

**ĐẠI HỌC BÁCH KHOA HÀ NỘI
TRƯỜNG CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG**



**TÀI LIỆU MÔ TẢ THIẾT KẾ PHẦN MỀM
(SOFTWARE DESIGN DESCRIPTION)**

HỆ THỐNG THUÊ XE ECOBIKE

Giảng viên hướng dẫn: **PGS. TS Nguyễn Thị Thu Trang**

Nhóm 1: **Hồ Anh - 20190037**

Nguyễn Trọng Bằng - 20190038

Hoàng Bá Công - 20190039

Hà Nội, 2023

Mục lục

1	Giới thiệu chung	3
1.1	Mục đích	3
1.2	Phạm vi	3
1.3	Từ điển thuật ngữ	3
1.4	Tham khảo	3
2	Requirement Analysis	4
2.1	Usecase Diagram	4
2.2	Usecase Specification	4
2.2.1	Usecase UC001 “View docking stations information”	4
2.2.2	Usecase UC002 “View bike information”	6
2.2.3	Usecase UC003 “Rent bike”	8
2.2.4	Usecase UC004 “Return bike”	10
2.2.5	Usecase UC005 “View rented bike information”	12
2.2.6	Usecase UC006 “Deposit”	13
3	Usecase Analysis	16
3.1	Interaction Diagram	16
3.1.1	Communication Diagram	16
3.1.2	Sequence Diagram	22
3.2	Analysis Class Diagram	26
4	Usecase Design	31
4.1	Interface Design - Interbank System	31
4.1.1	Analysis Class Diagram	31
4.1.2	Sequence Diagram	33
4.2	Class Design	35
4.2.1	General Design	35
4.2.2	Relational Class	36
4.2.3	Relational Exception	36

4.3	Graphical User Interface (GUI)	37
4.4	Data Modeling	42
4.4.1	E-R Diagram	42
4.4.2	Database Diagram	43
4.4.3	Mô tả Database	43
5	Nguyên tắc thiết kế	46
5.1	Mục tiêu	46
5.2	Chiến lược kiến trúc	46
5.3	Coupling và Cohesion	47
5.3.1	Phân tích tính Cohesion cho các lớp Controller	47
5.3.2	Phân tích tính coupling giữa các lớp Controller	47
5.4	Nguyên tắc thiết kế	47
5.4.1	Single Responsibility	47
5.4.2	Open/Closed	47
5.4.3	Liskov Substitution	48
5.4.4	Interface Segregation	48
5.5	Design Pattern	48

Chương 1

Giới thiệu chung

1.1 Mục đích

Tài liệu này đưa ra mô tả chi tiết về thiết kế kiến trúc, thiết kế giao diện, thiết kế cơ sở dữ liệu cũng như thiết kế chi tiết lớp cho từng chức năng và các thành phần trong hệ thống. Tài liệu giúp cho người lập trình viên, testers, maintainers, ... có cái nhìn cụ thể chi tiết về phần mềm để dễ dàng trong quá trình xây dựng. Tài liệu dành cho các bên liên quan (stakeholder) và các nhà phát triển phần mềm.

1.2 Phạm vi

Hệ thống thuê xe điện tử được cài đặt trên hệ thống máy tính của các bến xe trong thành phố, giúp khách hàng dễ dàng tìm kiếm bến xe, tìm kiếm xe trong bến và thanh toán chính xác. Hệ thống tương tác với API của các ngân hàng được liên kết với hệ thống.

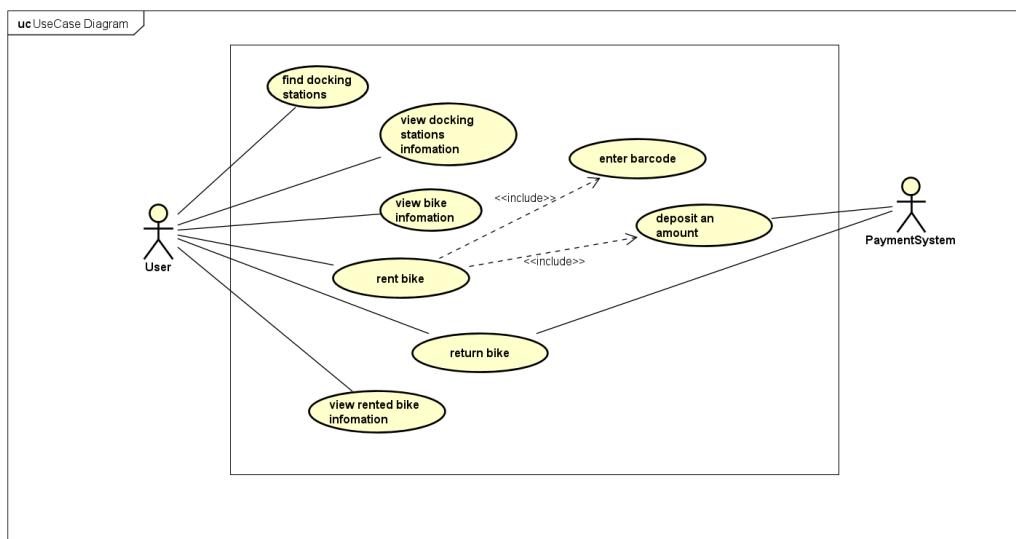
1.3 Từ điển thuật ngữ

1.4 Tham khảo

Chương 2

Requirement Analysis

2.1 Usecase Diagram



Hình 2.1: UseCase Diagram

2.2 Usecase Specification

2.2.1 Usecase UC001 “View docking stations information”

1. Usecase code
UC001

2. Introduction

Use case describes the function which allows client to see details of the docking stations

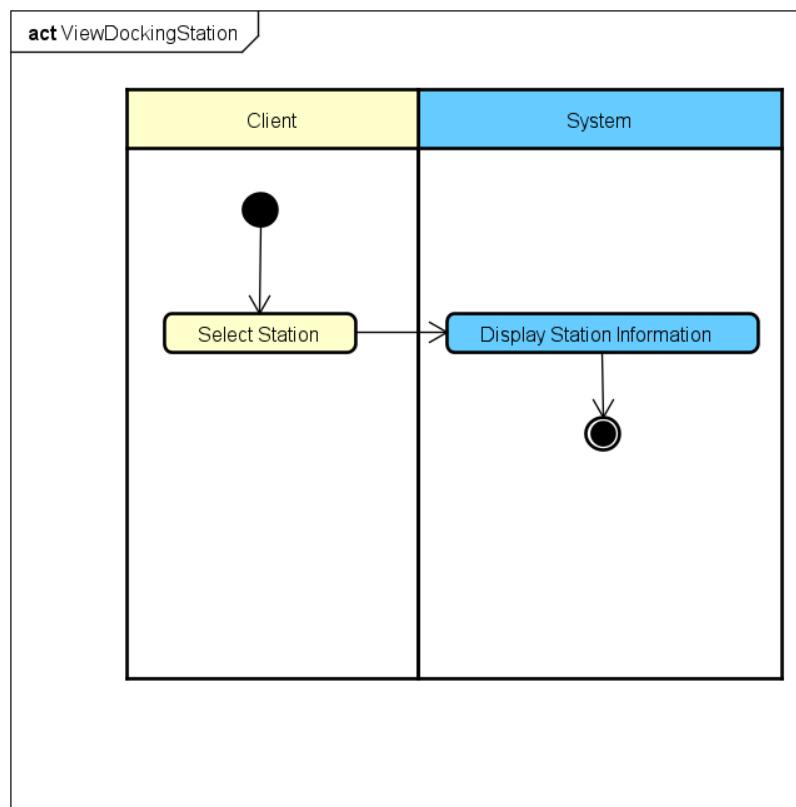
3. Actor

Client

4. Pre-condition**5. Basic flow of main scenario**

(a) Client selects docking station

(b) System displays details of selected docking station

6. Extension of alternate flows**7. Activity diagram**

Hình 2.2: View docking stations information

8. Input

9. Output

No.	Field	Description	Format	Example
1	Name	Name of station		Giai Phong Station
2	Address	Location of station		123 Giai Phong, Hai Ba Trung, Hanoi
3	Distance	Distance from client to station (km)	Non-negative real number (km)	4.9 km
4	Area	Area of station (km ²)	Non-negative real number (km ²)	1 km ²
5	Number of current vehicle	Total number of vehicles of all types in station	Non-negative integer	490
6	Vacant position of each vehicle type			Bike: 1A, 2B
7	Time	Time to get to station	Positive integer (minute)	15 minutes

10. Post condition

2.2.2 Usecase UC002 “View bike information”

1. Usecase code

UC002

2. Introduction

Use case describes the function which allows client to see details of the bike.

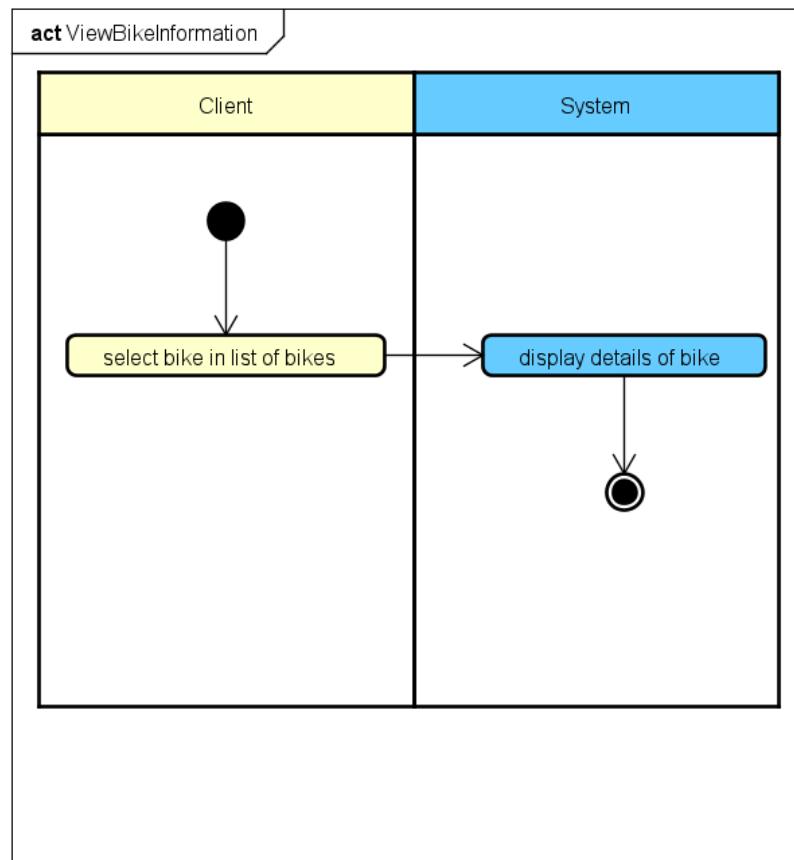
3. Actor

4. Pre-condition

Screen displaying list of bikes in station

5. Basic flow of main scenario

- (a) Client selects bike
 - (b) System displays details of selected bike
6. Extension of alternate flows
7. Activity diagram



Hình 2.3: View bike information

8. Input

9. Output

No.	Field	Description	Format	Example
1	License plate			37A46543
2	Bike value		Non-negative integer (VND)	5,000,000VND
3	Remaining battery		Non-negative integer (%)	60%

10. Post condition

2.2.3 Usecase UC003 “Rent bike”

1. Usecase code

UC003

2. Introduction

Use case describes the interaction between the client and the system when the client wants to rent a bike.

3. Actor

Client

4. Pre-condition

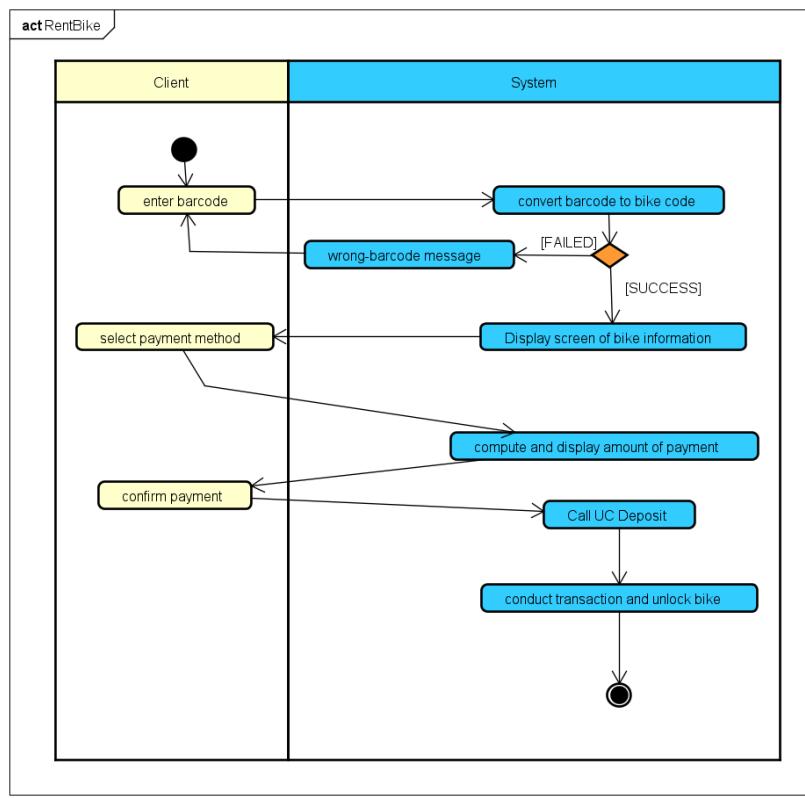
5. Basic flow of main scenario

- (a) Client enters barcode
- (b) System converts barcode to bike code
- (c) Display details of bike
- (d) Client selects payment method
- (e) System computes and displays amount to be paid
- (f) Client confirms amount
- (g) Call usecase ‘Deposit’
- (h) Unlock bike

6. Extension of alternate flows

No.	Step	Condition	Action	Next step
1	Step 2	Incorrect barcode	Display message “Barcode is not correct”	Step 1

7. Activity diagram



Hình 2.4: Rent bike

8. Input

No.	Field	Data type	Obligatory?	Valid condition	Example
1	Barcode	int	Yes	Exist in list of barcode	012346543

9. Output

No.	Field	Description	Format	Example
1	Client name			Nguyen Van A
2	ID Card			BI123
3	Expiration date			07/04/2024
4	Bike name			Mountain Bike MTB GIANT Talon 29 3
5	Deposit		Non-negative integer (VND)	400,000 VND
6	Station to get bike			Giai Phong Station
7	Time to get bike			9:00 a.m 15/12/2022

10. Post condition

2.2.4 Usecase UC004 “Return bike”

1. Usecase code

UC004

2. Introduction

Use case describes the interaction between the client and the system when the client wants to return a bike.

3. Actor

Client

4. Pre-condition

Client rented a bike.

5. Basic flow of main scenario

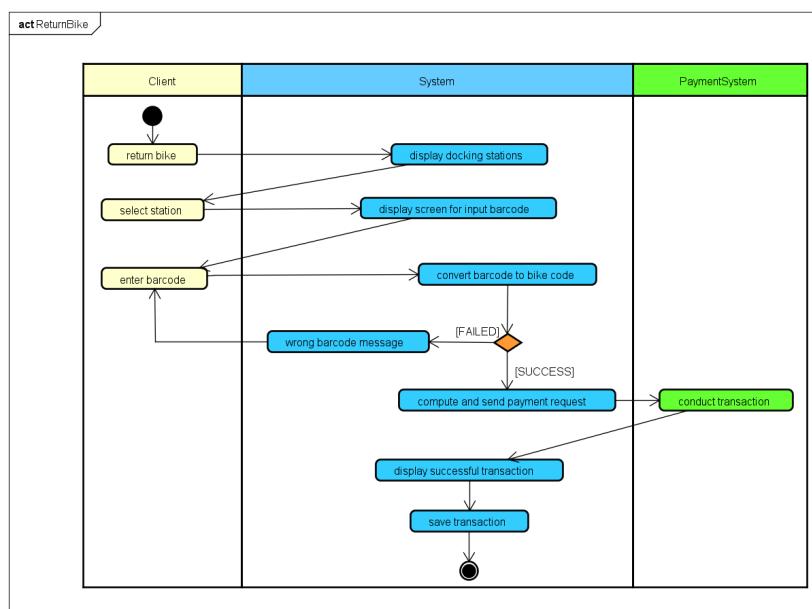
- (a) Client selects ‘return bike’
- (b) System displays list of docking stations
- (c) Client selects docking station
- (d) System displays screen to enter barcode
- (e) Client enter barcode
- (f) System convert barcode to bike code

- (g) The system calculates the cost and sends the request to the payment system
- (h) The payment system executes the transaction
- (i) Show successful car return
- (j) System record transaction

6. Extension of alternate flows

No.	Step	Condition	Action	Next step
1	Step 6	Incorrect barcode	Displays the message “barcode is incorrect”	Step 5

7. Activity diagram



Hình 2.5: Return bike

8. Input

No.	Field	Description	Data type	Obligatory?	Valid condition	Example
1.	Barcode		int	YES	Length less than 255	123004567

9. Output

No.	Field	Description	Display format	Example
1	Client name			Nguyen Van A
2	ID Card			KH12001
3	Expiration date			22/2/2023
4	Bike name			Mountain Bike MTB GIANT Talon 29 3
5	Deposit		Non-negative integer (VND)	2,000,000 VND
6	Rental cost		Non-negative integer (VND)	400,000 VND
7	Station to get bike			Giai Phong Station
8	Station to return bike			Phuong Mai Station

10. Post condition

2.2.5 Usecase UC005 “View rented bike information”

1. Usecase code

UC005

2. Introduction

Use case describes the interaction between the client and the system when he wants to see the rented bike information.

3. Actor

Client

4. Pre-condition

Client rented a bike.

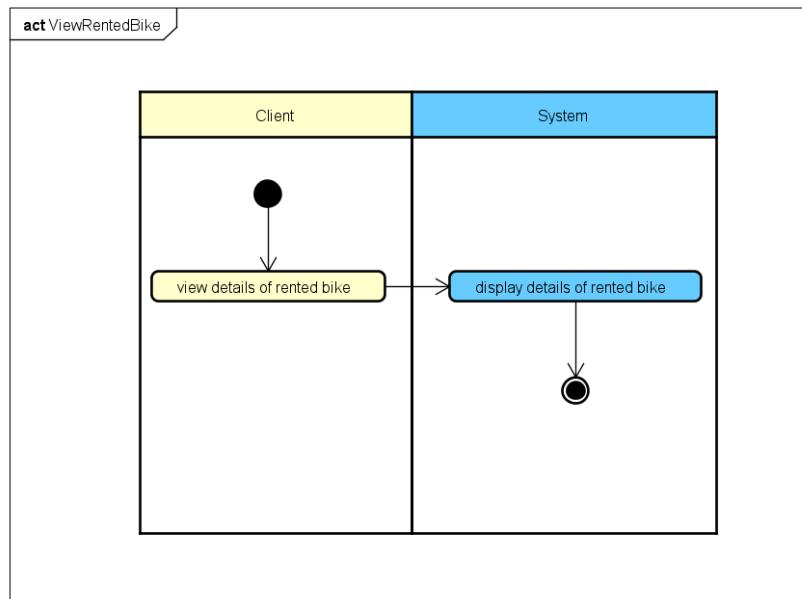
5. Basic flow of main scenario

(a) Client select ‘view details of rented bike’

(b) Display details of rented bike

6. Extension of alternate flows

7. Activity diagram



Hình 2.6: View rented bike information

8. Input

9. Output

No.	Field	Description	Format	Example
1	License plate			37A46543
2	Bike value		Non-negative integer (VND)	5,000,000VND
3	Remaining battery		Non-negative integer (%)	60%

10. Post condition

2.2.6 Usecase UC006 “Deposit”

1. Usecase code

UC006

2. Introduction

The use case describes the interaction between the client and the payment system and the system when making a deposit.

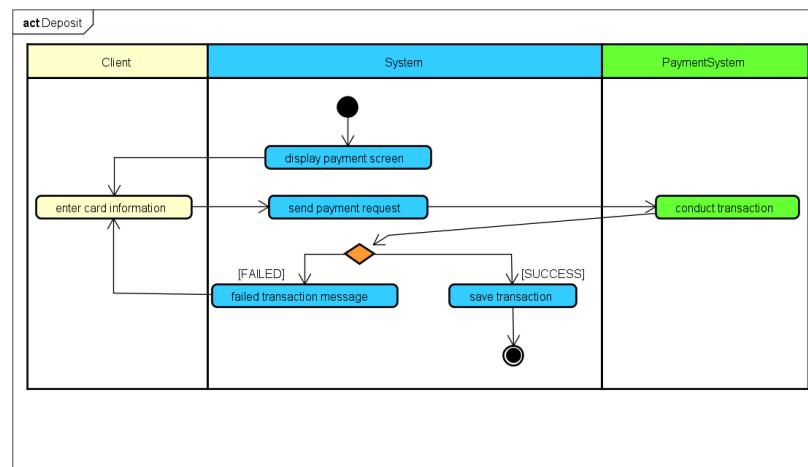
3. Actor

- (a) Client
 - (b) Payment System
4. Pre-condition
Client confirms the deposit
5. Basic flow of main scenario
- (a) System displays payment screen
 - (b) Client enter payment information
 - (c) System sends payment request to payment system
 - (d) The payment system executes the transaction
 - (e) System save the transaction

6. Extension of alternate flows

No.	Step	Condition	Action	Next step
1	Step 5	Incorrect card information	The system displays failed transaction "Incorrect card information"	Step 2
2	Step 5	Unavailable amount	The system displays failed transaction "Insufficient funds available in the account"	Step 2

7. Activity diagram



Hình 2.7: Deposit

8. Input

No.	Field	Data type	Obligatory?	Valid condition	Example
1.	Account name	String	YES	Length does not exceed 255	Nguyen Van A
2.	ID Card	String	YES	Card code exists	0123764521123
3.	Bank	String	YES	Bank linked to the system	BIDV
4.	Expiration date	Date	YES		20/10/2030
5.	Security code	String	YES		0x23cd223
6.	Transaction content	String	YES		

9. Output

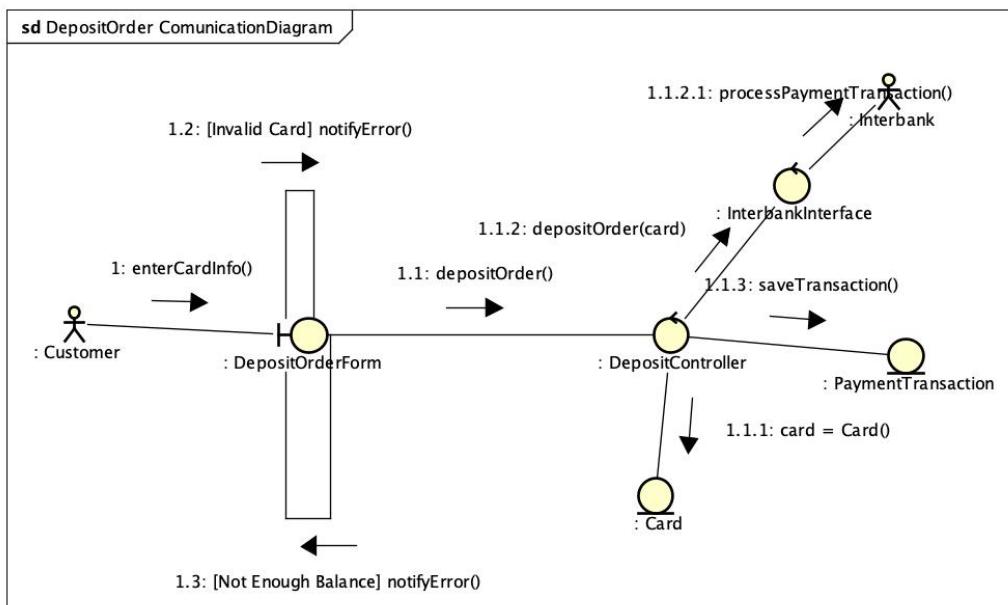
10. Post condition

Chương 3

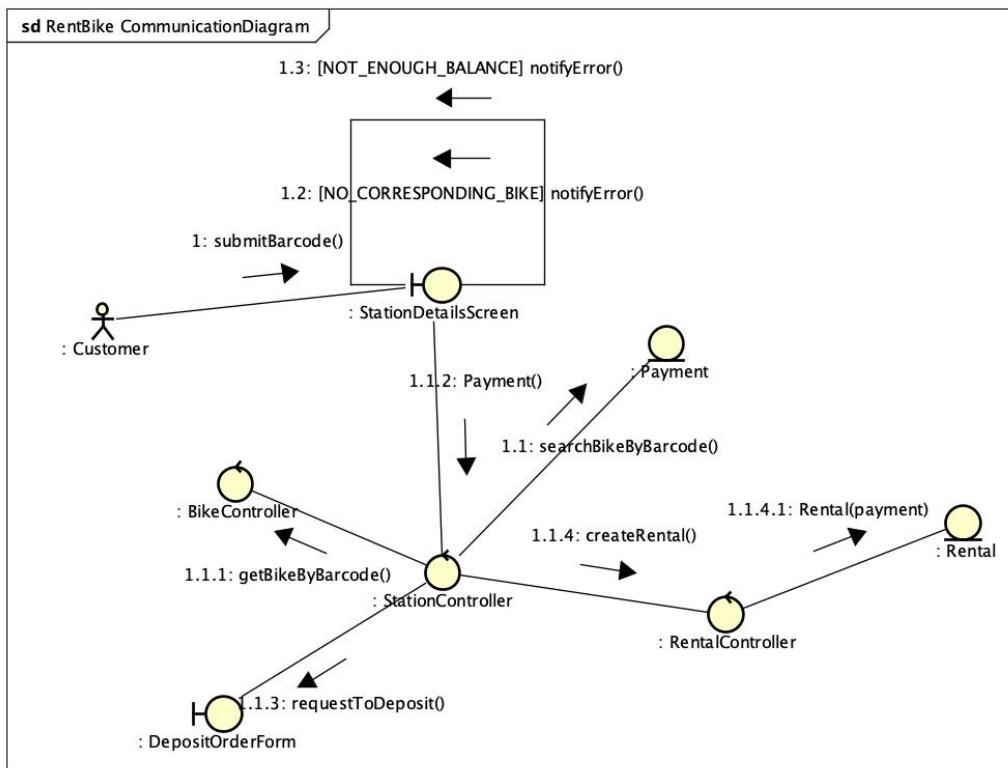
Usecase Analysis

3.1 Interaction Diagram

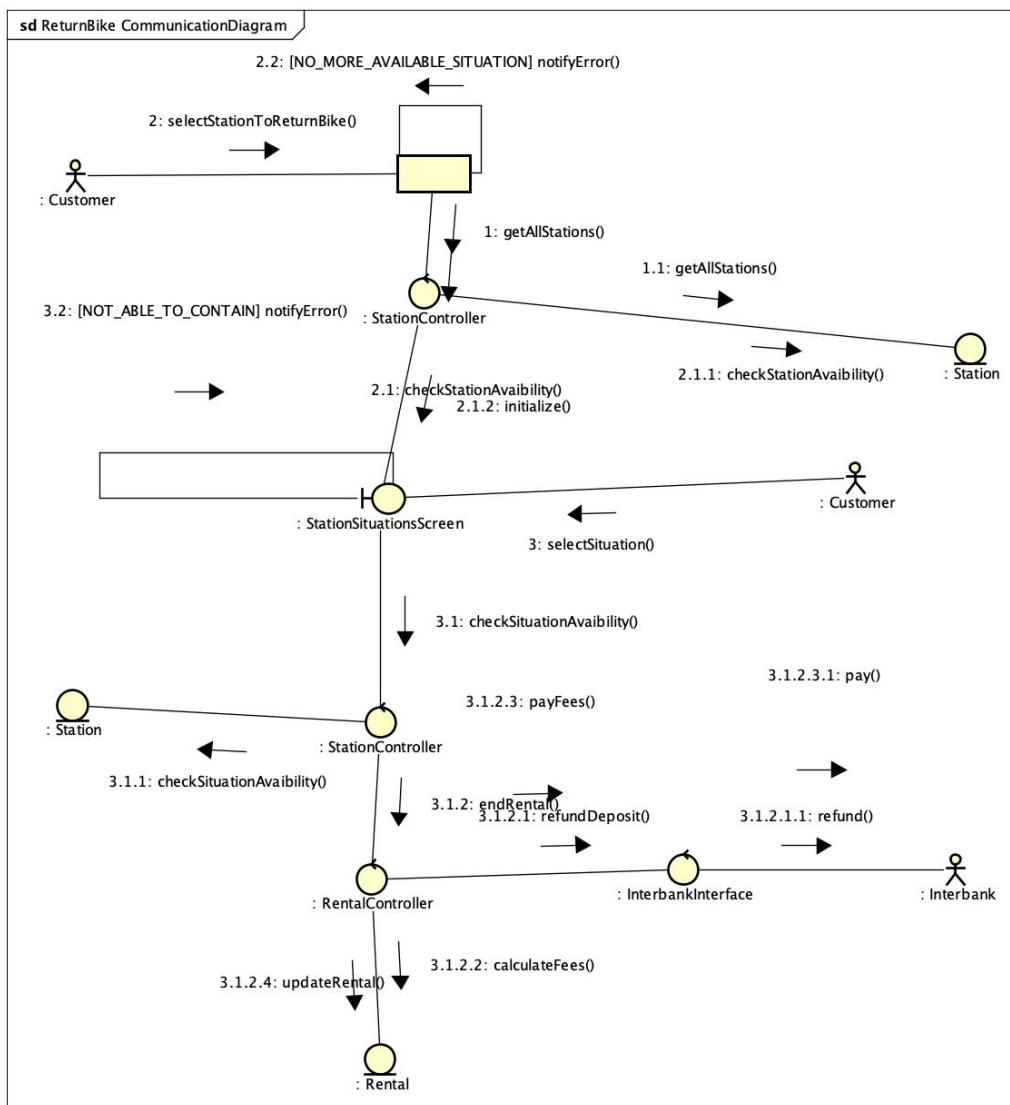
3.1.1 Communication Diagram



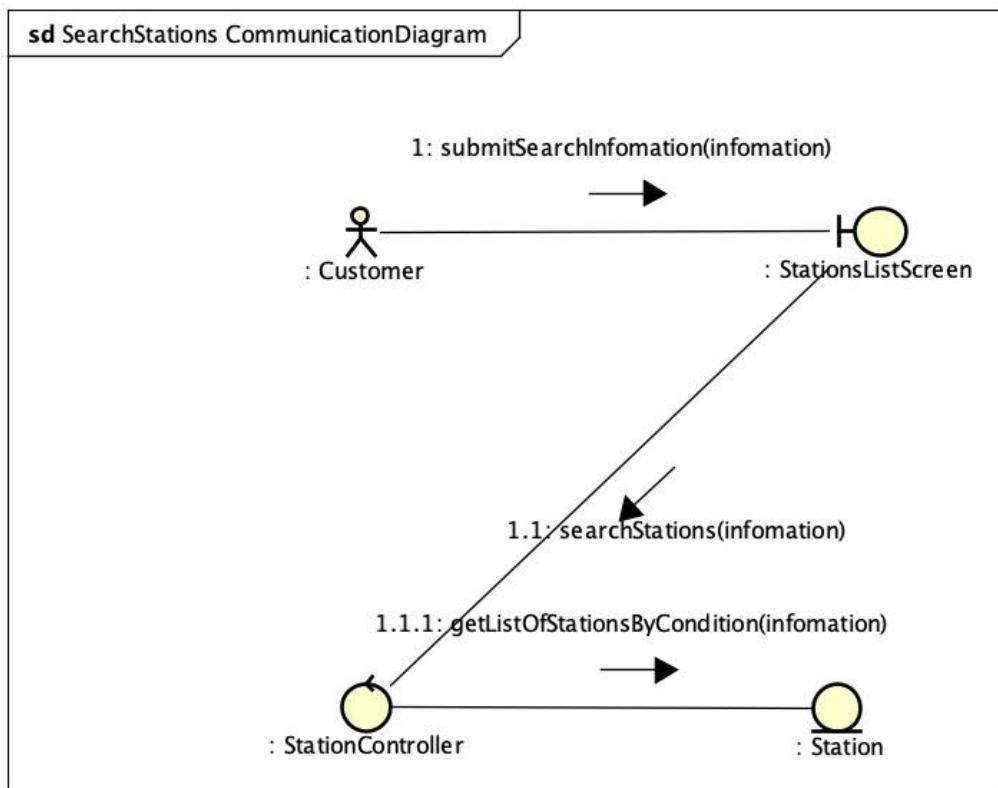
Hình 3.1: Deposit Order Comunication Diagram



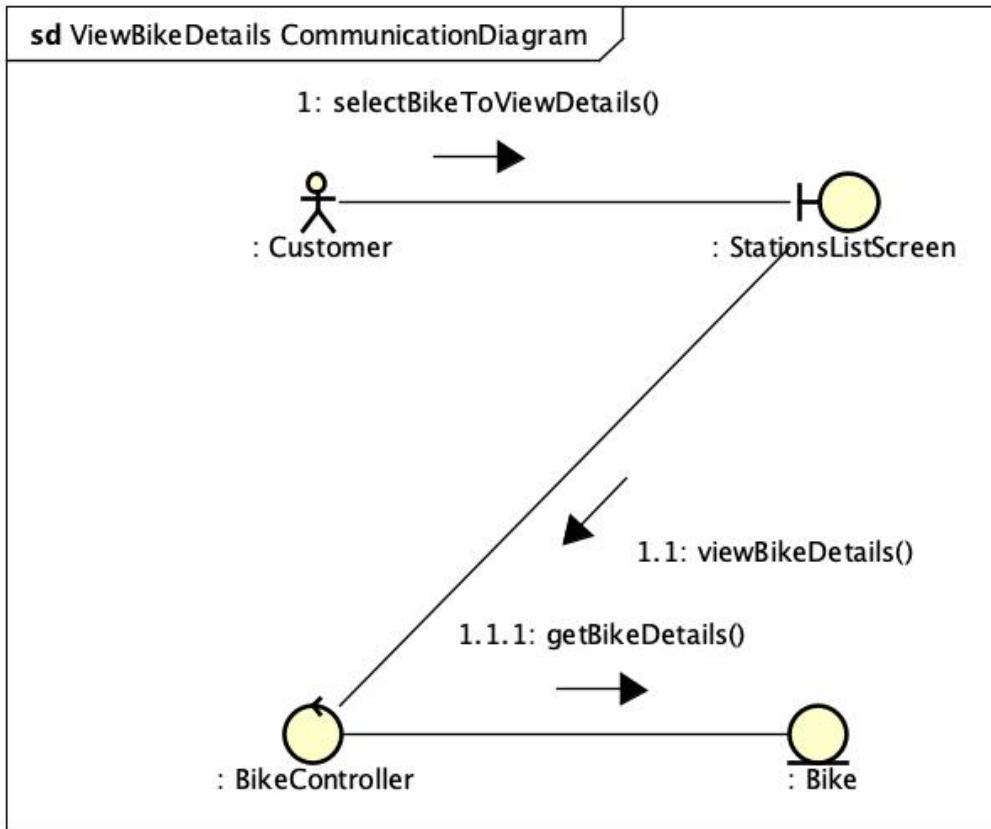
Hình 3.2: Rent Bike Communication Diagram



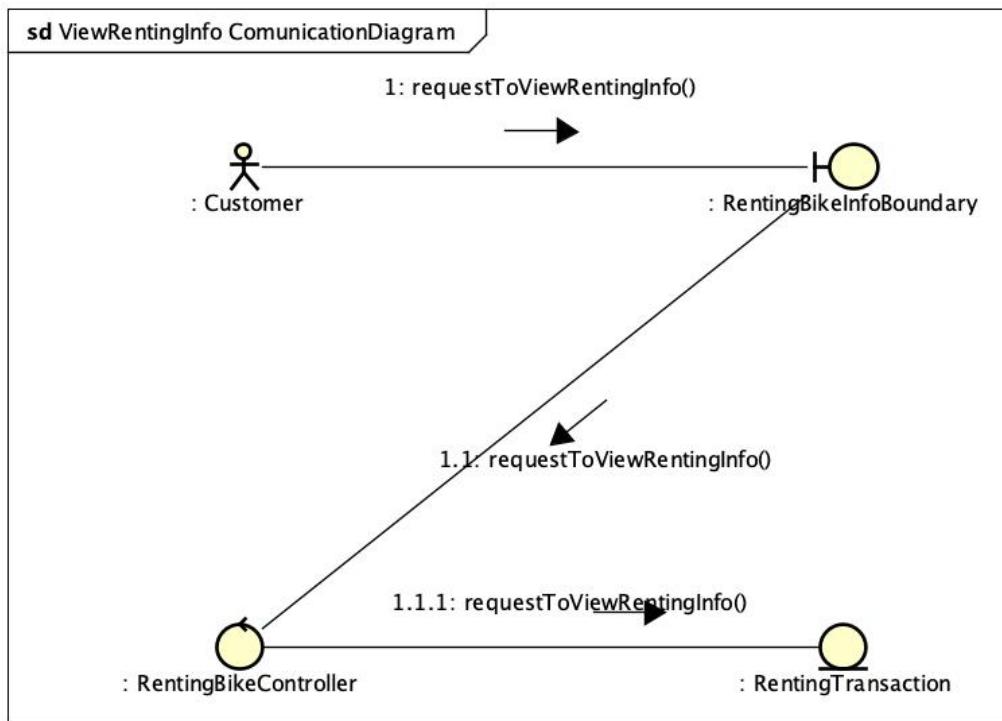
Hình 3.3: Return Bike Communication Diagram



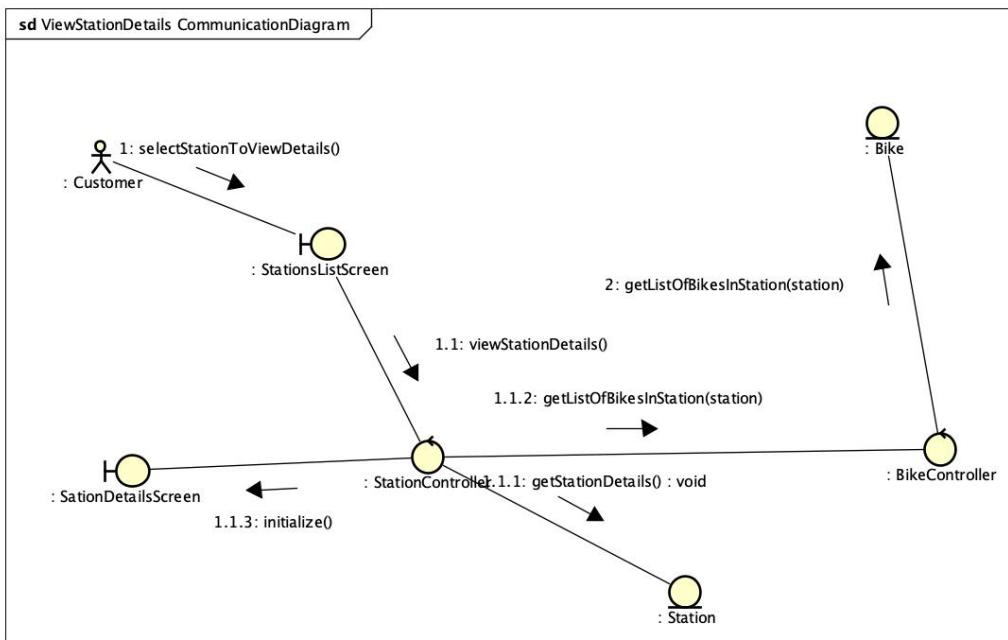
Hình 3.4: Search Stations Communication Diagram



Hình 3.5: View Bike Details Communication Diagram

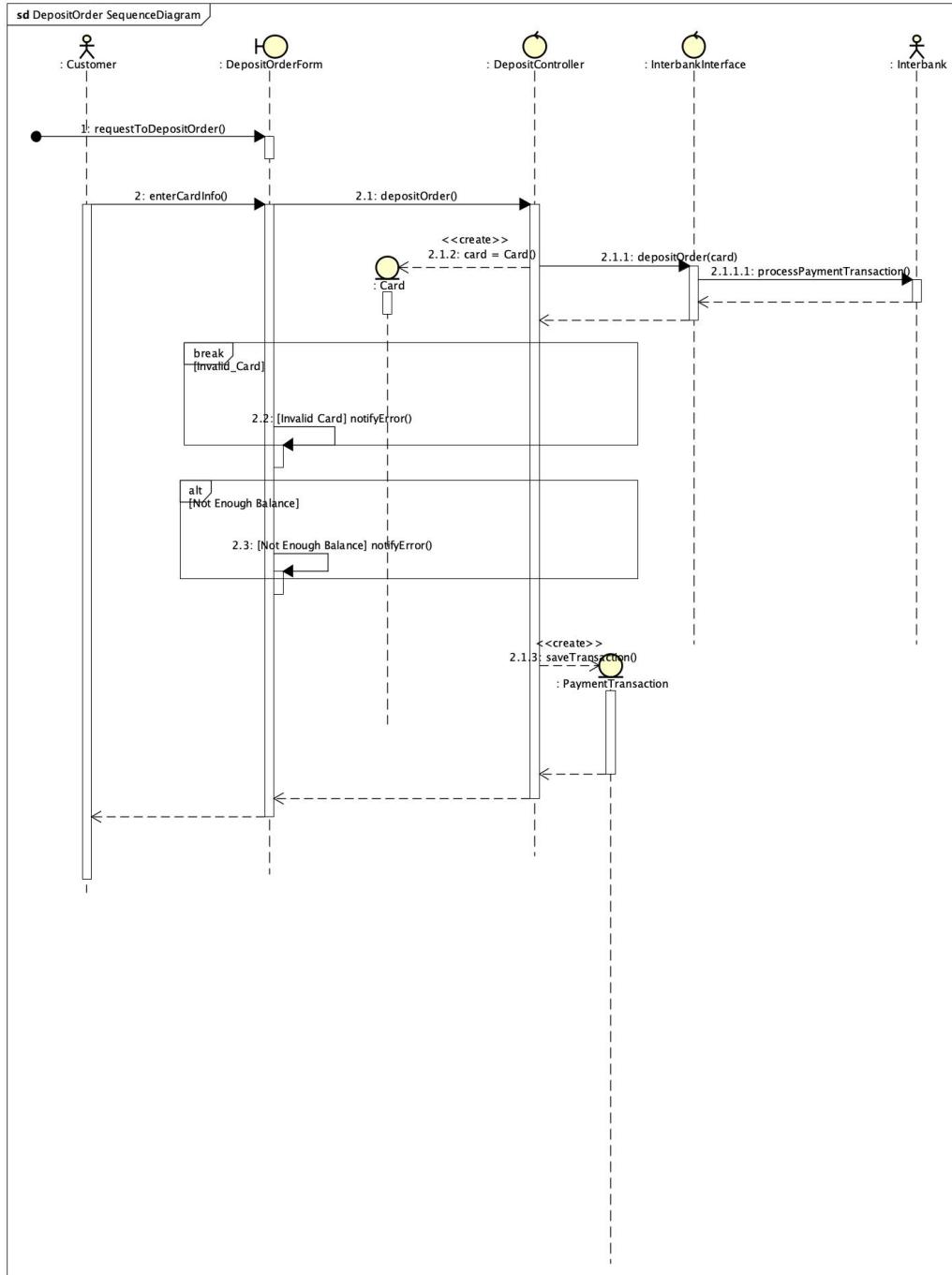


Hình 3.6: View Renting Info Communication Diagram

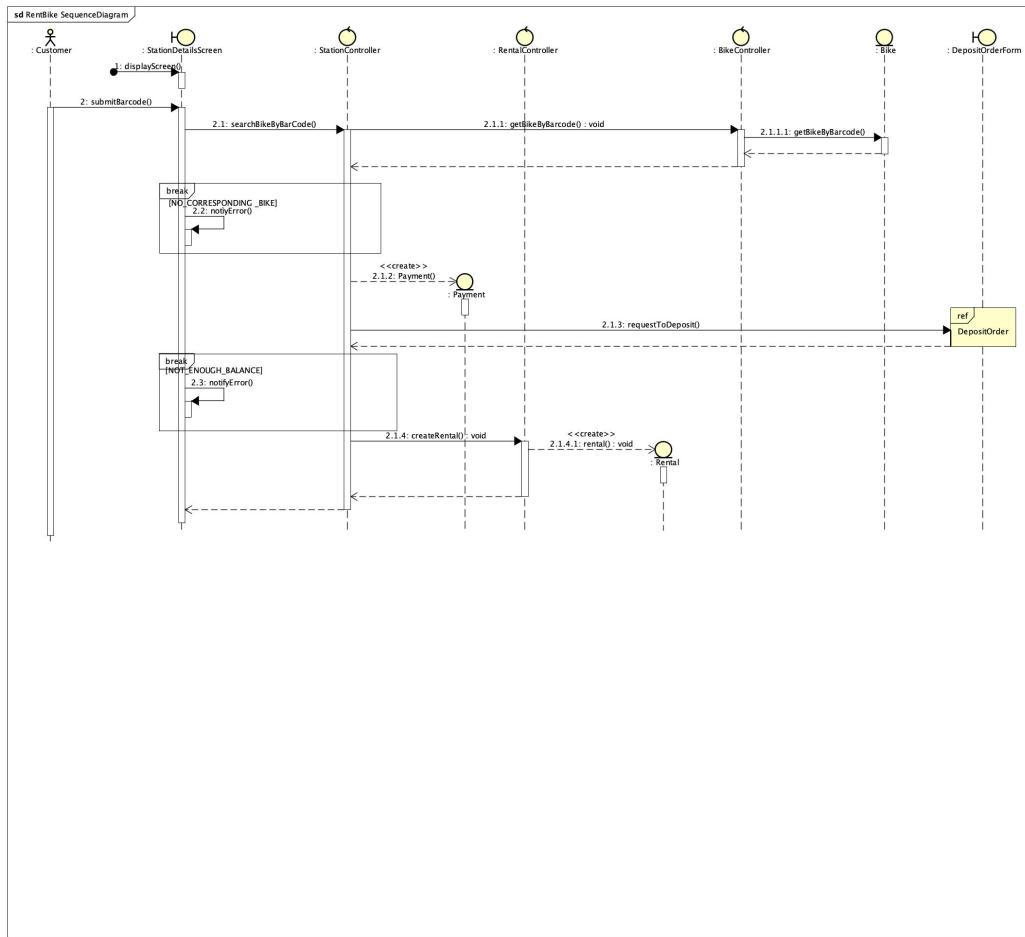


Hình 3.7: View Station Details Communication Diagram

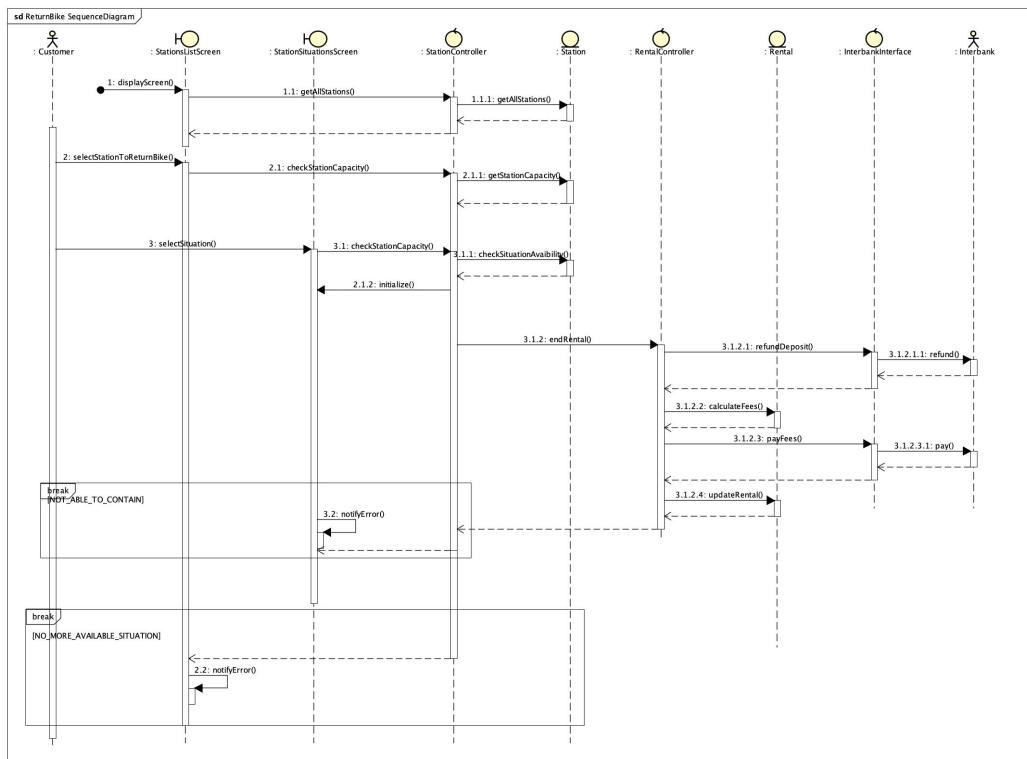
3.1.2 Sequence Diagram



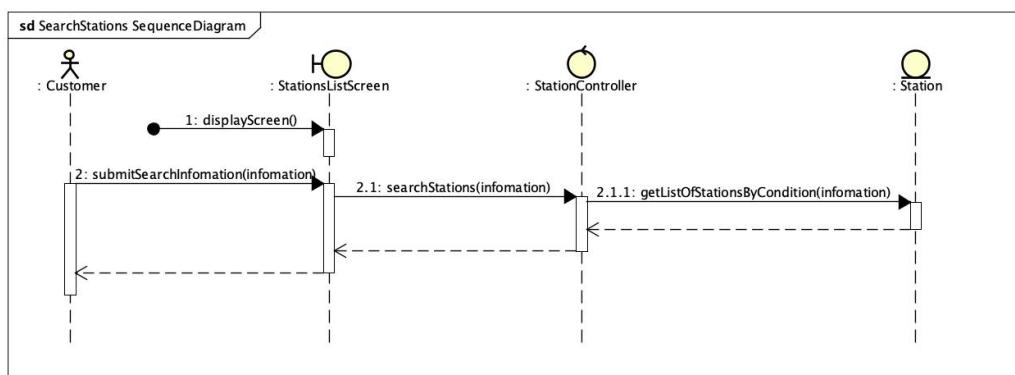
Hình 3.8: Deposit Order Sequence Diagram



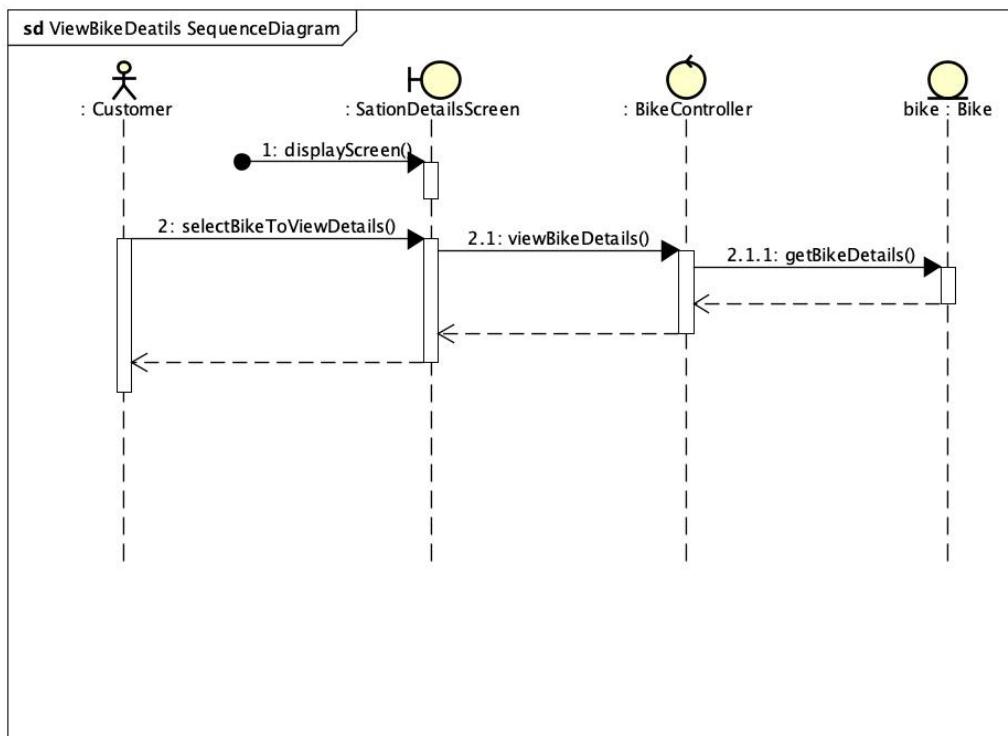
Hình 3.9: Rent Bike Sequence Diagram



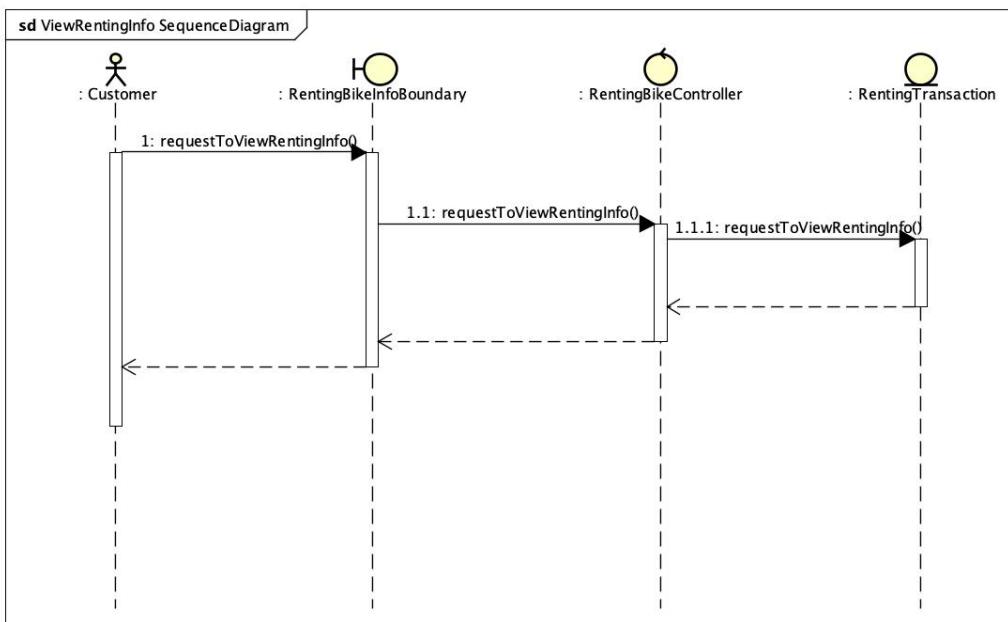
Hình 3.10: Return Bike Sequence Diagram



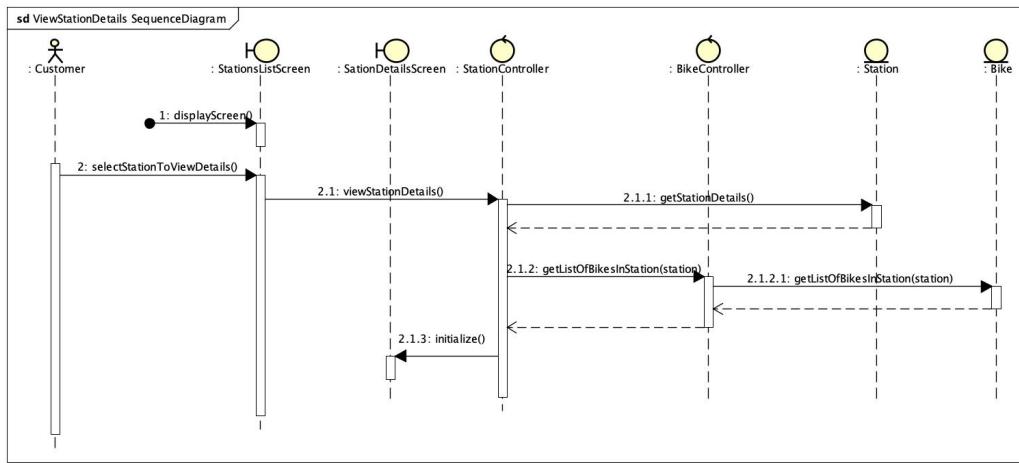
Hình 3.11: Search Stations Sequence Diagram



Hình 3.12: View Bike Details Sequence Diagram

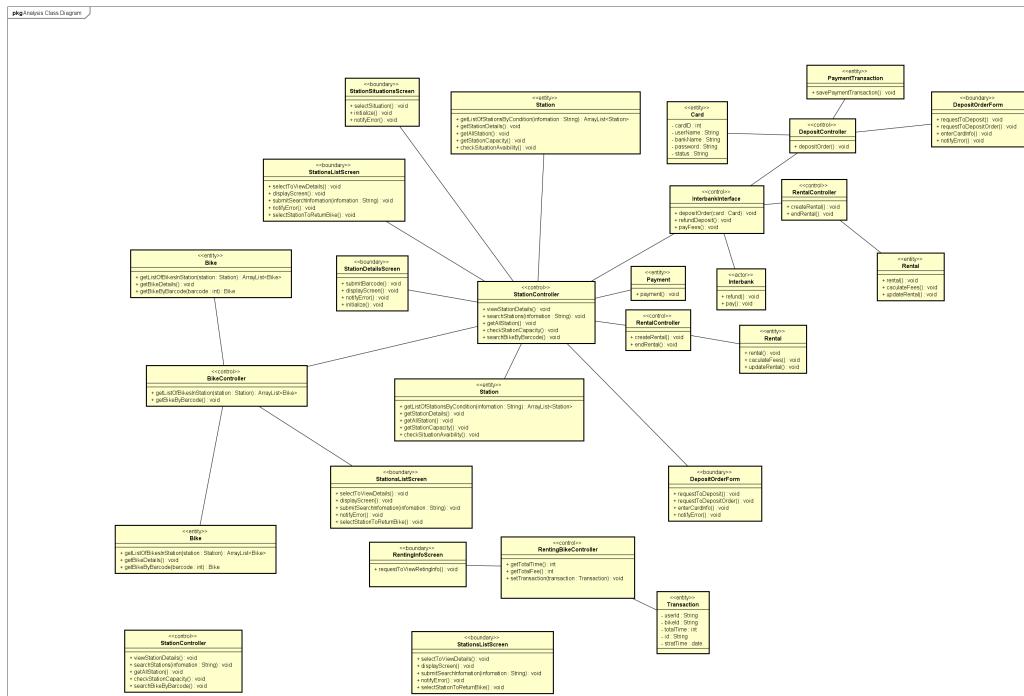


Hình 3.13: View Renting Info Sequence Diagram

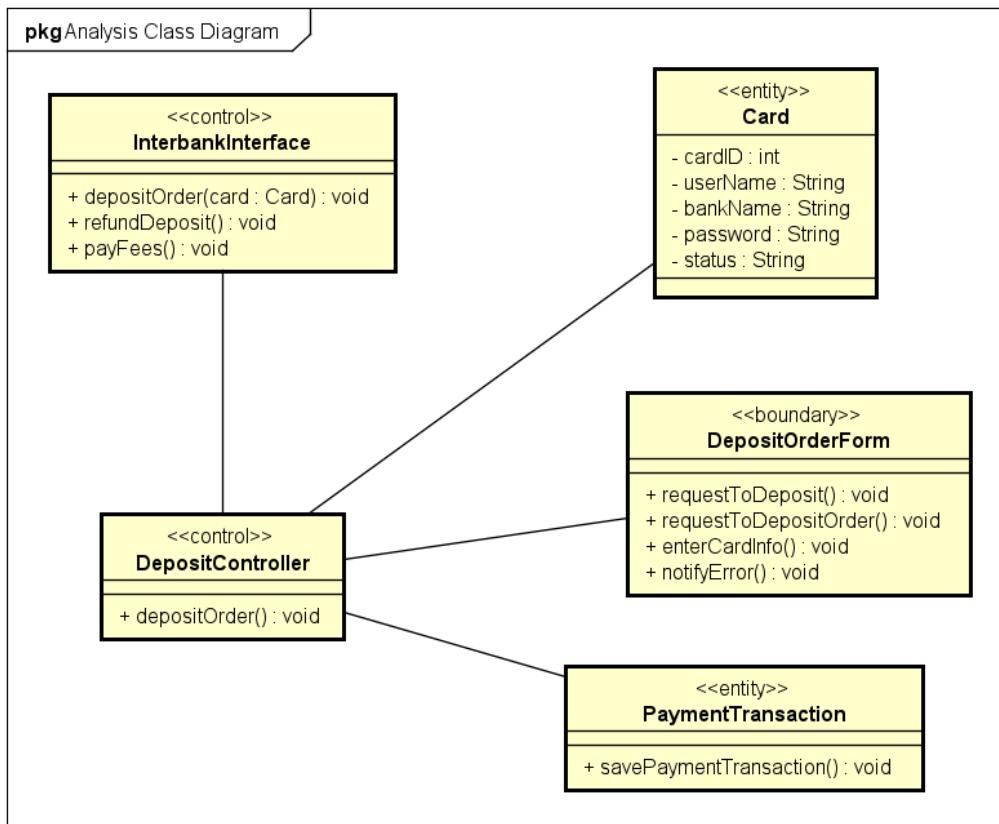


Hình 3.14: View Station Details Sequence Diagram

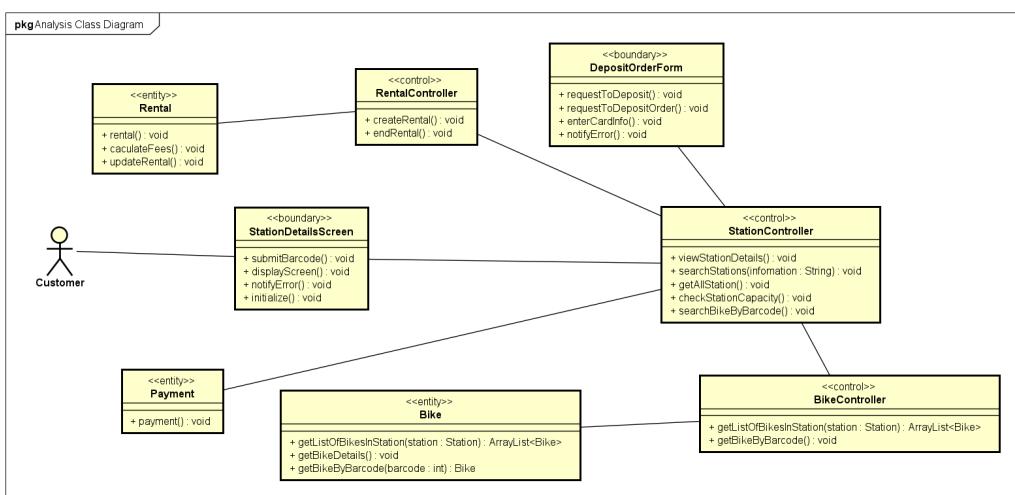
3.2 Analysis Class Diagram



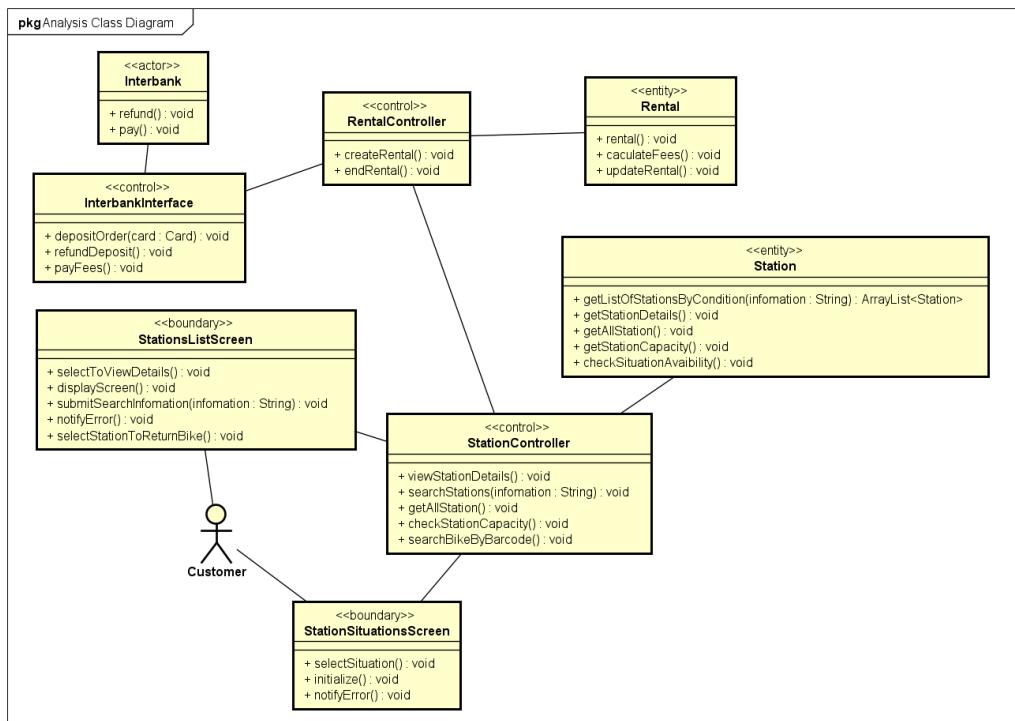
Hình 3.15: All Class Diagram



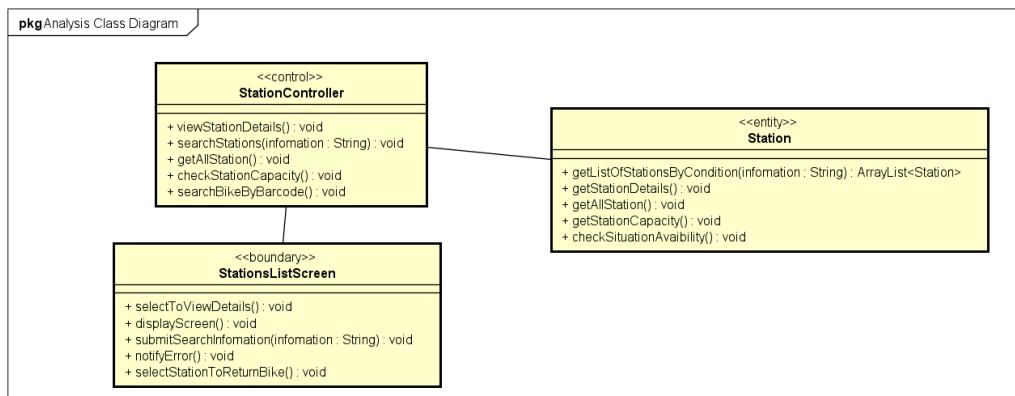
Hình 3.16: Deposit Order Class Diagram



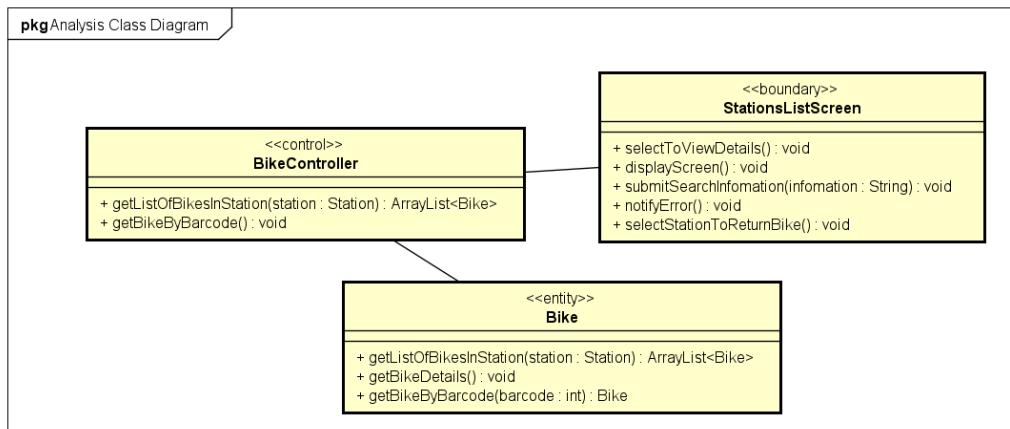
Hình 3.17: Rent Bike Class Diagram



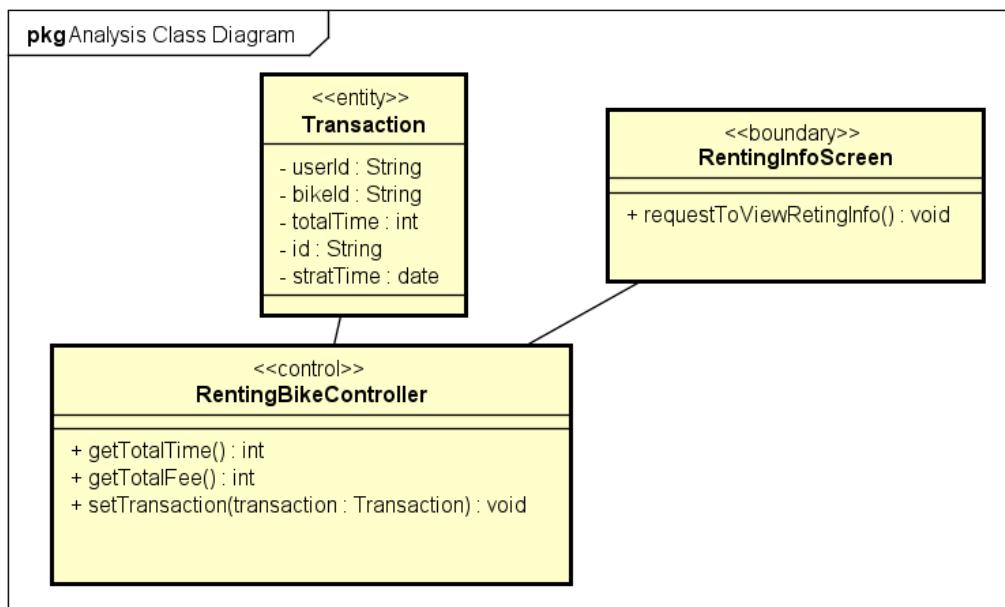
Hình 3.18: Return Bike Class Diagram



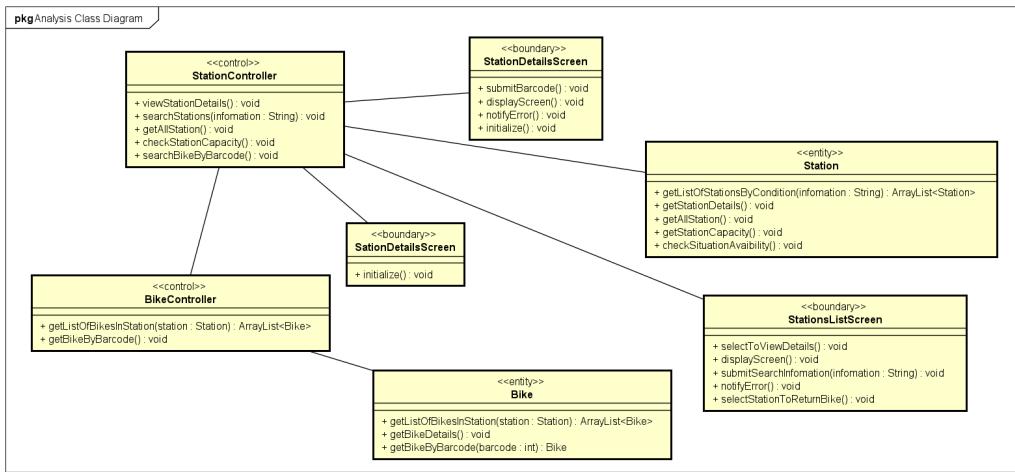
Hình 3.19: Search Stations Class Diagram



Hình 3.20: View Bike Details Class Diagram



Hình 3.21: View Renting Info Class Diagram



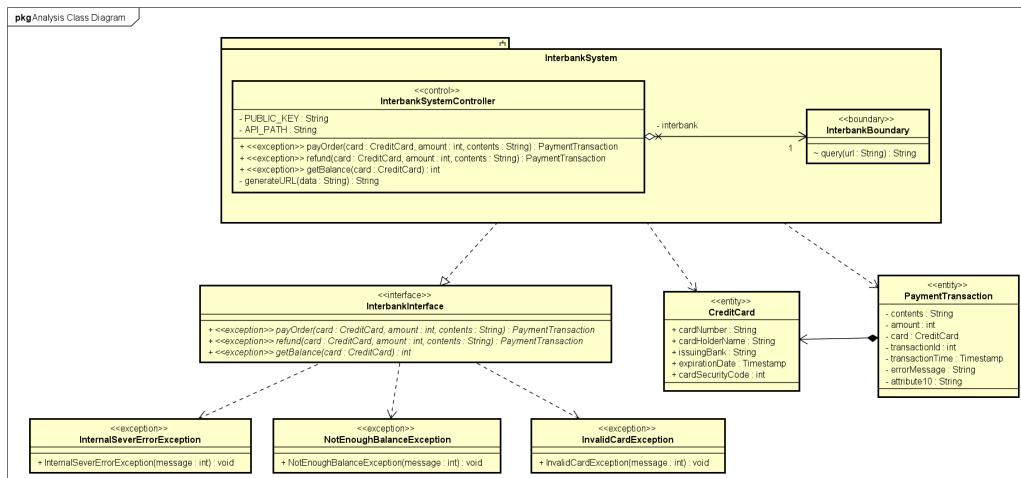
Hình 3.22: View Station Details Class Diagram

Chương 4

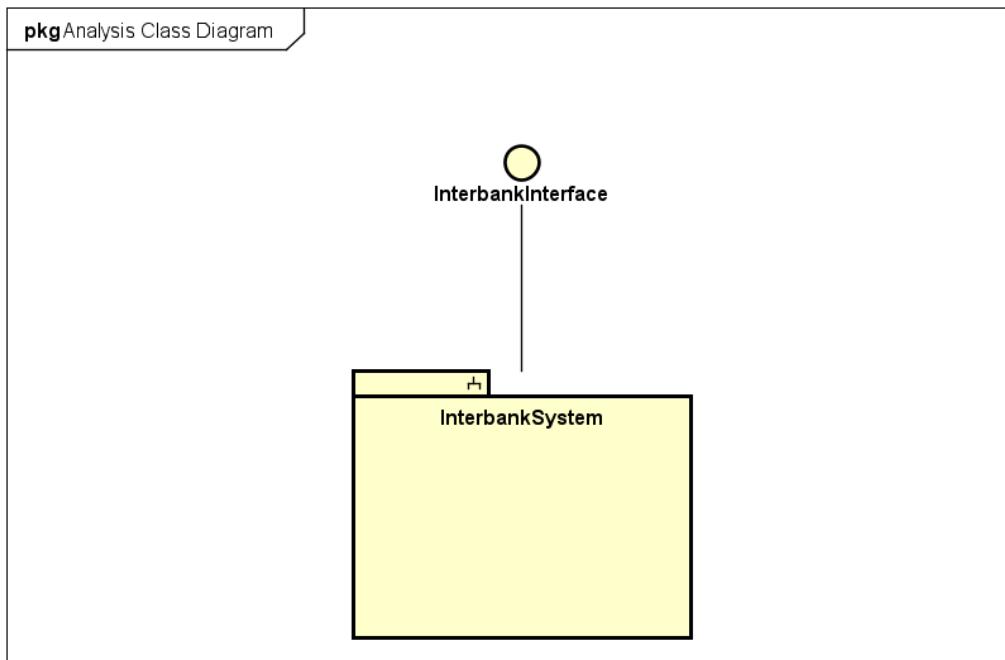
Usecase Design

4.1 Interface Design - Interbank System

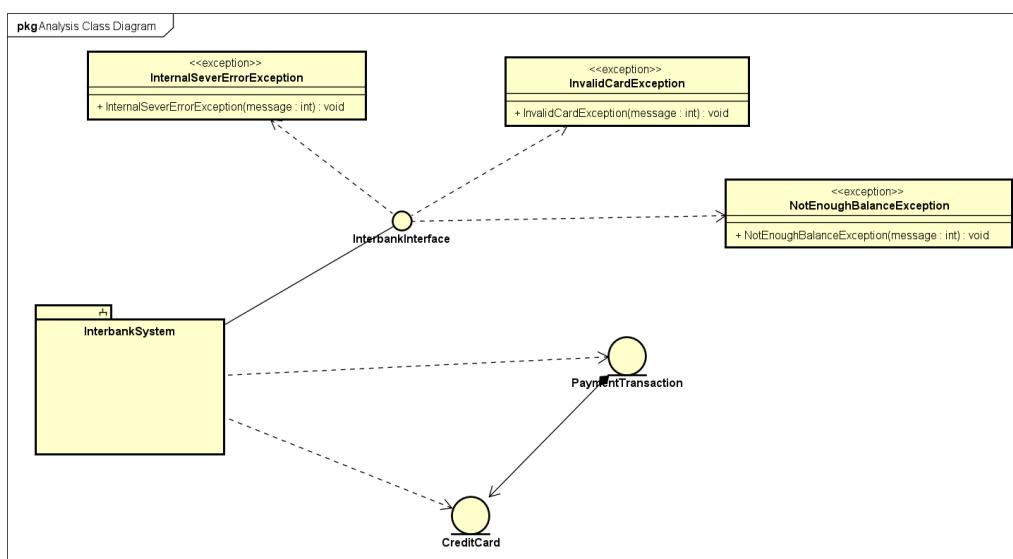
4.1.1 Analysis Class Diagram



Hình 4.1: Checkpoint

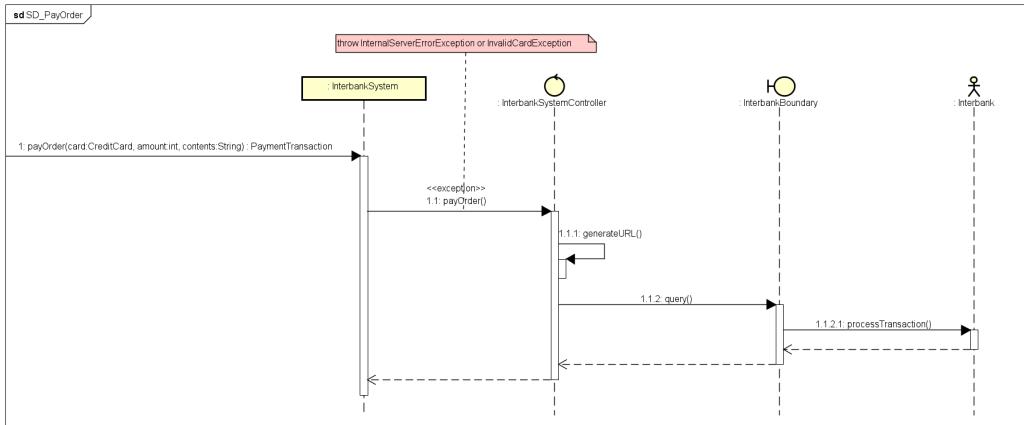


Hình 4.2: Subsystem

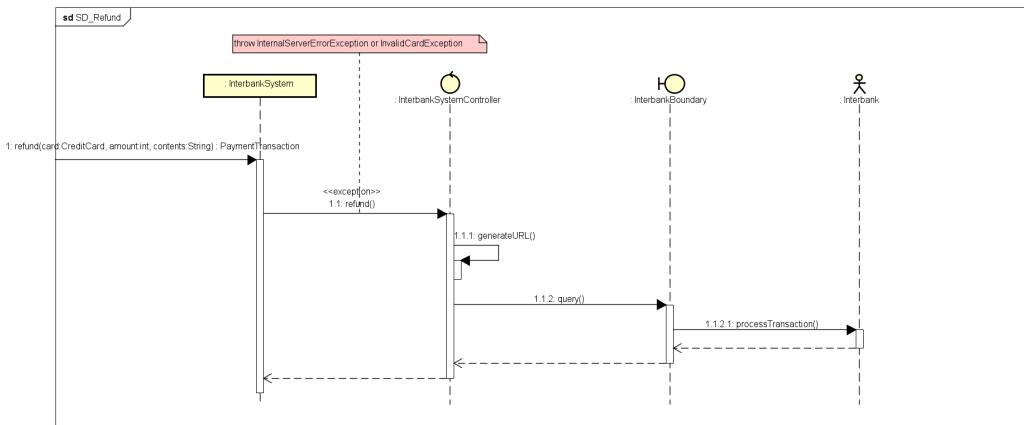


Hình 4.3: Subsystem Dependencies

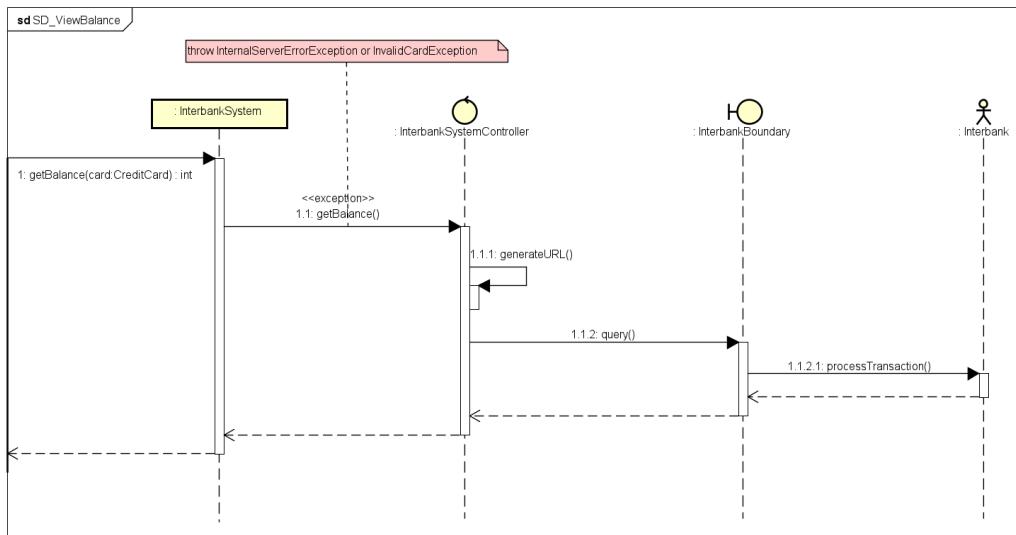
4.1.2 Sequence Diagram



Hình 4.4: Sequence Diagram của Pay Order



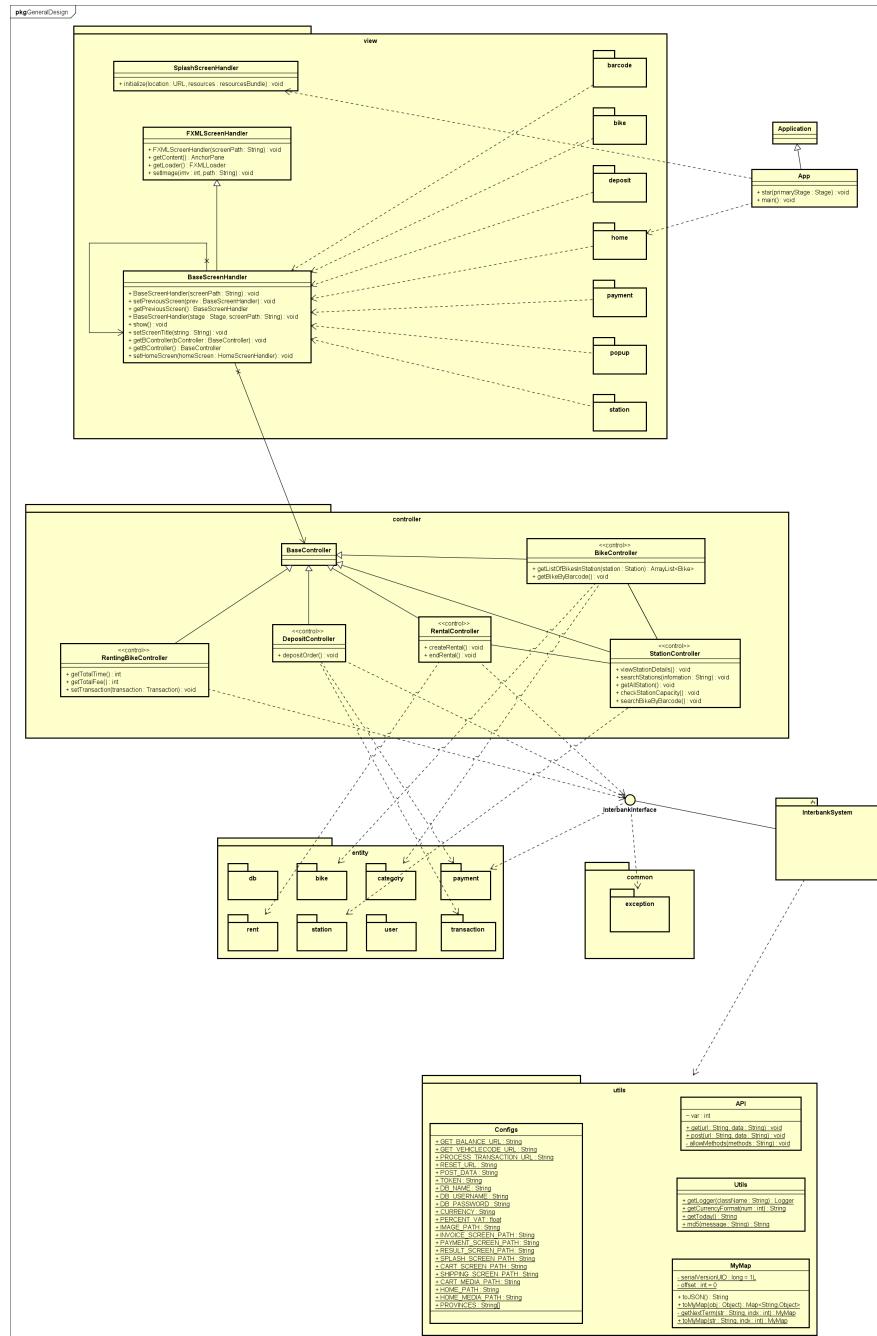
Hình 4.5: Sequence Diagram của Refund



Hình 4.6: Sequence Diagram của View Balance

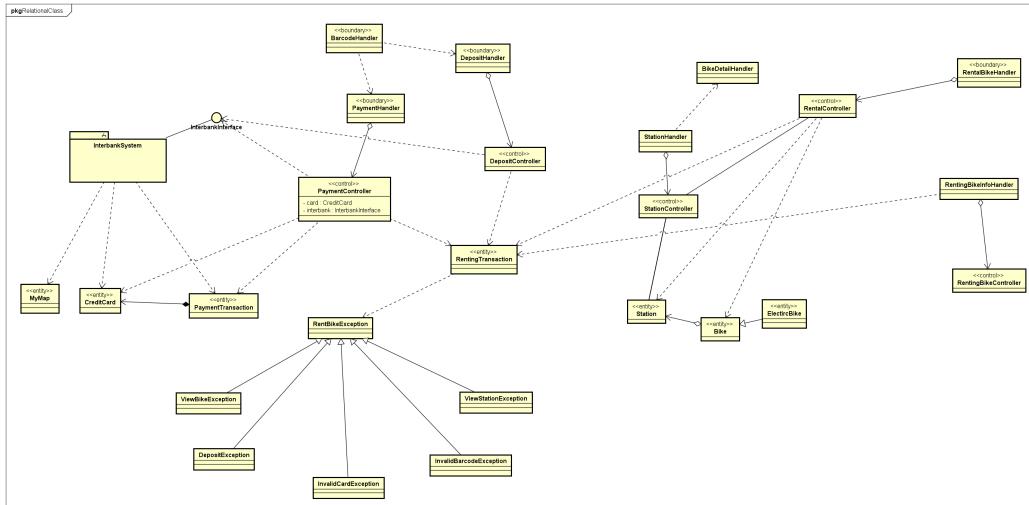
4.2 Class Design

4.2.1 General Design



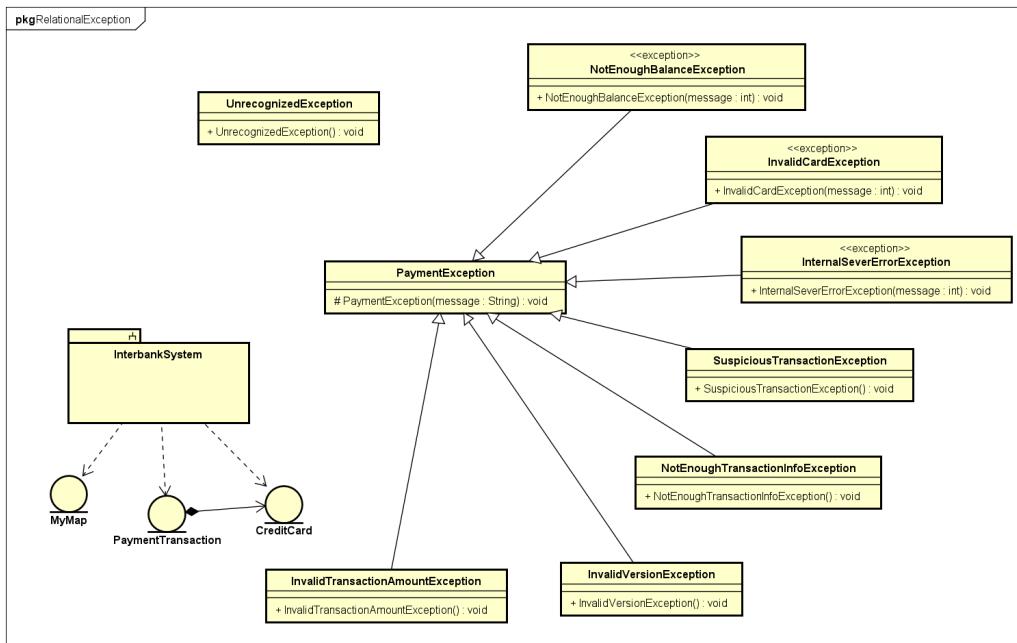
Hình 4.7: General Design

4.2.2 Relational Class



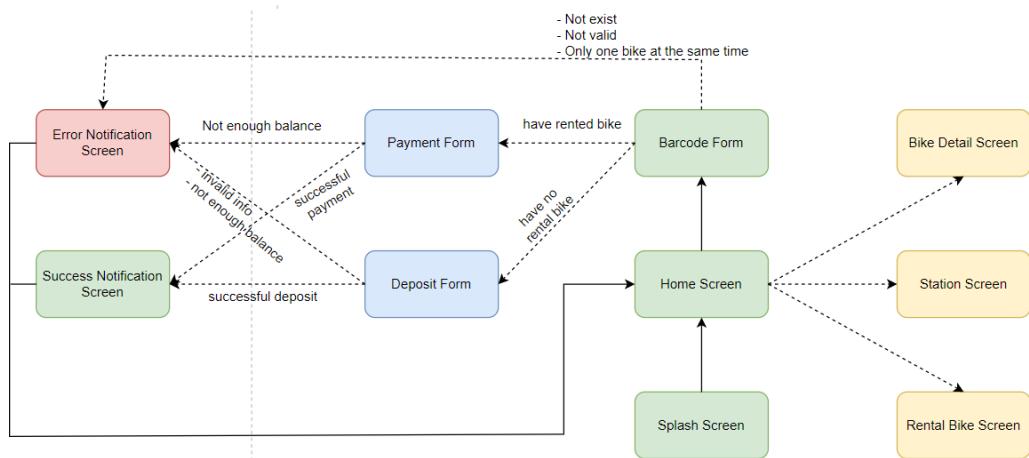
Hình 4.8: Relational Class

4.2.3 Relational Exception



Hình 4.9: Relational Exception

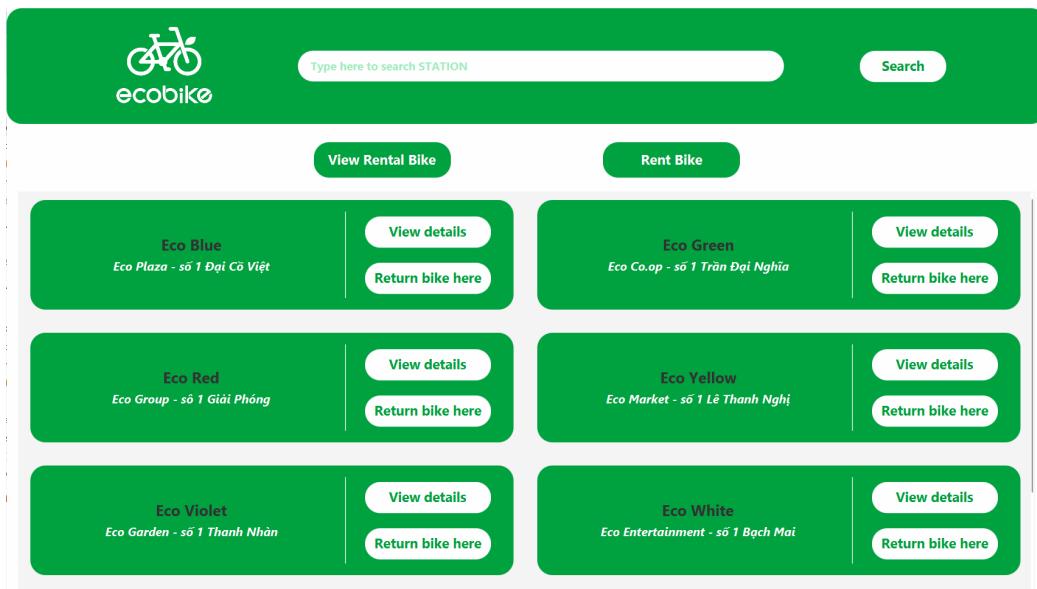
4.3 Graphical User Interface (GUI)



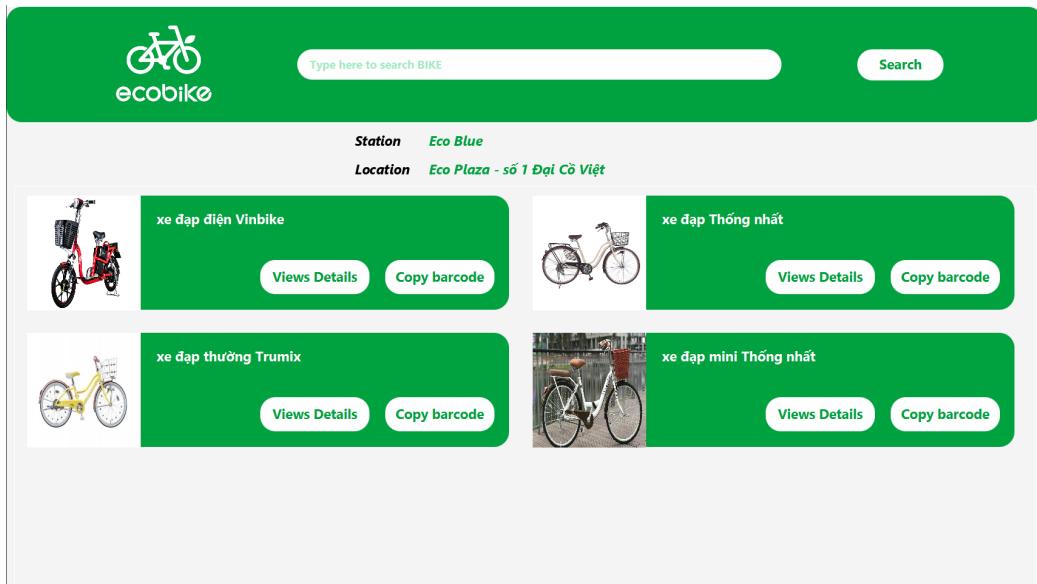
Hình 4.10: Luồng điều khiển của màn hình



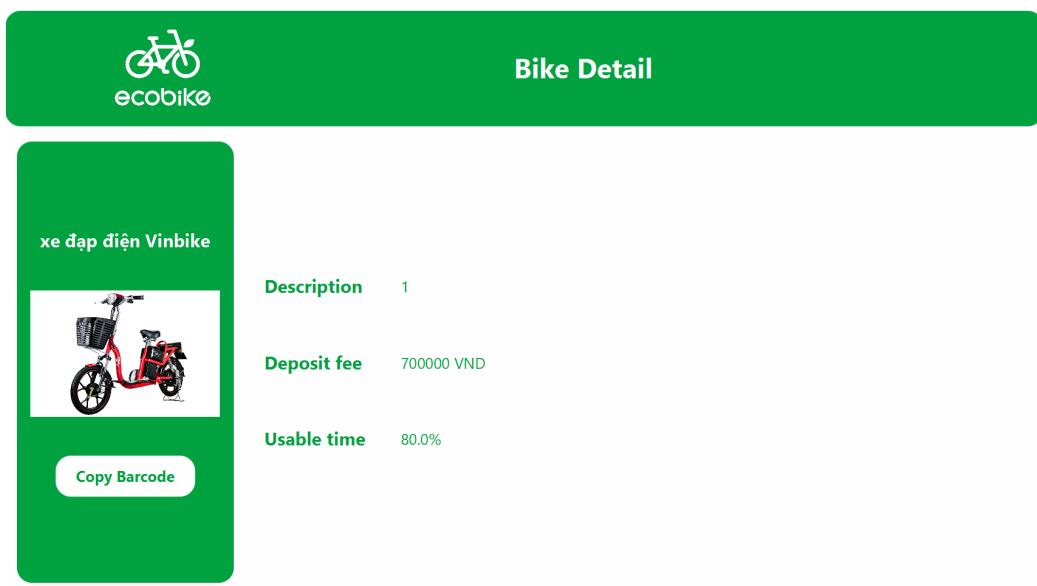
Hình 4.11: Màn hình Splash



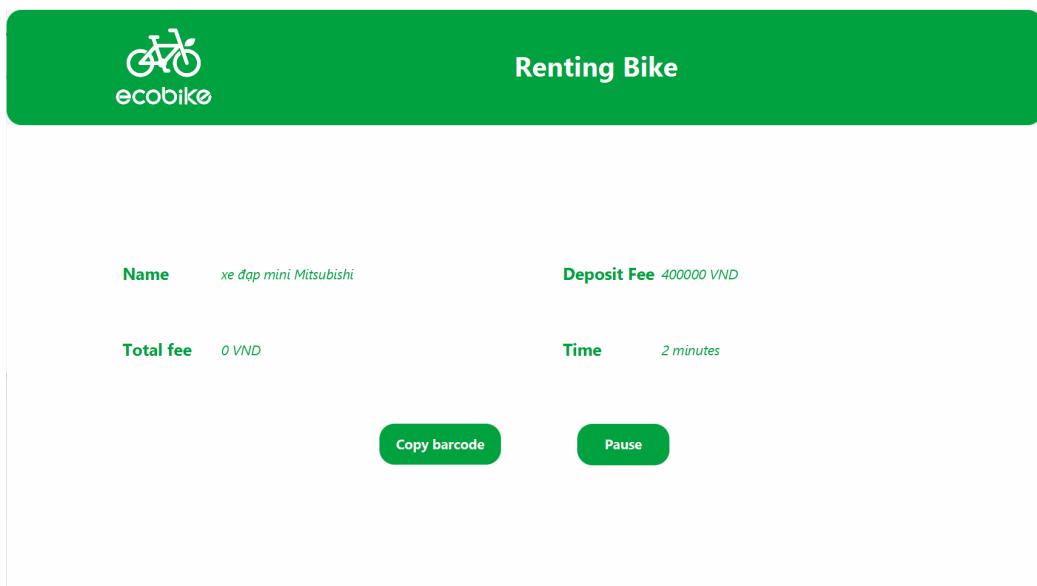
Hình 4.12: Màn hình chính



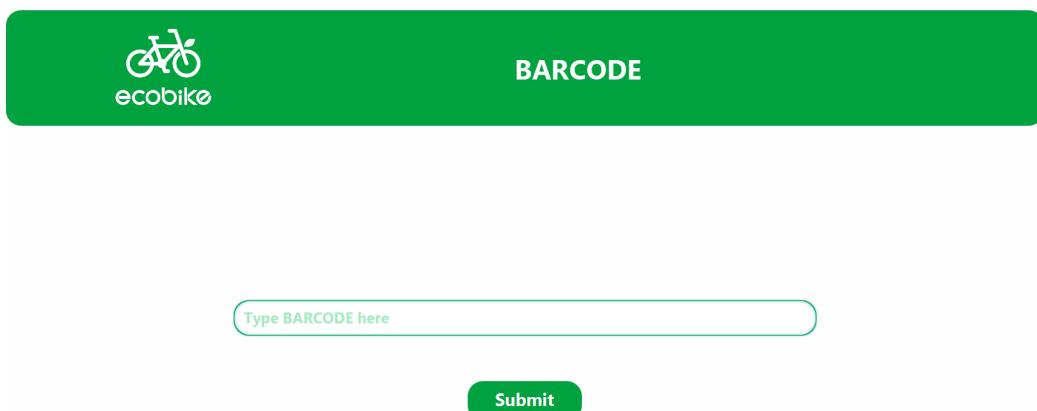
Hình 4.13: Màn hình hiển thị tất cả thông tin của station



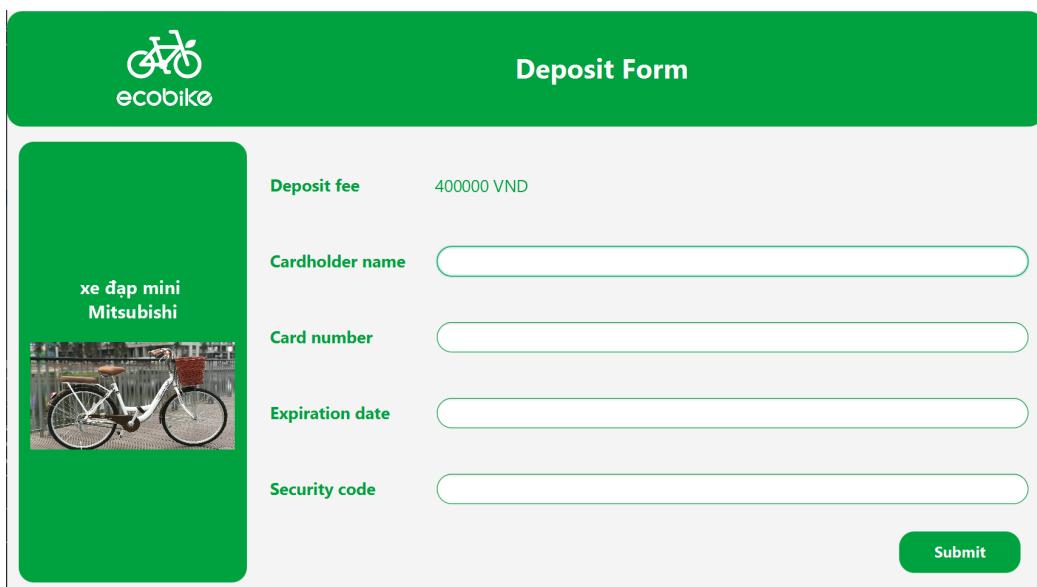
Hình 4.14: Màn hình hiển thị tất cả thông tin của xe



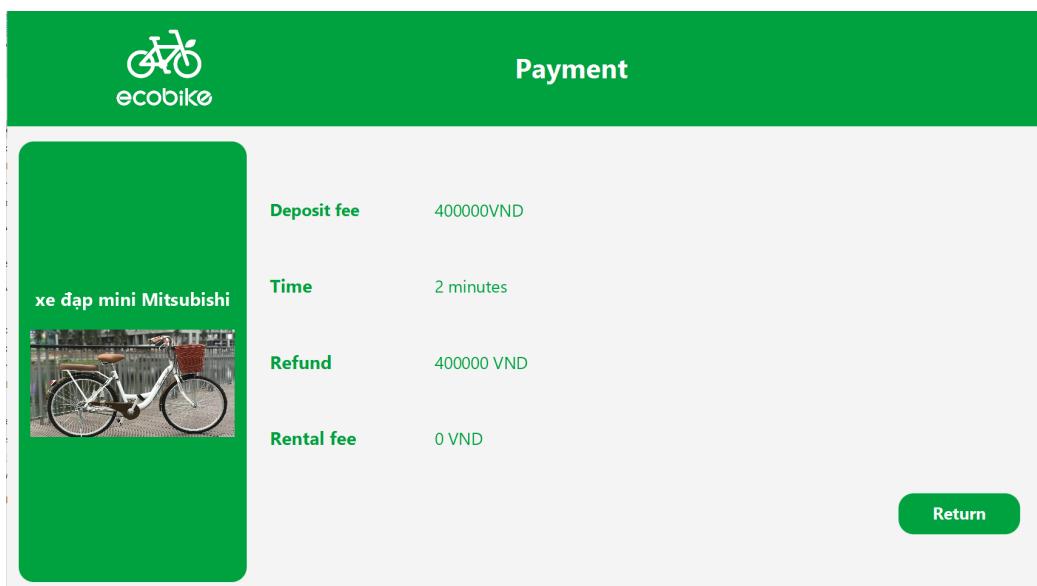
Hình 4.15: Màn hình hiển thị thông tin xe đang thuê



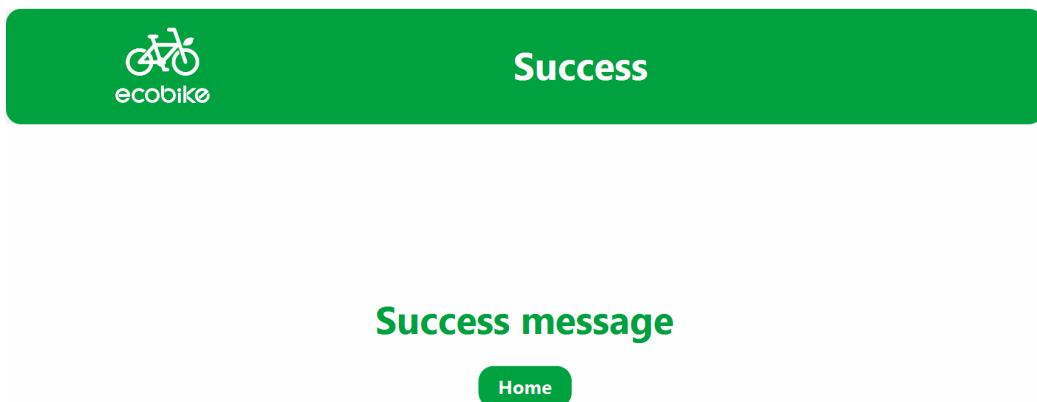
Hình 4.16: Màn hình nhập Barcode



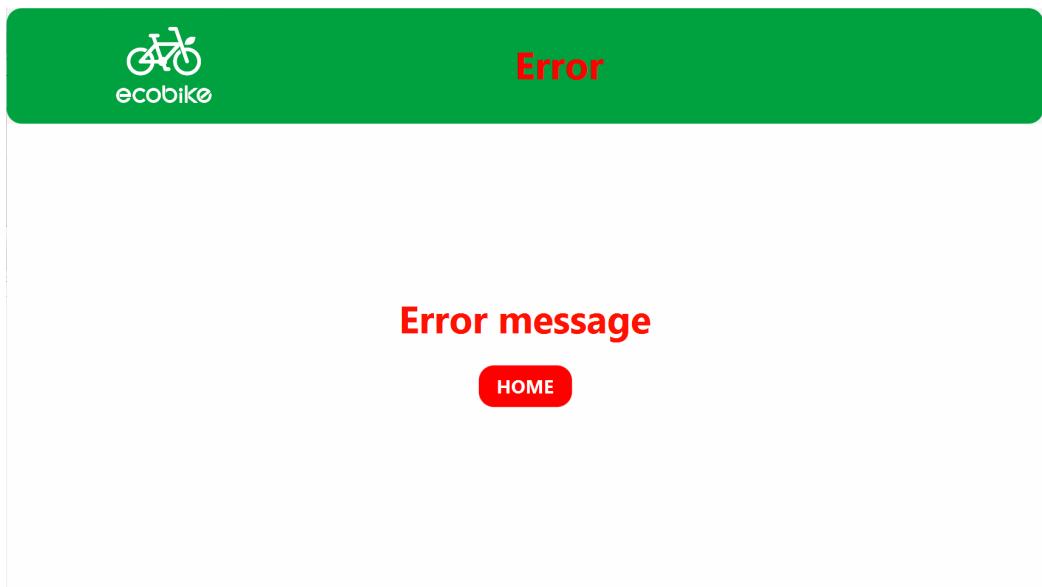
Hình 4.17: Màn hình đặt xe



Hình 4.18: Màn hình thanh toán thuê xe



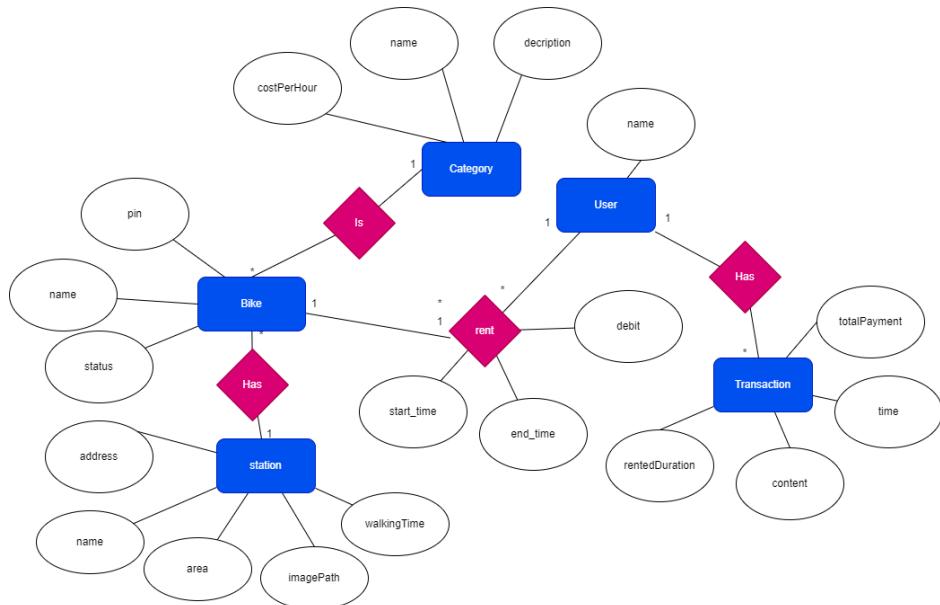
Hình 4.19: Màn hình thông báo giao dịch thành công với Interbank System



Hình 4.20: Màn hình thông báo giao dịch thất bại với Interbank System

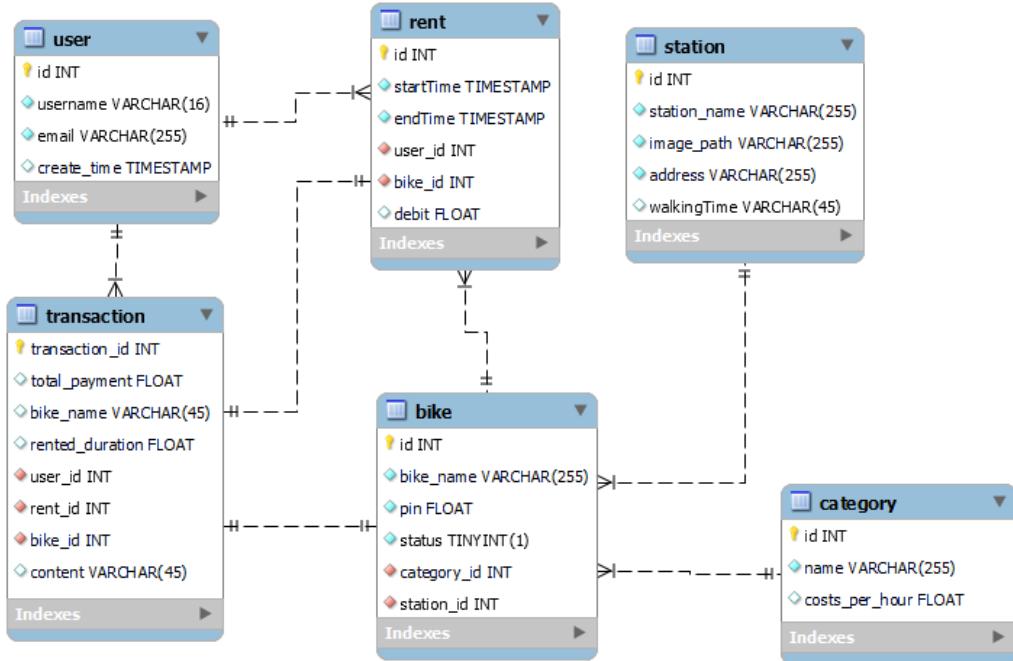
4.4 Data Modeling

4.4.1 E-R Diagram



Hình 4.21: E-R Diagram

4.4.2 Database Diagram



Hình 4.22: Bảng Station

4.4.3 Mô tả Database

#	PK	FK	Column name	Data type	Mandatory	Description
1	x		userId	Integer	Yes	ID người dùng, auto increase
2			userName	VARCHAR(255)	Yes	Tên người dùng

Hình 4.23: Bảng User

#	PK	FK	Column name	Data type	Mandatory	Description
1	x		stationId	Integer	Yes	ID bãi xe, auto increase
2			stationName	VARCHAR(255)	Yes	Tên bãi xe
3			address	VARCHAR(255)	Yes	Địa chỉ bãi xe
4			walkingTime	Float	No	Thời gian người dùng đi bộ tới bãi xe (Đơn vị: giờ)
5		x	imagePath	Integer	Yes	Ảnh minh họa bãi xe

Hình 4.24: Bảng Station

#	PK	FK	Column name	Data type	Mandatory	Description
1	x		categoryId	Integer	Yes	ID loại xe, auto increase
2			categoryName	VARCHAR(255)	Yes	Tên loại xe, thuộc tập {"Xe đạp đơn, Xe đạp đôi, Xe đạp điện"}
3			costPerHour	Float	Yes	Số tiền phải trả trên 1 giờ

Hình 4.25: Bảng Category

#	PK	FK	Column name	Data type	Mandatory	Description
1	x		bikeld	Integer	Yes	ID xe, auto increase
2			bikeName	VARCHAR(255)	Yes	Tên xe
3			pin	Float	No	Lượng pin hiện tại của xe (đối với xe đạp điện)
4			status	BIT	No	Trạng thái của xe
5		x	stationId	Integer	Yes	Mã bãi xe
6		x	categoryId	Integer	Yes	Mã thể loại xe

Hình 4.26: Bảng Bike

#	PK	FK	Column name	Data type	Mandatory	Description
1	x		rentId	Integer	Yes	ID của 1 giao dịch thuê xe, auto increase
2			startTime	TIMESTAMP	Yes	Thời gian bắt đầu thuê
3			endTime	TIMESTAMP	No	Thời gian kết thúc thuê
4			debit	Float	No	Khoản phí thuê xe, đơn vị: nghìn VNĐ
5		x	userId	Integer	Yes	Mã người dùng
6		x	bikeld	Integer	Yes	Mã xe đang thuê

Hình 4.27: Bảng Rent

#	PK	FK	Column name	Data type	Mandatory	Description
1	x		transactionId	Integer	Yes	Mã giao dịch, auto increase
2			totalCost	Float	Yes	Phí cần trả, đơn vị: nghìn VNĐ
3			content	LONG TEXT	Yes	Nội dung giao dịch
4			rentedDuration	Float	No	Thời gian thuê xe, đơn vị: Giờ
5		x	userId	Integer	Yes	Mã người dùng
6		x	bikeld	Integer	Yes	Mã xe đã thuê
7		x	rentId	Integer	Yes	Mã giao dịch thuê

Hình 4.28: Bảng Transaction

Chương 5

Nguyên tắc thiết kế

5.1 Mục tiêu

Các kết quả cần phải đạt được: tối ưu hóa bộ nhớ kết hợp với tối ưu hóa tốc độ nhằm nâng cao trải nghiệm của người dùng. Tuy nhiên đây là dự án có thời gian phát triển ngắn nên ưu tiên những mẫu thiết kế đơn giản và các nguyên tắc thiết kế cũng chưa được chẽ.

5.2 Chiến lược kiến trúc

Do điều kiện cơ sở vật chất hạn hẹp, cụ thể là mỗi bãi xe chỉ có một hoặc một vài máy tính chuyên dụng cho khách hàng sử dụng khi thuê xe, trong khí đó sức chứa của bãi xe lên đến hàng trăm chiếc, nên một vài thiết kế sau cần được quan tâm.

- Ứng dụng được phát triển trên nền tảng duy nhất là Java, database là MySQL. Điều này giúp hiệu năng của hệ thống cao hơn so với các ứng dụng được phát triển trên nhiều ngôn ngữ khác nhau. Bên cạnh đó còn giúp lập trình viên không bị phân tán quá nhiều.
- Cài đặt nhiều sub system nhất có thể, Điều này giúp cho hệ thống dễ dàng mở rộng và bảo trì về sau.
- Xây dựng các interface cho user, hardware để tăng tính tái sử dụng.
- Cover hết các lỗi có thể xảy ra.
- Phục vụ nhiều người nhất có thể trong cùng một lúc

5.3 Coupling và Cohesion

5.3.1 Phân tích tính Cohesion cho các lớp Controller

Lớp StationController thỏa mãn Functional Cohesion vì: class này đảm bảo chức năng đọc dữ liệu từ database và trả về danh sách các bãi xe để Main-Handler hiển thị khi chương trình bắt đầu. Để thực hiện nhiệm vụ này, StationController cung cấp một hàm chính là getAllStation() nhằm đọc danh sách các bãi xe, để lấy danh sách xe của một bãi xe. Bên cạnh đó có các hàm getAllBike() để lấy các thông tin về danh sách các xe. Các hàm này hoàn toàn độc lập với nhau nhưng cùng đảm bảo cho chức năng để hiển thị màn hình chính khi mới khởi tạo. Và các hàm thanh toán tiền đặt cọc và hành thanh toán hóa đơn thuê xe riêng biệt để tính toán trước khi người dùng sử dụng chức năng liên quan đến các hoạt động thuê xe và trả xe.

Lớp

5.3.2 Phân tích tính coupling giữa các lớp Controller

Các lớp trong package controller là hoàn toàn tách biệt với nhau và chúng không có chức năng nào gọi cho nhau. Đảm bảo tính tách biệt

5.4 Nguyên tắc thiết kế

5.4.1 Single Responsibility

Mỗi lớp trong source code chỉ chịu trách nhiệm về một nhiệm vụ cụ thể nào đó và chỉ phục vụ cho một mục tiêu duy nhất. Một lớp quá nhiều chức năng thì trở nên cồng kềnh, khó đọc, khó bảo trì.

5.4.2 Open/Closed

Theo nguyên lý này, mỗi khi chúng ta thêm mới chức năng chúng ta nên viết class mới extend từ class đã có chứ không nên chỉnh sửa trực tiếp nội dung trên class đã viết. Một ví dụ tiêu biểu nhất trong bài tập lớn là Inter bank subsystem, khi giao thức kết nối và API thay đổi thì chúng ta chỉ cần viết một subsystem khác implement các phương thức payOrder, refund ... Điều này đáp ứng được tính open/closed, tức là thay đổi yêu cầu mà không phải sửa lại thiết kế cũ. Một ví dụ khác nữa, chẳng hạn thay đổi yêu cầu tính toán thì chỉ cần implement lại phương thanh toán của interface.

5.4.3 Liskov Substitution

Nguyên tắc này nói rằng, các đối tượng của class con có thể thay thế cho lớp cha ở mọi tính huống mà không gây ra lỗi, hoặc nếu không, chúng ta có sự trừu tượng sai. Có thể thấy các lớp Entity tuân theo nguyên tắc này bởi chúng đều được kế thừa bởi Class Record.

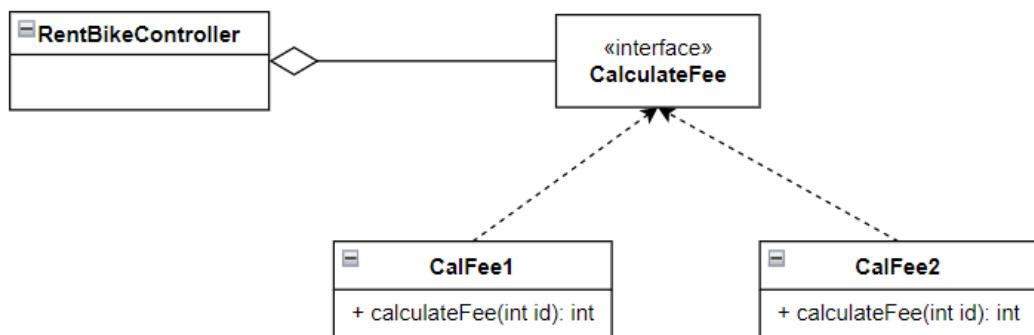
5.4.4 Interface Segregation

Các interface của source code không quá lớn. Không cần phải chia thành các interface nhỏ hơn.

5.5 Design Pattern

Strategy: Behavioral Pattern

- Trong case study có yêu cầu thay đổi các tính phí. Ở đây nhóm sử dụng Strategy để xây dựng chức năng này.



Hình 5.1: Strategy pattern cho chức năng tính phí

Singleton: Creational Pattern

- Sử dụng cho class CalFee tránh việc khởi tạo đối tượng nhiều lần
- Class ConnectionDB chỉ khởi tạo duy nhất 1 lần, tránh việc mỗi khi cần truy cập đến database thì khởi tạo 1 instance thì mất rất nhiều thời gian.