <Use case detail> Add service

5.3. Add new promotion

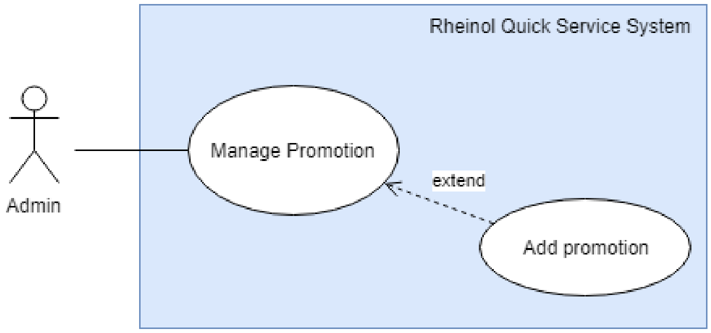

Figure 17 - <Use case detail> Add new promotion

USE CASE – RQS_UC_9			
Use Case No.	**RQS_UC_9**	**Use Case Version**	**2.0**
Use Case Name	**Add Promotion**		
Author	**NamDHP**		
Date	**22/05/2019**	**Priority**	**High**

Actor:

Admin

Summary:

This use case allows Admin could add new promotion.

Goal:

When admin added promotion successfully, a new promotion is saved in System.

Triggers:

There is a new promotion that the enterprise wants to apply and add to the system.

Preconditions:

Guest must log in into the web application with role Admin.

Postconditions:

Success: a new promotion is saved in the system.

Fail: promotion is not added to the System and shows a notification.

Main Success Scenario:

Step	**Actor Action**	**System Response**
1	Admin goes to new promotion view.	The system requires information from Admin: - Promotion’s name: free input text, required, length 6-100.

		- Time’s promotion: DateTime input, required, there are 2 fields “From” and “To”. - Product: product’s list. - Service: service’s list. - Promotion’s value: input text, required, number: from 1 to 100 (with value “%”). - Promotion’s image: required, select one picture. - Promotion’s description: input text, length: 5 – 250. - Promotion’s members: select from list checked box. - Promotion’s type: required, select one type.
2	Admin inputs information	List product: the system will show popup modal and admin can select any products. List service: the system will show popup modal and admin can select many services.
3.1	Admin sends a command to add a new promotion request.	The system validates information, the system will display a message if the information is not matched. [Exception 1,2]
3.2		The promotion is added to the database and system will show message successfully. All input text will reset.

Exceptions:

No	Actor Action	System Response
1	Admin sends a command to add a new promotion request.	The system shows an error message to ask user input missing required fields or condition fields.
2	Admin does not select a product or service.	The message will appear to notify admin that admin has to select at least one product or service to be used in promotion.

Business rules:

The new promotion is added that must have the approval of the superior with a complete document.

The promotion can apply with multiple products and services.

If Promotion’s type is “combo”, the products or services are selected together according to regulations, the promotion will apply. If Promotion’s type is “single”, and the products or services are selected to follow the promotion, the promotion will apply. The promotion will apply to the member’s type that was selected. When the new promotion is added to the System, the promotion’s status is always disabled.

Table 10 - <Use case detail> Add new promotion

6. <System> Overview use case

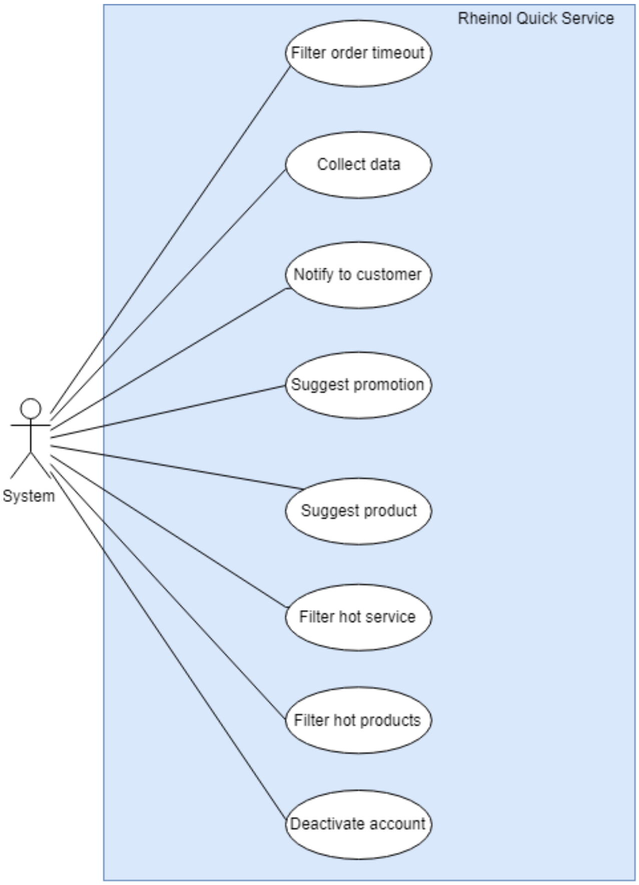

Figure 18 - <System> Overview use case

6.1. Notify to customer

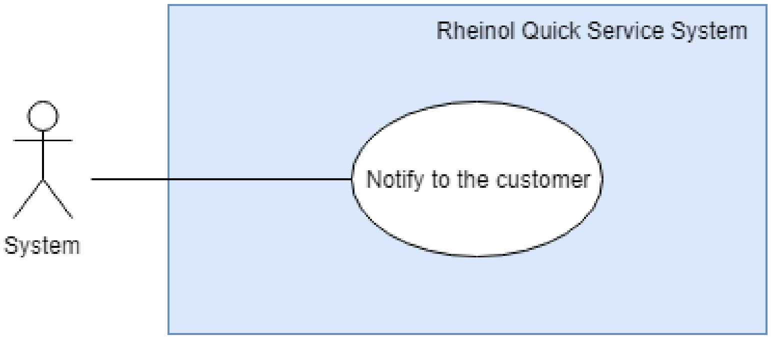

Figure 19 - <Use case detail> Notify to customer

USE CASE – RQS_UC_10			
Use Case No.	**RQS_UC_10**	**Use Case Version**	**2.0**
Use Case Name	**Notify to the customer**		
Author	**VuLTD**		
Date	**02/06/2019**	**Priority**	**High**

Actor:

System

Summary:

This use case allows the system could notify some information to the customer.

Goal:

The system could notify the promotion’s information or warranty service.

Triggers:

There are promotions or warranty services that need to be notified to customers.

Preconditions:

N/A.

Postconditions:

Success: the system notifies information to the customer.

Fail: the system cannot notify information to the customer.

Main Success Scenario:

Step	**Actor Action**	**System Response**
1	The system notifies information to the customer.	The notification will be sent to the customer.

Exceptions:

No	**Actor Action**	**System Response**

1	The system notifies information to the customer.	The notification will not be sent to the customer.
Business rules: The system will send the promotion's notification before 7 days when the promotion is applied. The system will send the warranty's notification before 7 days when the maintenance service is due.		

Table 11 - <Use case detail> Notify to customer

7. <Director> Overview use case

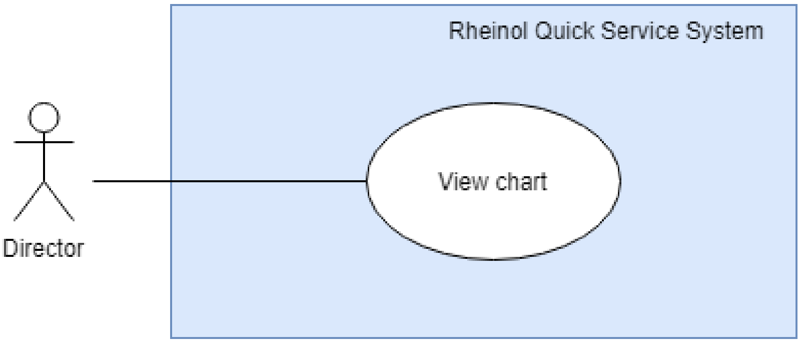

Figure 20 - <Director> Overview use case

USE CASE – RQS _UC_11			
Use Case No.	**RQS _UC_11**	**Use Case Version**	**2.0**
Use Case Name	**View chart**		
Author	**VuLTD**		
Date	**02/06/2019**	**Priority**	**High**
Actor: Director **Summary:** This use case allows the director could view the chart. **Goal:** When the director selects the report's type and chooses a suitable timeline, the director can view report chart. **Triggers:** The director wants to view the report.			

Preconditions:

Guest must log in into the web application with role Director.

Postconditions:

Success: report chart will be shown.

Fail: report chart will not be shown and the failed message.

Main Success Scenario:

Step	**Actor Action**	**System Response**
1	The director goes to report view.	Report screen display fields data which information for the user: Type Timeline: select box, select one type. Time input: input type date, required.
2	Director select and input information	
3.1	Director sends a command to view the report request.	The system check information, the system will display a message if the information is not matched. [Exception 1]
3.2		The report chart will be shown.

Exceptions:

No	**Actor Action**	**System Response**
1	Director sends a command to view the report request.	The system shows an error message to ask user input missing required fields or condition fields.
2	The time is not suitable	The message will appear to notify admin that time is invalid.

Business rules:

When the director wants to view "custom date timeline" type, the director could only select a time that is an interval of 10 days apart.

Table 12 - <Use case detail> View chart

G. Architectural Diagram

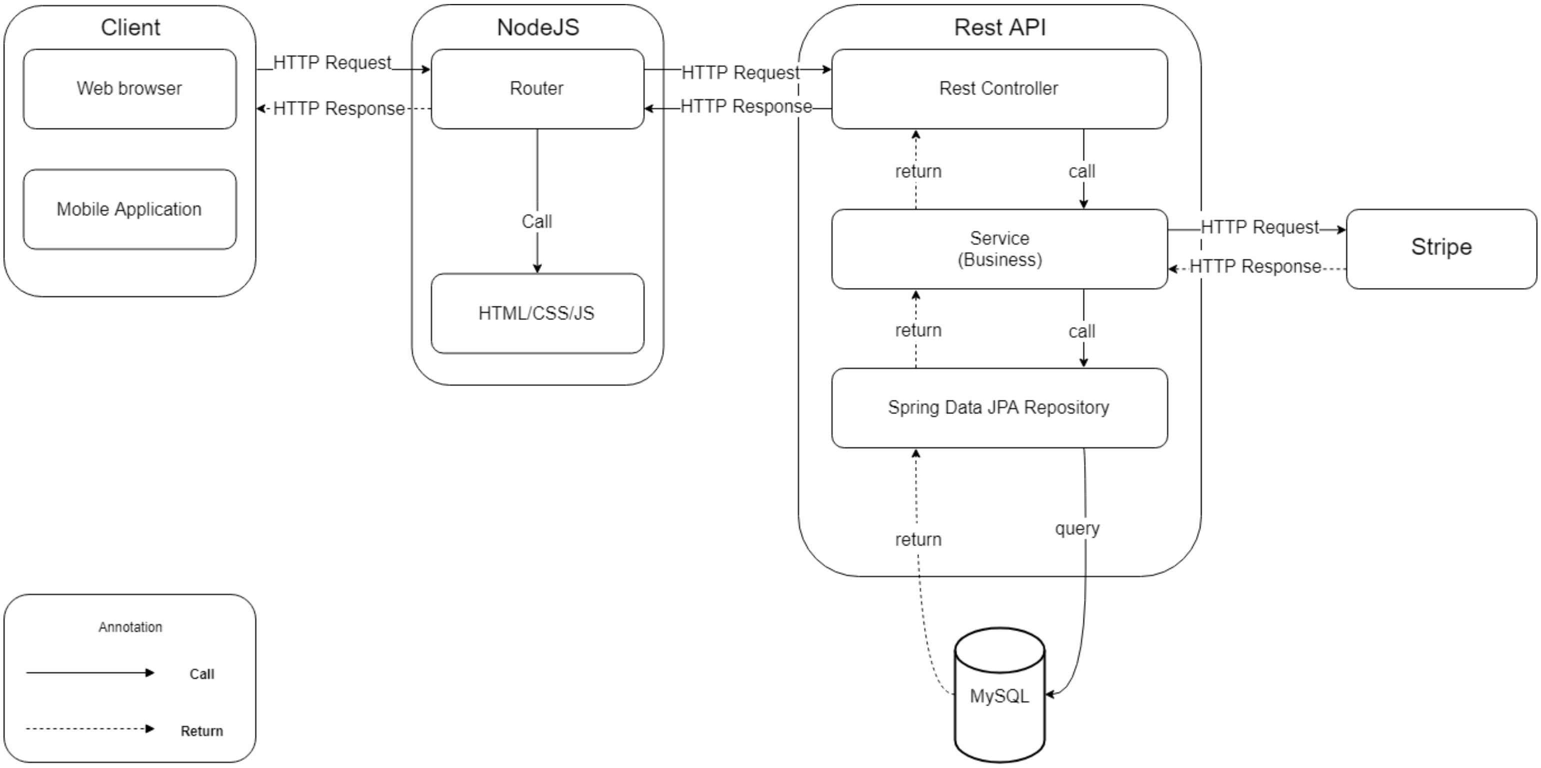

Figure 21 - Architectural Diagram

H. Component Diagram

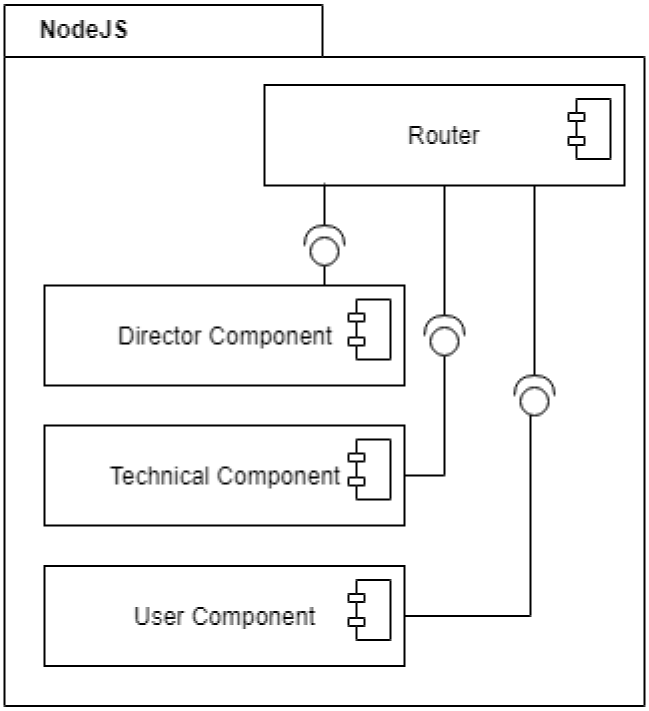

Figure 22 - Component diagram

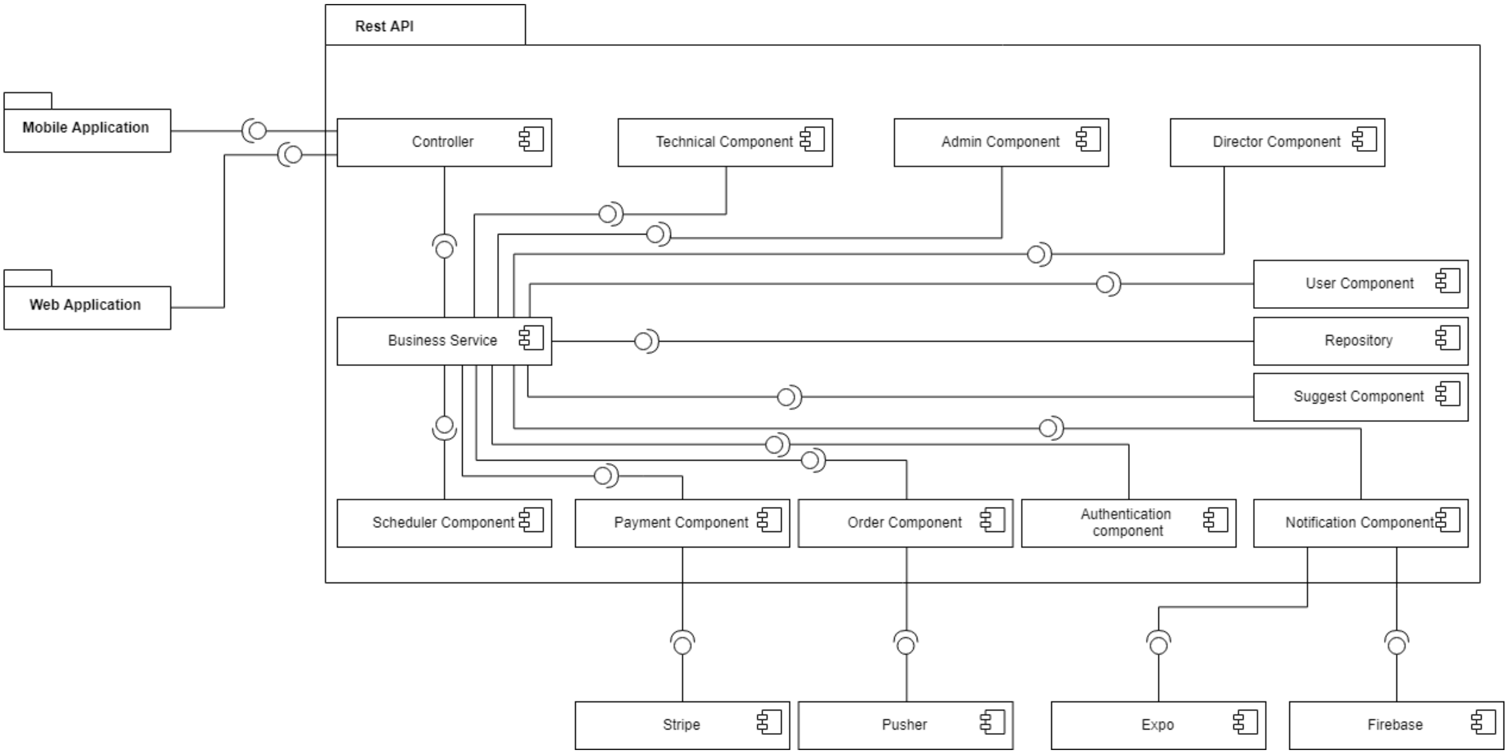

Figure 23 - Component diagram

I. Class Diagram

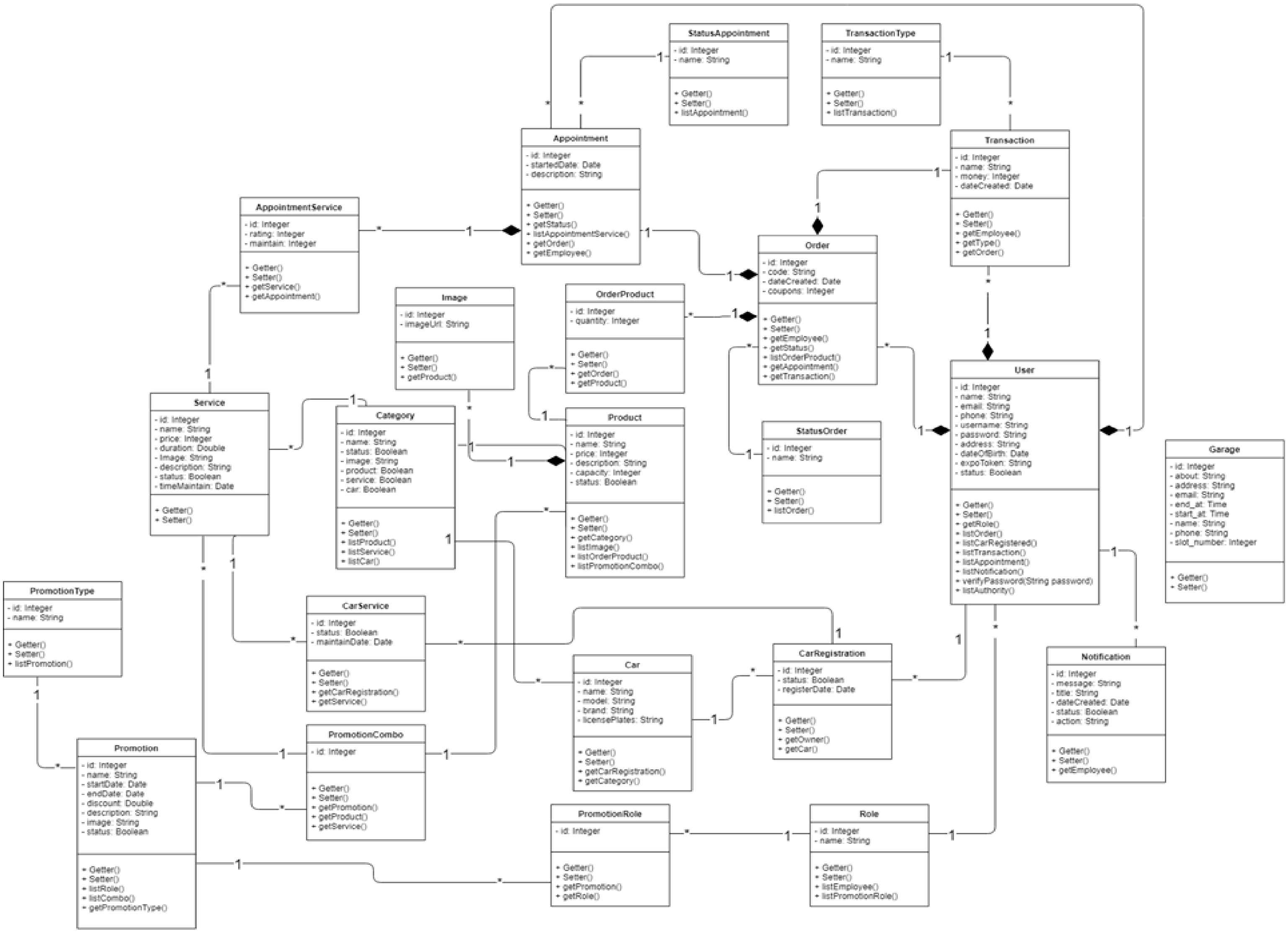

Figure 24 - Class diagram

J. Entity Relationship Diagram

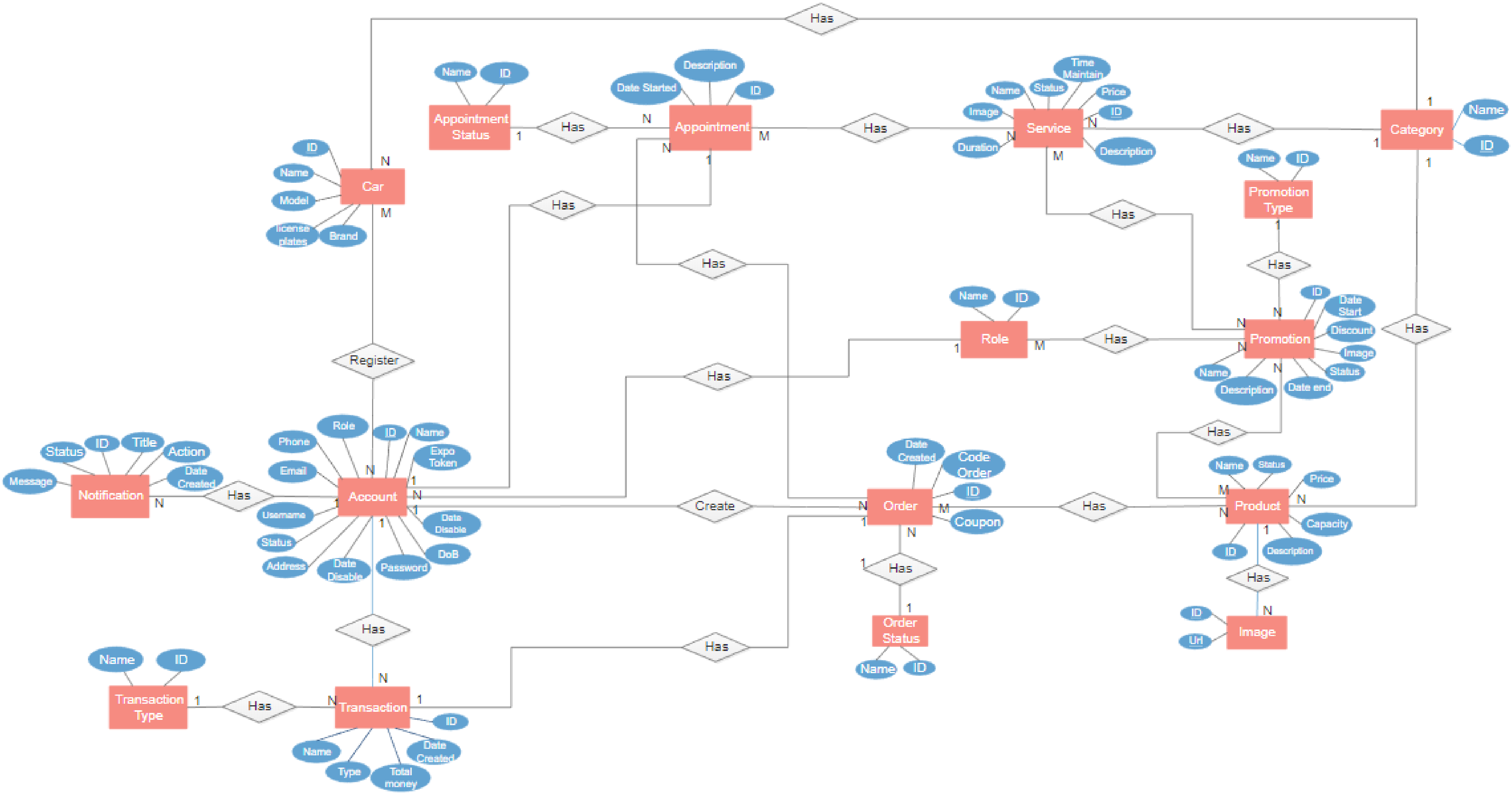

Figure 25 - Entity relationship diagram

K. Interaction Diagram

1. Web application

1.1. Log in

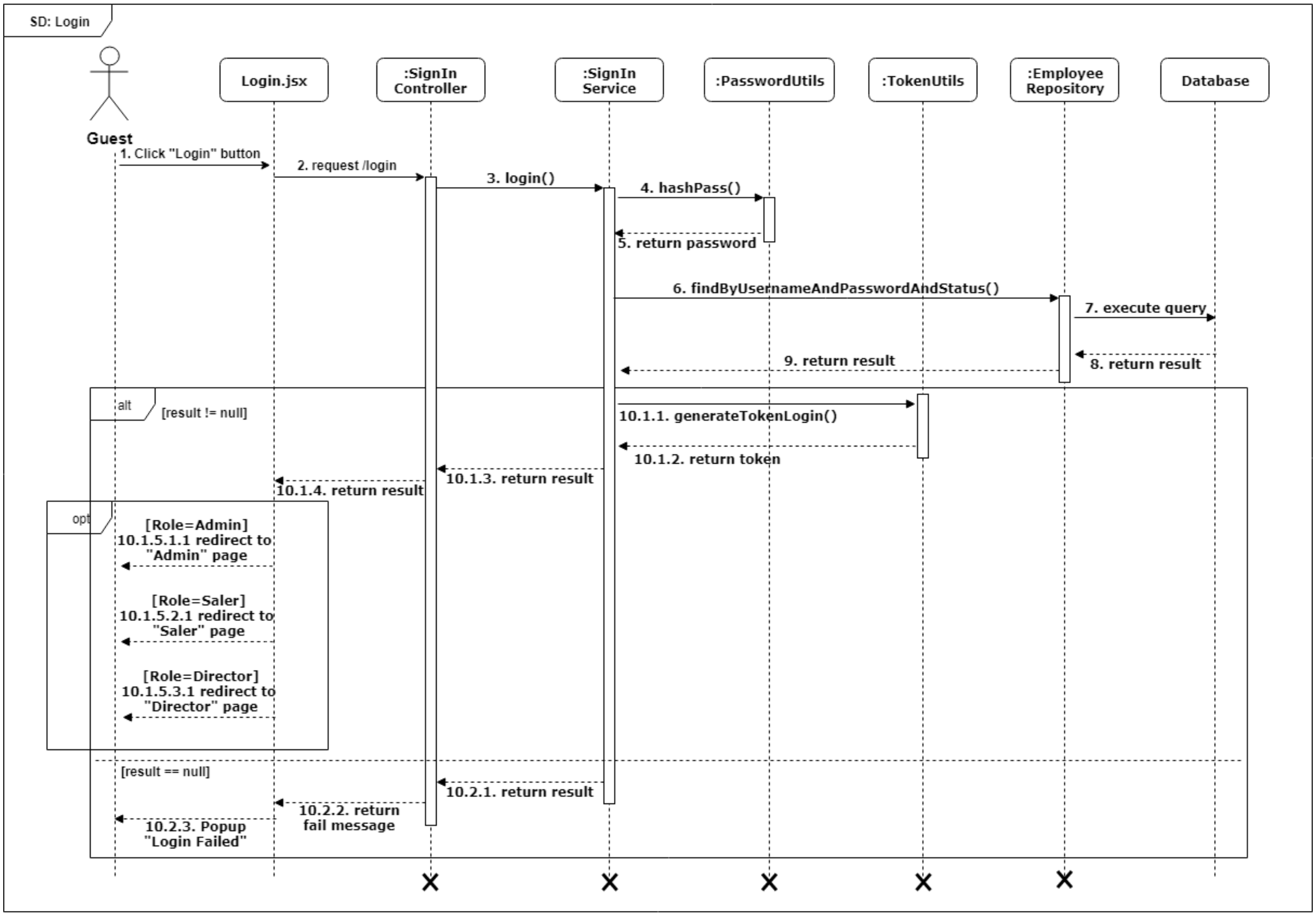

Figure 26 - <Sequence> Log in

1.2. Add new order

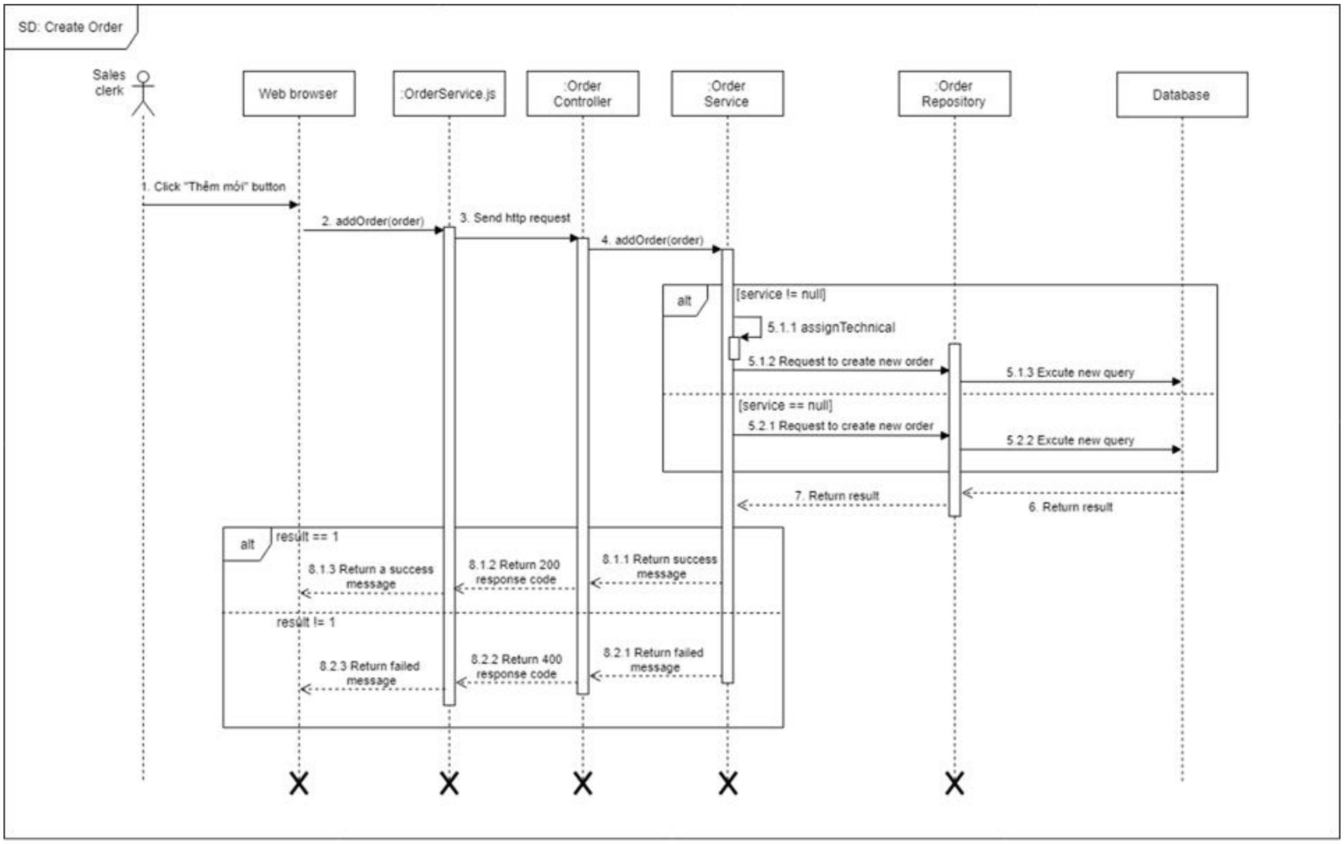

Figure 27 - <Sequence> Add new order

1.3. Add new service

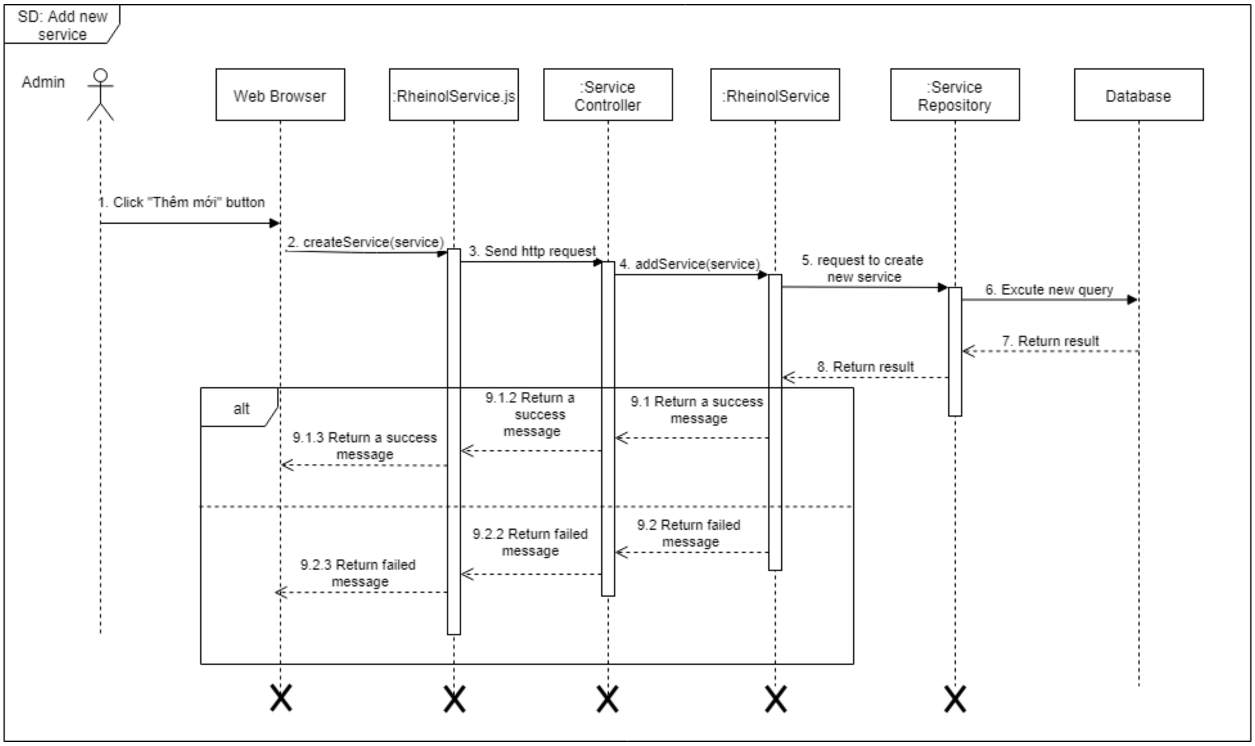
SD: Add new service
Admin
Web Browser
:RheinolService.js
:Service Controller
:RheinolService
:Service Repository
Database
1. Click "Thêm mới" button
2. createService(service)
3. Send http request
4. addService(service)
5. request to create new service
6. Excute new query
7. Return result
8. Return result
alt
9.1.2 Return a success message
9.1 Return a success message
9.1.3 Return a success message
9.2.2 Return failed message
9.2 Return failed message
9.2.3 Return failed message

Figure 28 - <Sequence> Add new service

1.4. Add new product

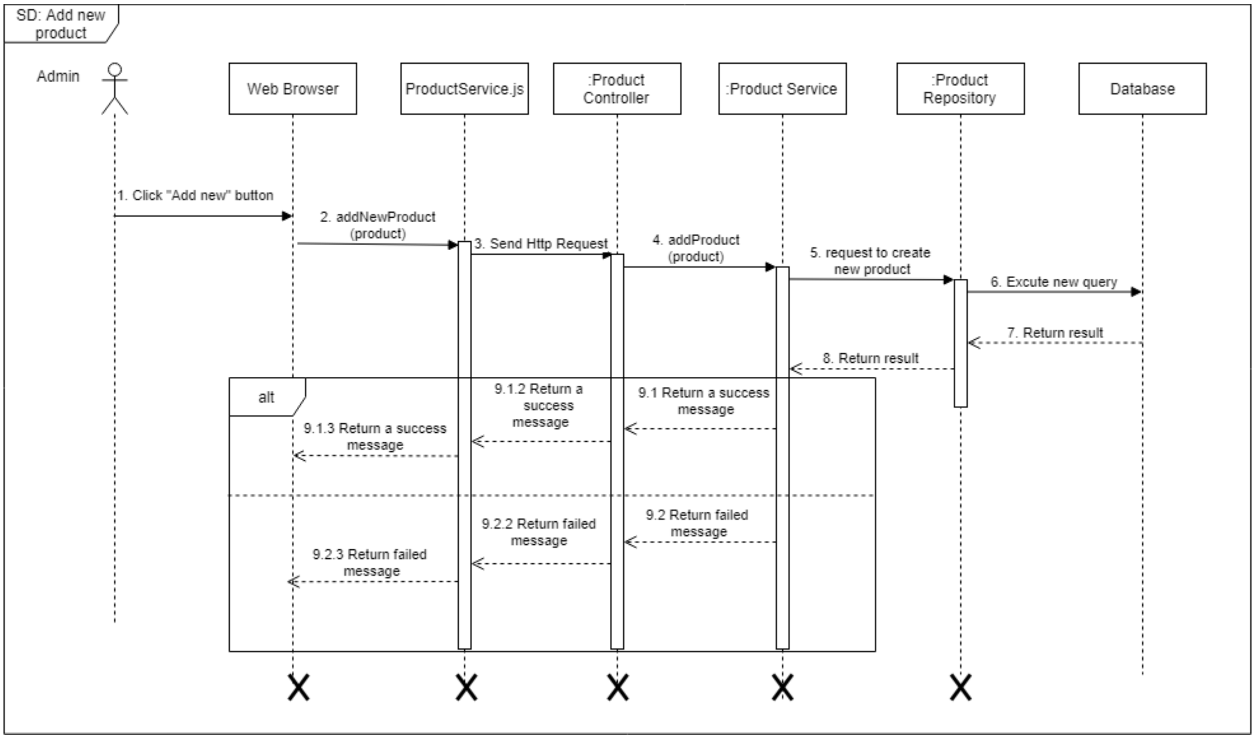
SD: Add new product
Admin
Web Browser
ProductService.js
:Product Controller
:Product Service
:Product Repository
Database
1. Click "Add new" button
2. addNewProduct (product)
3. Send Http Request
4. addProduct (product)
5. request to create new product
6. Excute new query
7. Return result
8. Return result
alt
9.1.2 Return a success message
9.1 Return a success message
9.1.3 Return a success message
9.2.2 Return failed message
9.2 Return failed message
9.2.3 Return failed message

Figure 29 - <Sequence>Add new product

1.5. Notify to customer

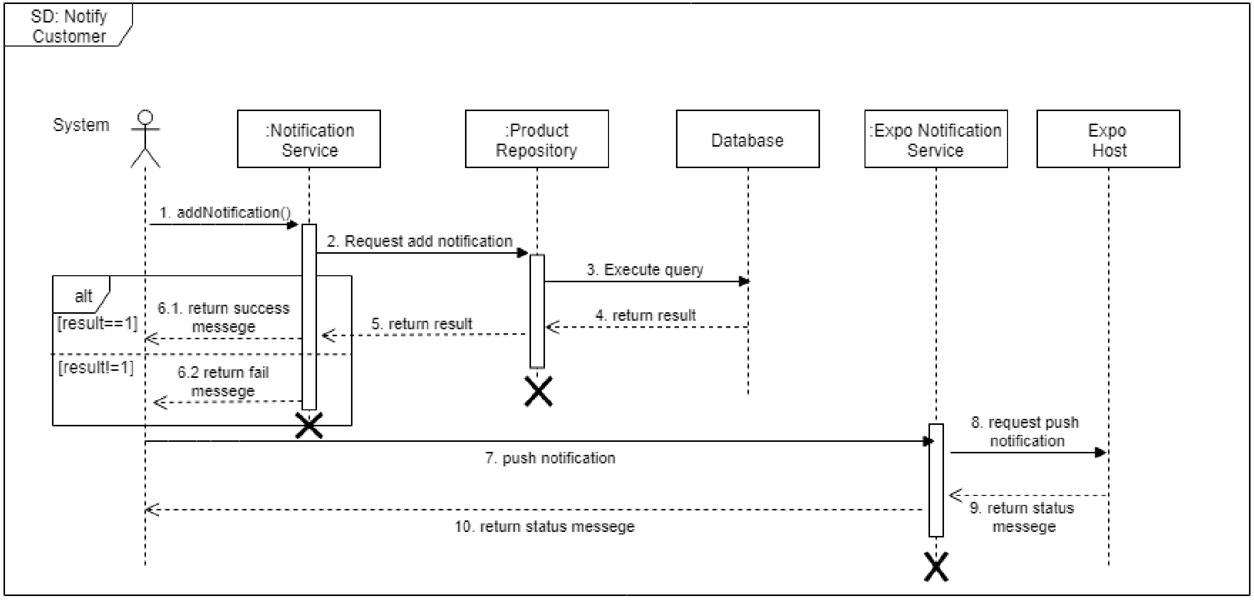

Figure 30 - <Sequence> Notify to customer

1.6. Filter order timeout

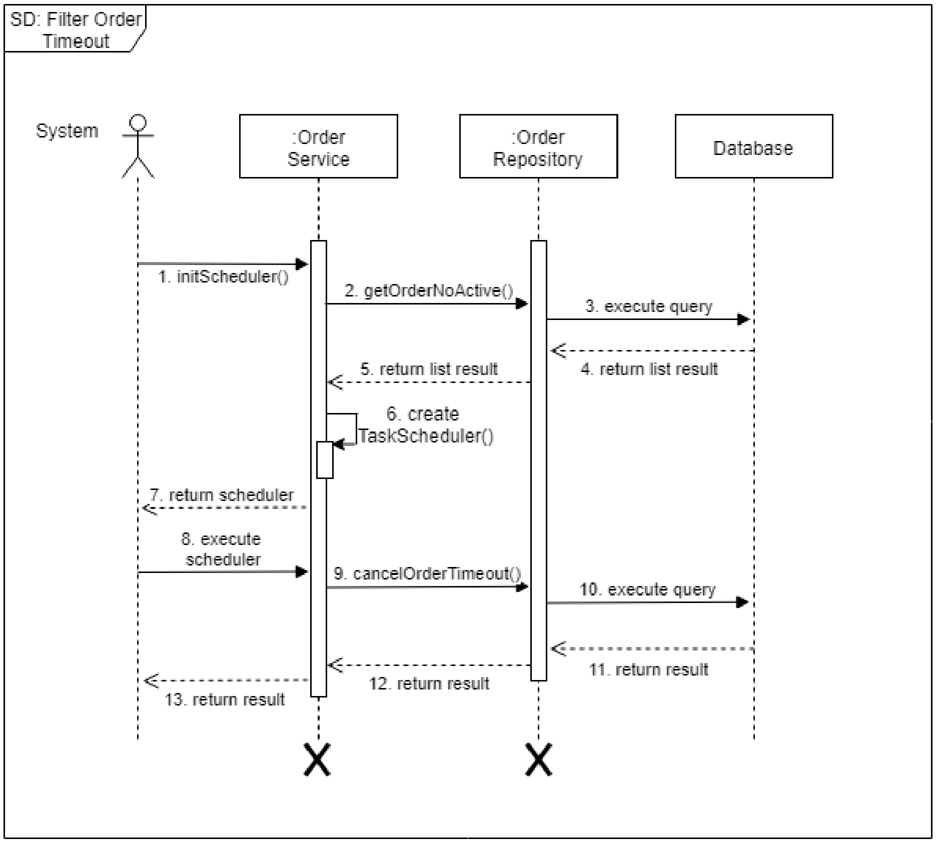

Figure 31 - <Sequence> Filter order timeout

2. Mobile application

2.1. Log in

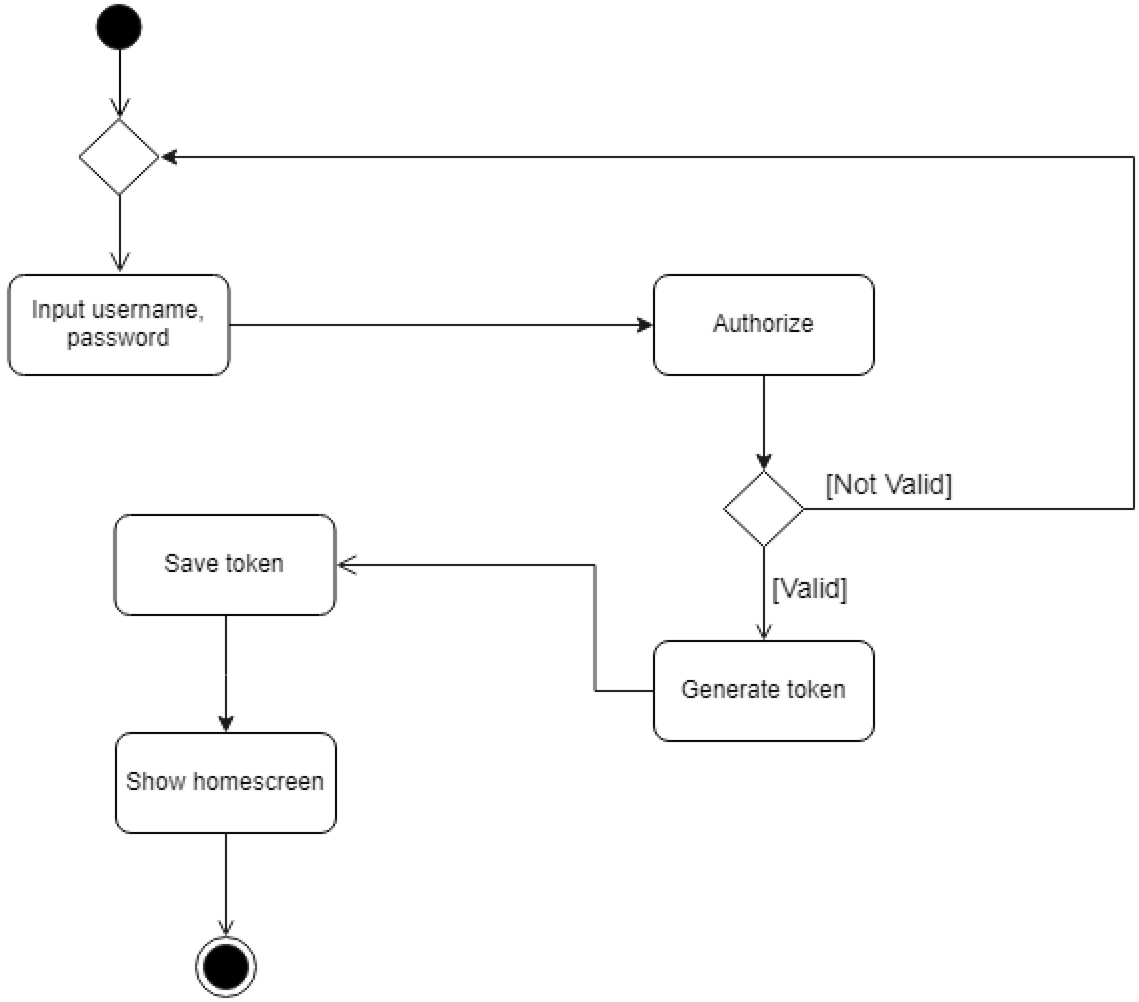

Figure 32 - <Activity> Log in

2.2. Register card

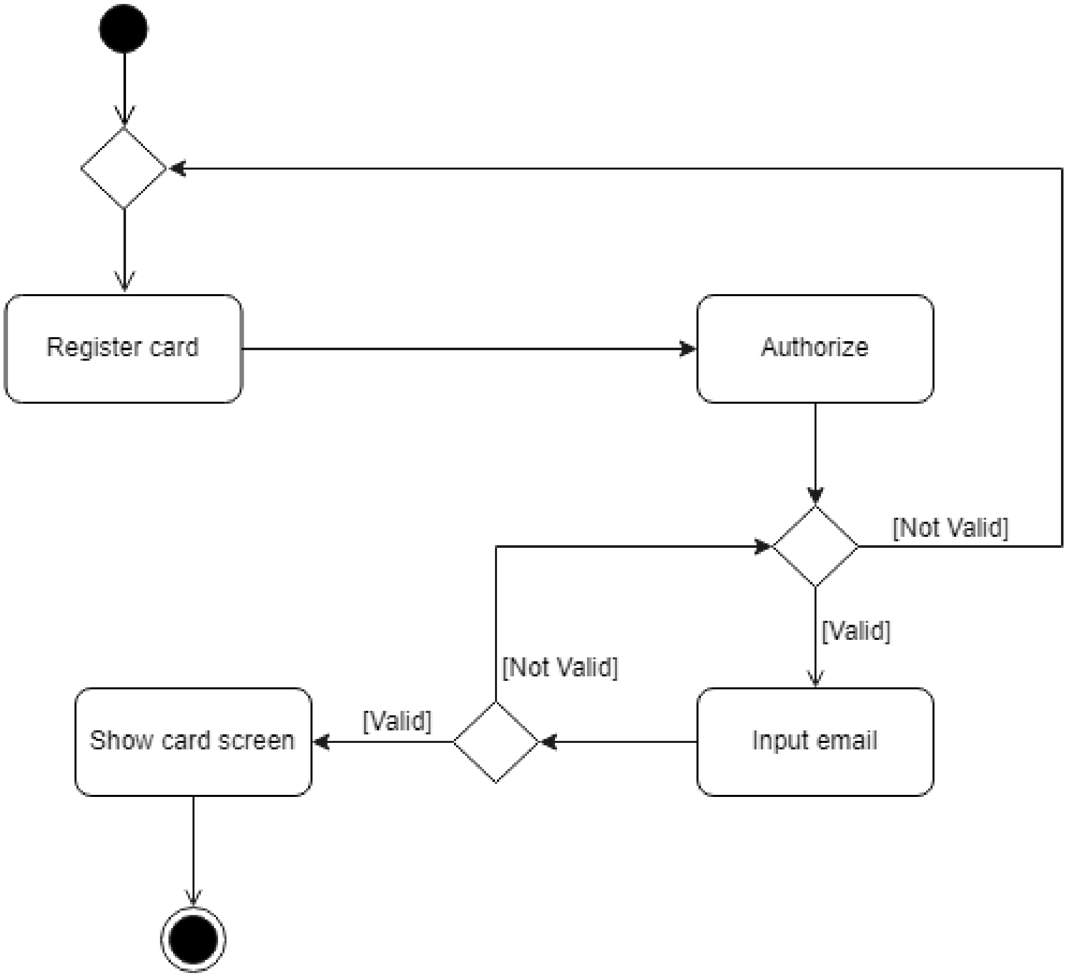

Figure 33 - <Activity> Register card

2.3. Book service

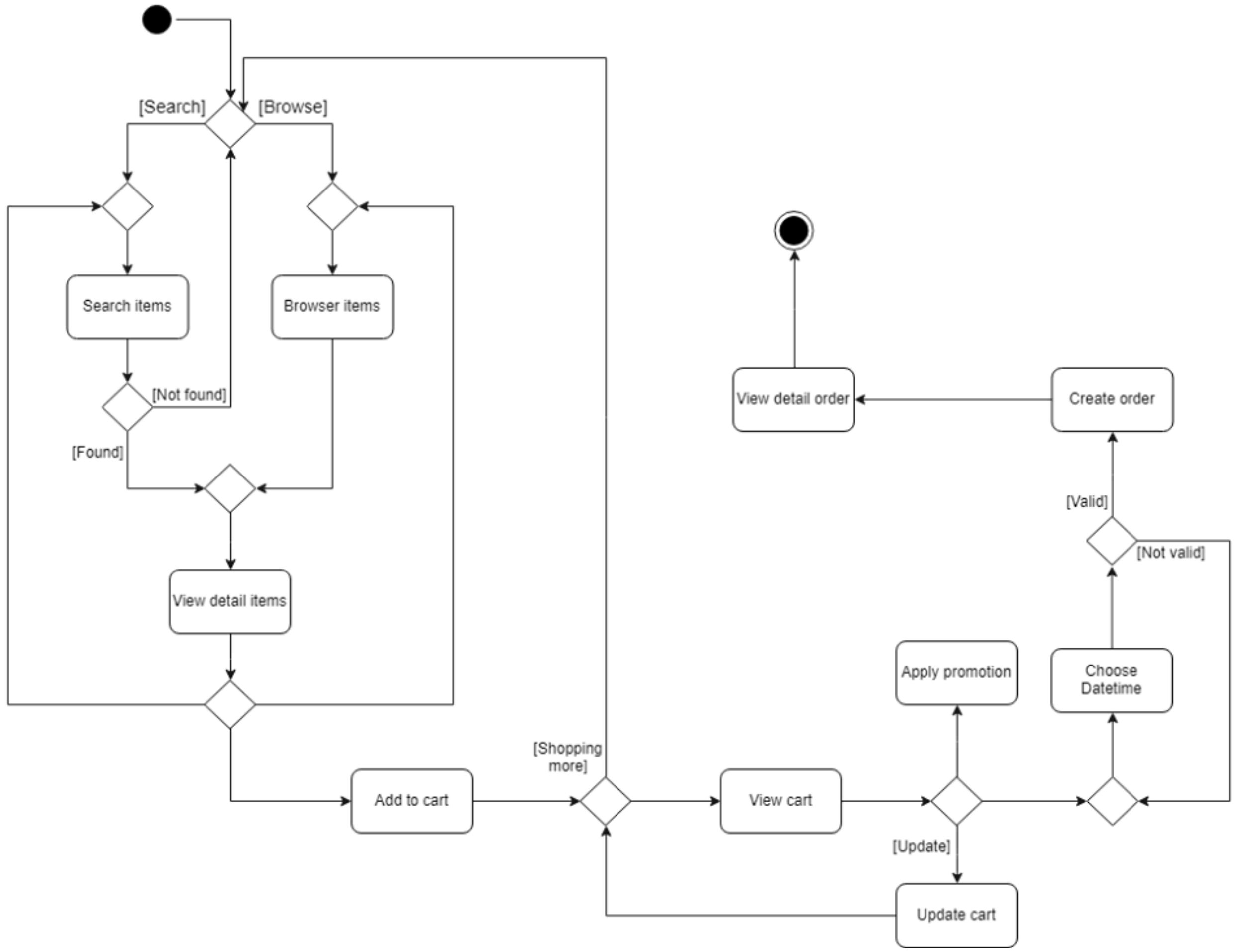

Figure 34 - <Activity> Book service

2.4. Buy product

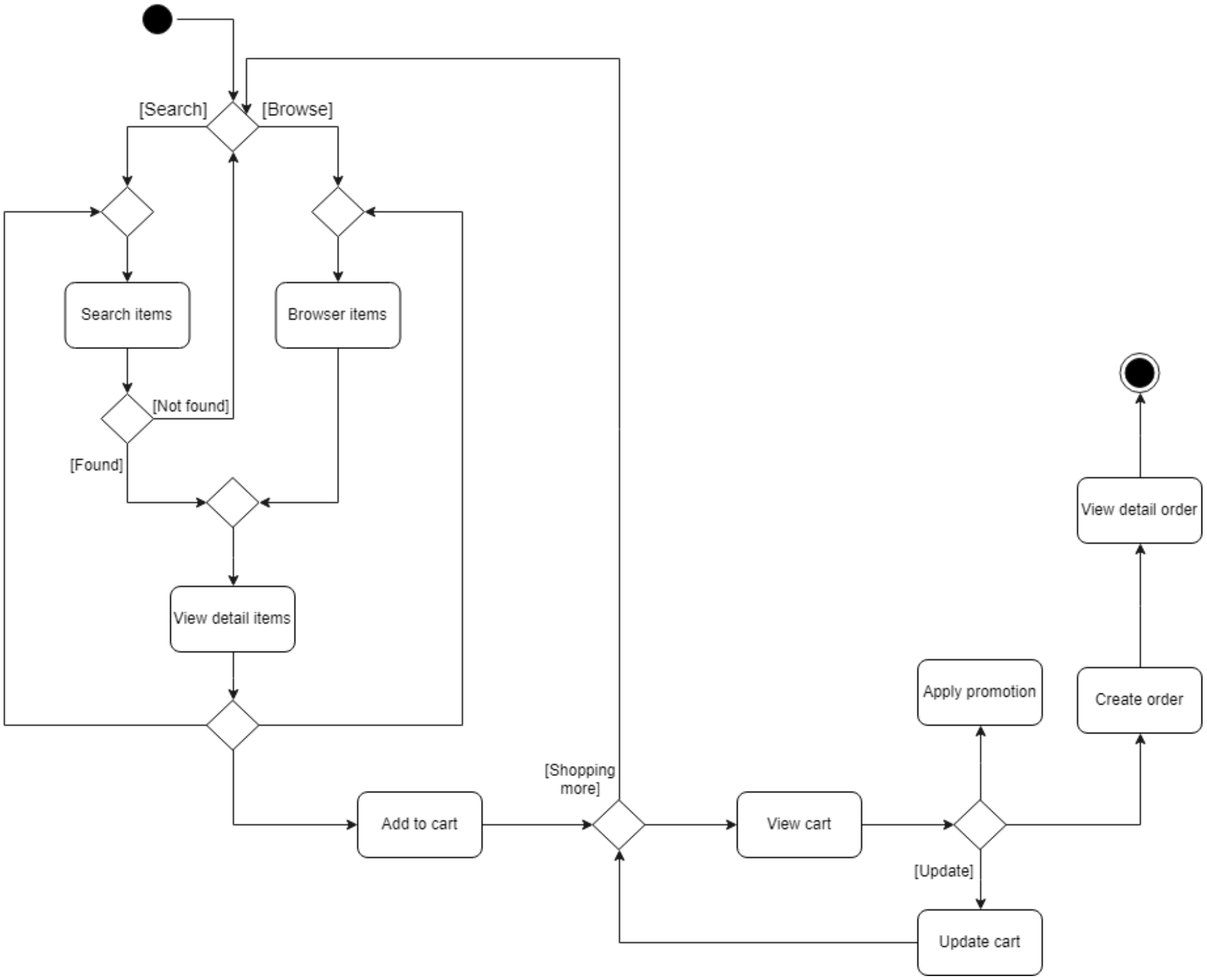

Figure 35 - <Activity> Buy product

2.5. Register Account

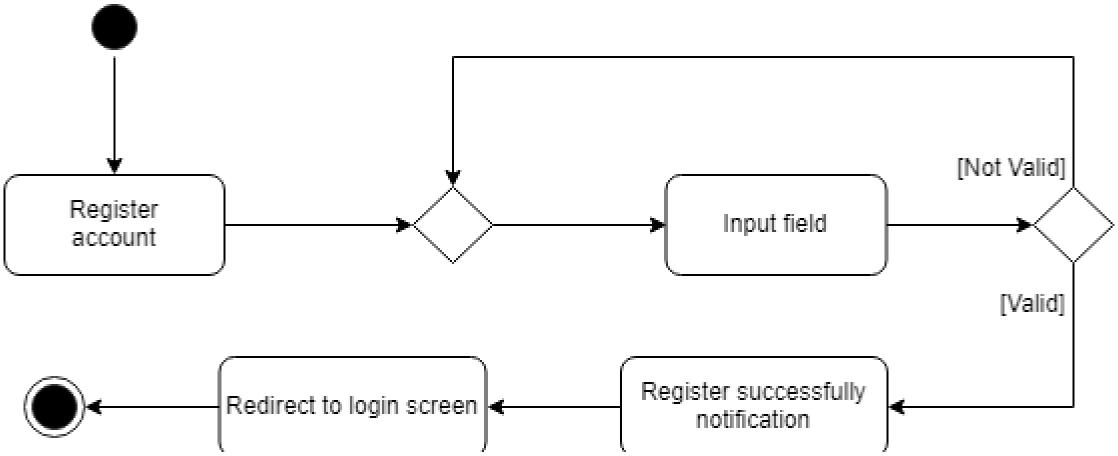

Figure 36 - <Activity> Register account

2.6. Payment order

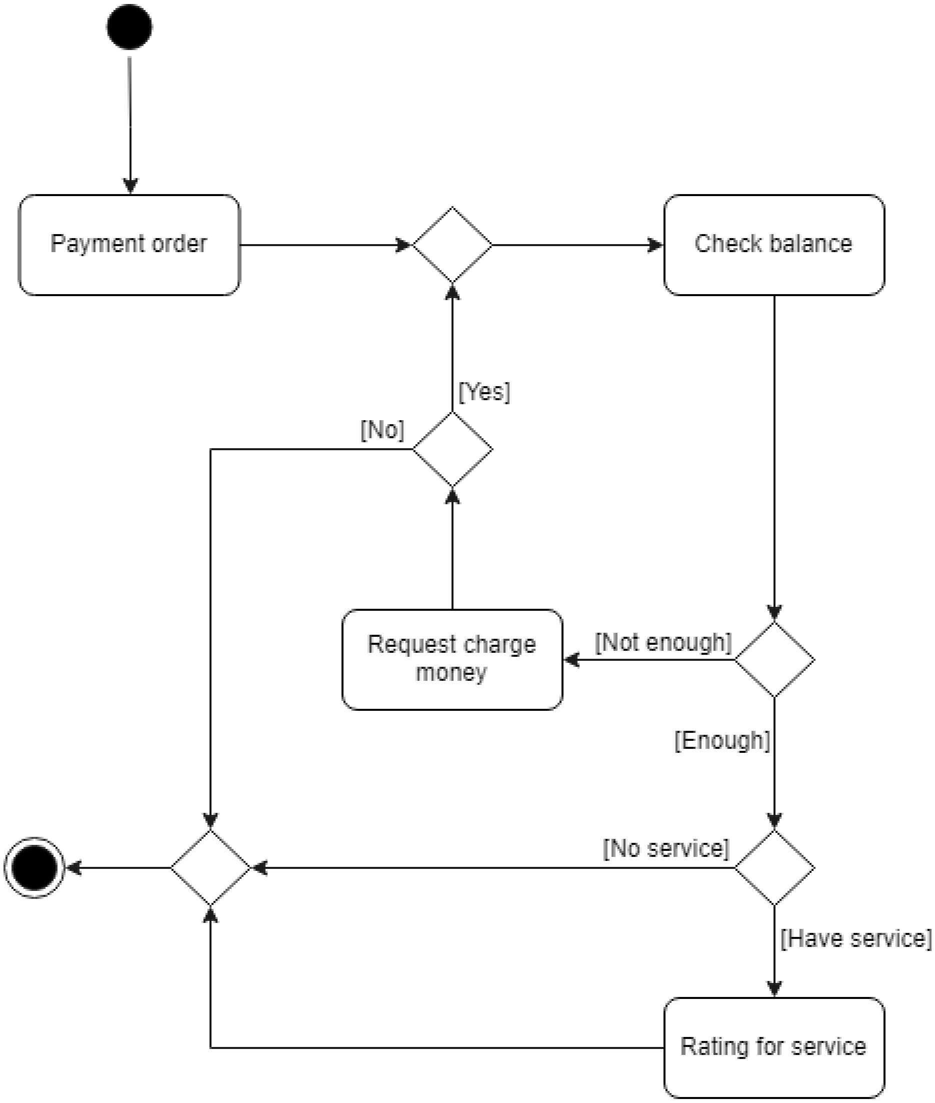

Figure 37 - <Activity> Payment order

L. Physical Diagram

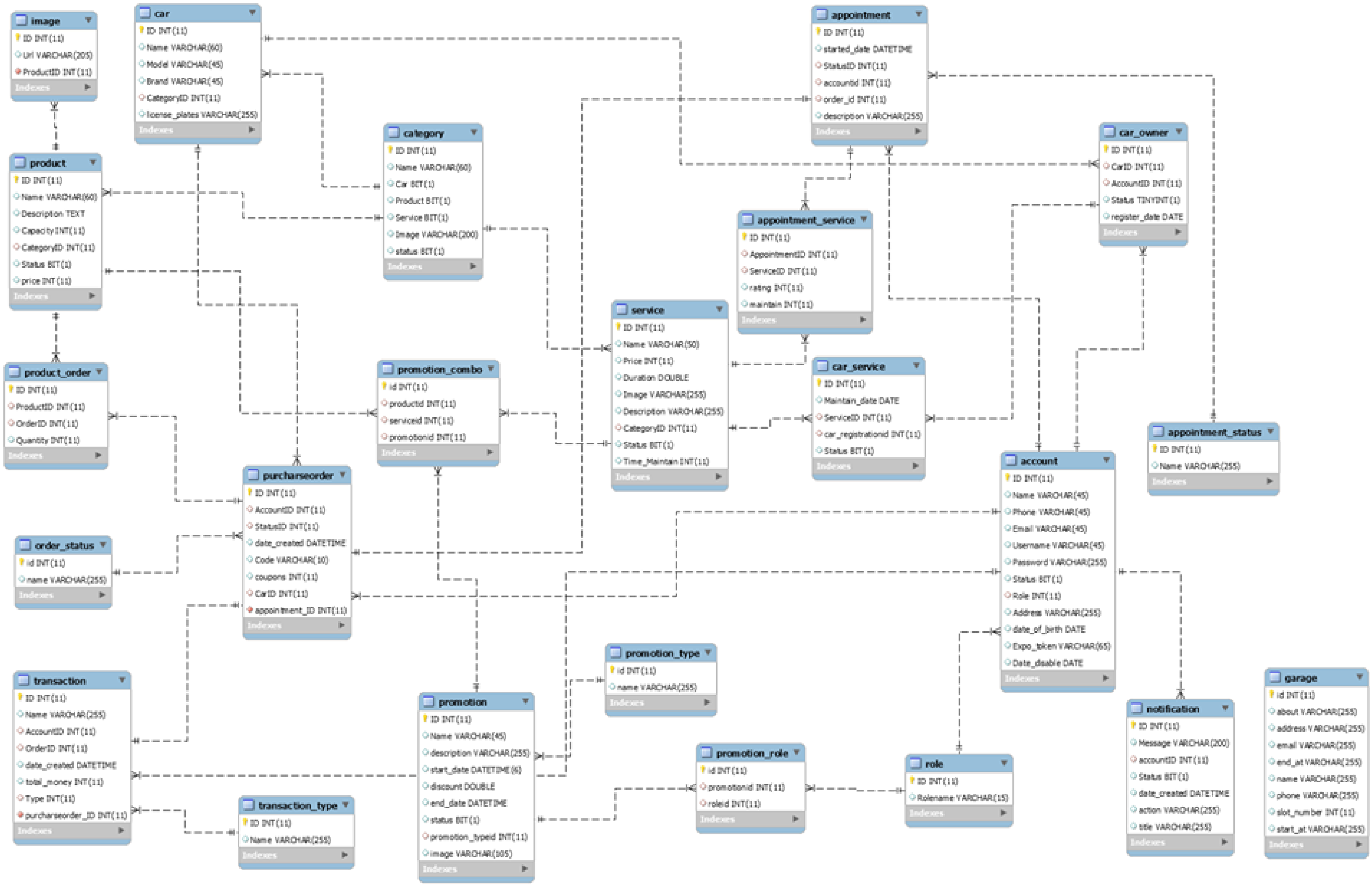

Figure 38 - Physical diagram

M. Framework's Architectural Diagram

1. React Js Architectural

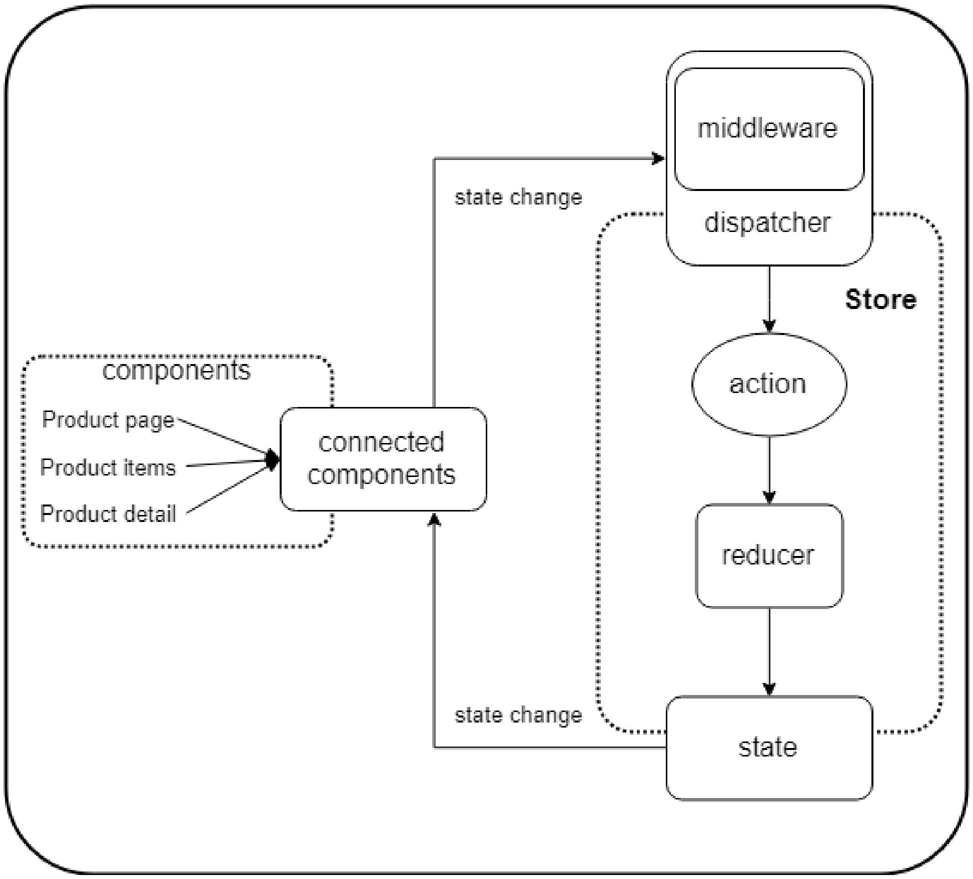

Figure 39 - React Js Architectural

2. Spring boot Architectural

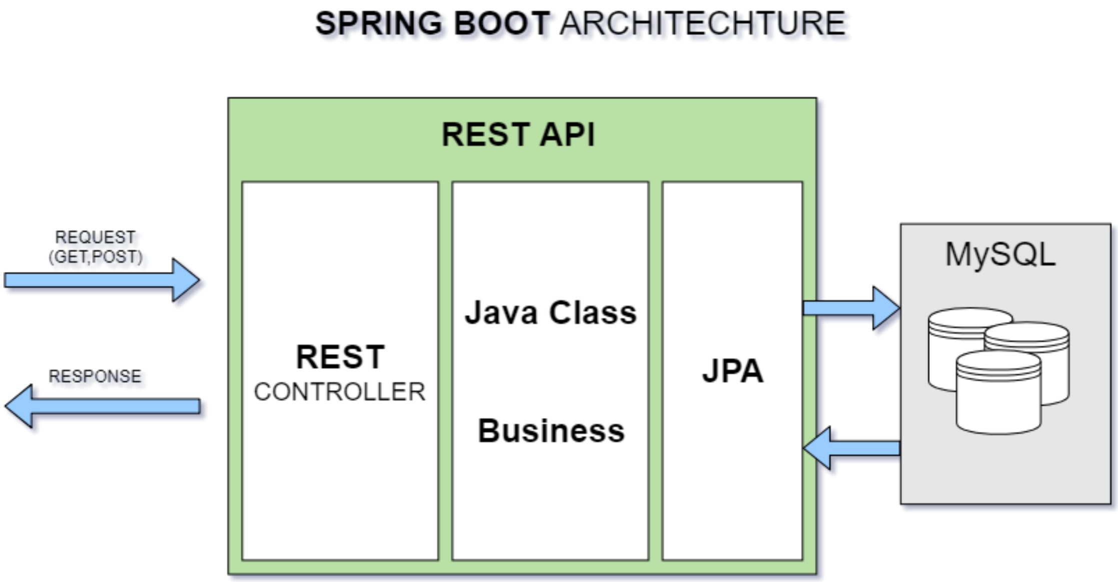

Figure 40 - Spring boot architectural

N. Future Plan

- Apply franchise system.
- Auto generate report for director.
- Stock management.
- Human management.
- Device management.
- Statistics of best-selling products.