

HANOI UNIVERSITY OF SCIENCE AND TECHNOLOGY
School of Information and communications technology

Software Design Document

Version 1.3

Capston Project

Subject: Thiết kế và xây dựng phần mềm

Group 8

Họ và tên	MSSV
Trần Ngọc Phiên	20183603
Trần Hải Trung	20183644
Đỗ Văn Thông	20183636
Nguyễn Việt Đức	20183892

Hà Nội, tháng 1 năm 2022

Table of Contents

Table of Contents	1
1 Introduction	3
Objective.....	3
Scope	3
Glossary	3
References	3
2 Overall Description	4
2.1 General Overview	4
2.2 Assumptions/Constraints/Risks	4
2.2.1 Assumptions.....	4
2.2.2 Constraints	4
3 System Architecture and Architecture Design	5
3.1 Architectural Patterns	5
3.2 Interaction Diagrams	5
3.2.1 Usecase Tìm kiếm bãi xe	5
3.2.2 Usecase Xem chi tiết bãi xe	6
3.2.3 Usecase Thuê xe	7
3.2.4 Usecase Thanh toán giao dịch.....	8
3.2.5 Usecase Xem xe đang thuê	8
3.2.6 Usecase Trả xe	9
3.3 Analysis Class Diagrams	10
3.3.1 Usecase Tìm kiếm bãi xe	10
3.3.2 Usecase Xem chi tiết bãi xe	10
3.3.3 Usecase Thuê xe	11
3.3.4 Usecase Thanh toán giao dịch.....	11
3.3.5 Usecase Xem xe đang thuê	11

3.3.6	Usecase Trã xe	12
3.4	Unified Analysis Class Diagram	12
3.5	Security Software Architecture	12
4	Detailed Design	13
4.1	User Interface Design	13
4.1.1	Screen Configuration Standardization	13
4.1.2	Screen Transition Diagrams.....	13
4.1.3	Screen Specifications	13
4.2	Data Modeling	21
4.2.1	Conceptual Data Modeling	21
4.2.2	Database Design.....	22
4.3	Class Design	33
4.3.1	General Class Diagram	33
4.3.2	Class Diagrams	33
4.3.3	Class Design.....	35
14	Design Considerations	58
14.1	Goals and Guidelines	58
14.2	Architectural Strategies	58
14.3	Cohesion.....	58
14.4	Design Principles.....	61
14.5	Design Patterns.....	62

1 Introduction

Tài liệu thiết kế phần mềm thuê xe đạp Capston mô tả chi tiết về toàn bộ thiết kế chương trình, bao gồm kiến trúc của chương trình, mô hình dữ liệu, lớp thiết kế chi tiết, các nguyên tắc để thiết kế phần mềm

Objective

Giúp người đọc tài liệu hiểu rõ chi tiết về phần mềm một cách mạch lạc .

Scope

Tài liệu này bao gồm :

- Kiến trúc hệ thống, Biểu đồ quan hệ, Biểu đồ lớp,
- Thiết kế giao diện người dùng
- Thiết kế cơ sở dữ liệu, mô tả rõ chi tiết dữ liệu
- Thiết kế lớp chi tiết
- Phương pháp thiết kế chương trình

Glossary

References

- [1] Centers for Medicare & Medicaid Services, "System Design Document Template," [Online]. Available: <https://www.cms.gov/Research-Statistics-Data-and-Systems/CMS-Information-Technology/XLC/Downloads/SystemDesignDocument.docx>.

2 Overall Description

2.1 General Overview

Mục đích của phần mềm là tạo ra một ứng dụng có thể hỗ trợ cho việc mượn trả xe dễ dàng hơn. Người dùng có thể đăng ký tài khoản cho mình, sau đó có thể đăng nhập để sử dụng các chức năng của hệ thống. Người dùng có thể xem thông tin của các bãi xe vị trí của họ bằng cách lựa chọn trên bản đồ hoặc có thể tìm kiếm. Trong mỗi bãi xe có nhiều loại xe khác nhau với phí thuê khác nhau, người dùng có thể xem thông tin chi tiết của các xe trong bãi xe lựa chọn loại xe phù hợp nhất với nhu cầu và mong muốn của mình. Người dùng phải đặt cọc một khoản tiền để có thể thuê xe, số tiền này người dùng có thể thanh toán qua các ngân hàng liên kết hoặc ví điện tử. Trong quá trình thuê xe, người dùng có thể xem thông tin về xe mà họ đang thuê. Sau khi sử dụng xong, đưa xe vào ổ khóa tại mỗi vị trí để xe để trả xe tự động. Hệ thống mong muốn mang đến cho người dùng những trải nghiệm tốt nhất trong quá trình sử dụng dịch vụ.

2.2 Assumptions/Constraints/Risks

2.2.1 Assumptions

Hệ thống giả lập được xây dựng và phát triển hỗ trợ người dùng có thể sử dụng trên máy tính cá nhân. Hệ thống có thể chạy trên các máy tính sử dụng hệ điều hành window... Người dùng có thể tương tác với phần mềm thông qua chuột và bàn phím, hiển thị giao diện lên màn hình. Trong tương lai có thể xây dựng và phát triển trên các thiết bị di động như ios, android để người dùng dễ dàng sử dụng hơn.

2.2.2 Constraints

Xây dựng bằng ngôn ngữ java, sử dụng thư viện javafx để tạo ra các thành phần giao diện. Hệ thống có thể chạy được trên hệ điều hành windows. Cần kết nối mạng internet để kết nối đến database.

3 System Architecture and Architecture Design

3.1 Architectural Patterns

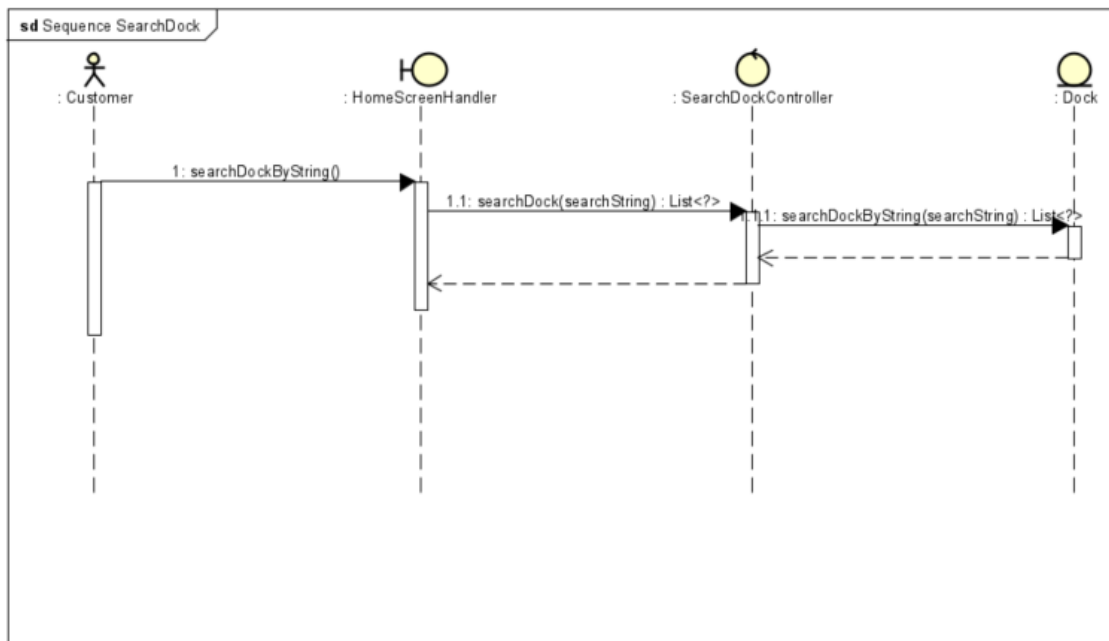
Mô hình MVC đã được sử dụng cho bài toán này.

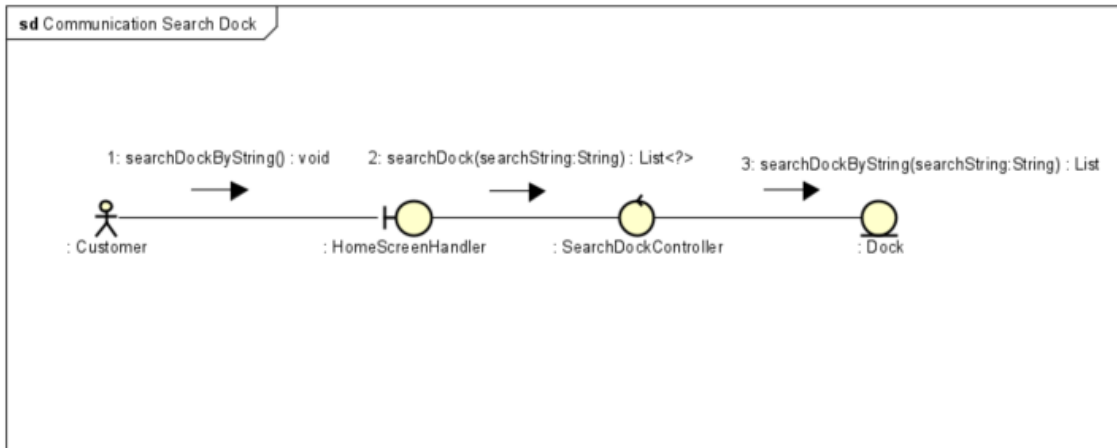
- Model (M): nơi giao tiếp với cơ sở dữ liệu, thực hiện truy vấn cơ sở dữ liệu, chứa dữ liệu để trả về cho controller. Trong bài toán này thì các class trong package entity sẽ đóng vai trò là các model của ứng dụng.
- View (V): nơi hiển thị dữ liệu và tương tác với người dùng. Trong bài toán này thì lớp các class handler đóng vai trò như các view của ứng dụng.
- Controller (C): nơi điều hướng và xử lý logic của ứng dụng. Các controller của bài toán này đóng vai trò là các controller trong mô hình này.

Lý do sử dụng mô hình này bởi vì có thể tách biệt các use-case dễ dàng chia công việc vì các module sẽ tách biệt với nhau. Thời gian phát triển ứng dụng nhanh gọn nhẹ. Dễ dàng kiểm thử.

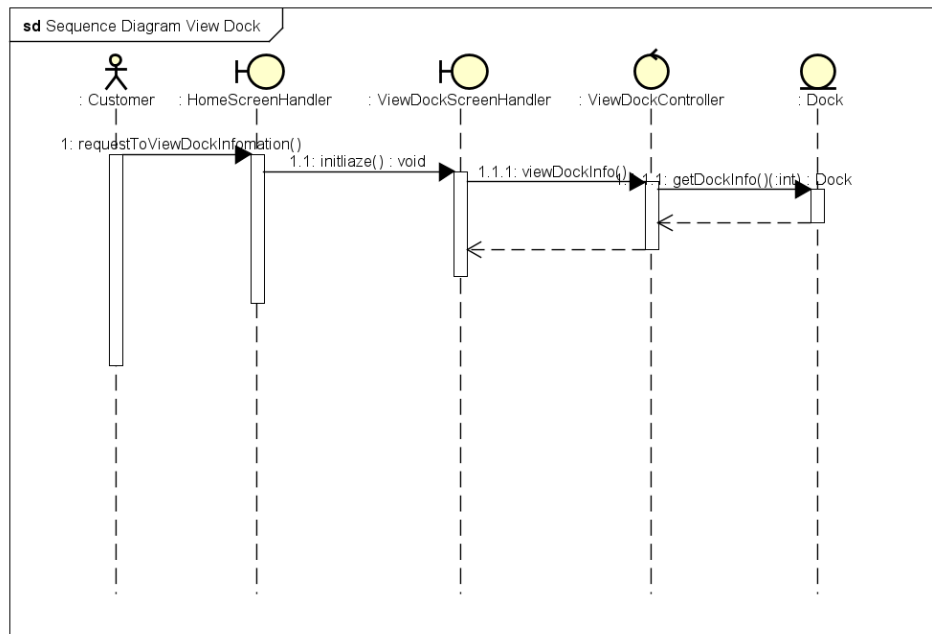
3.2 Interaction Diagrams

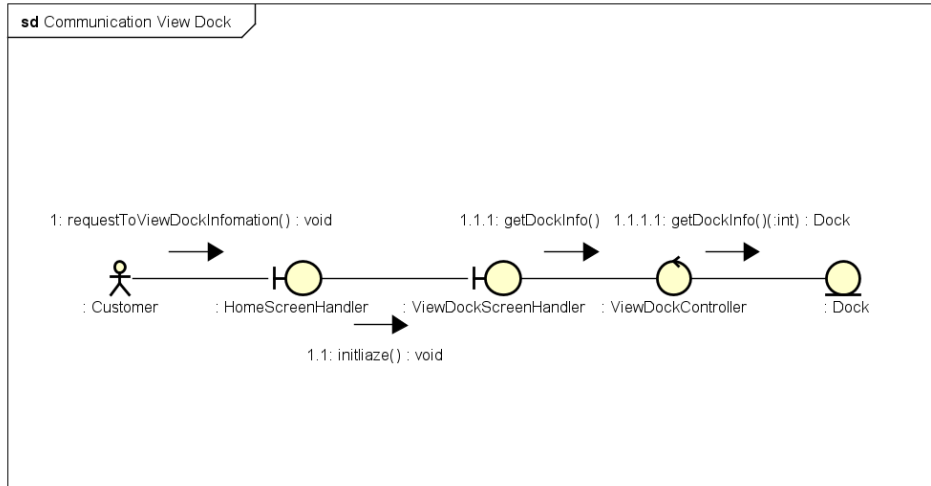
3.2.1 Usecase Tìm kiếm bãi xe



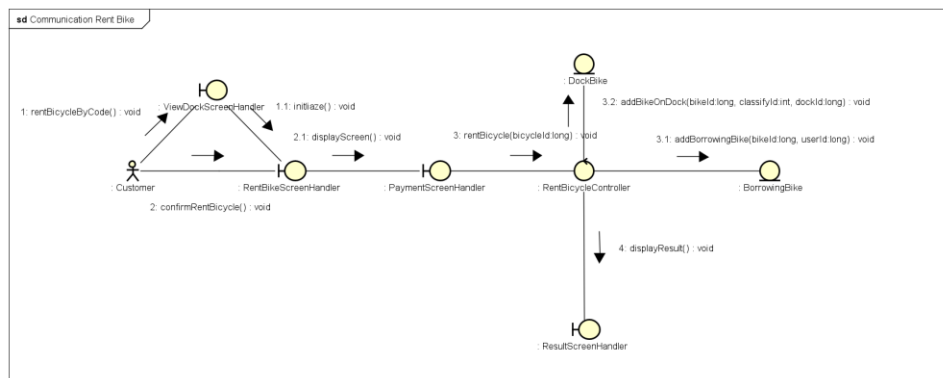
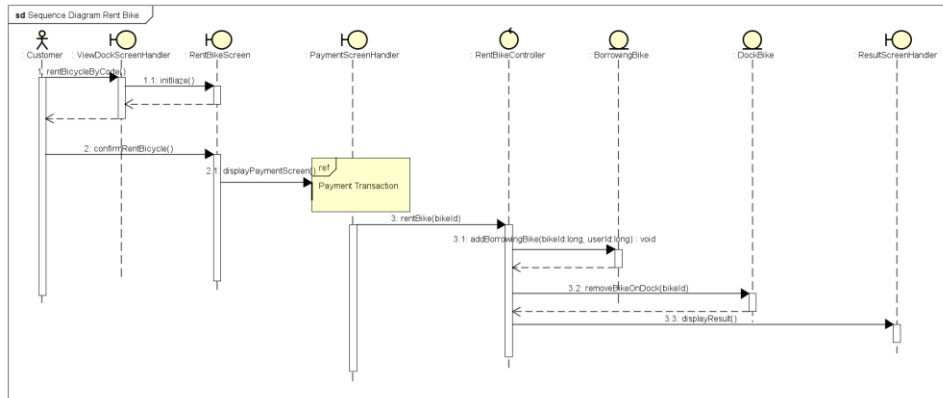


3.2.2 Usecase Xem chi tiết bãi xe

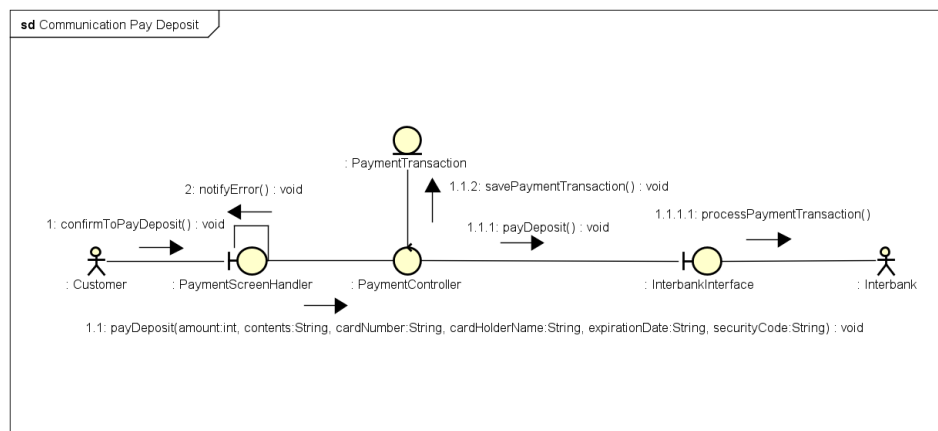
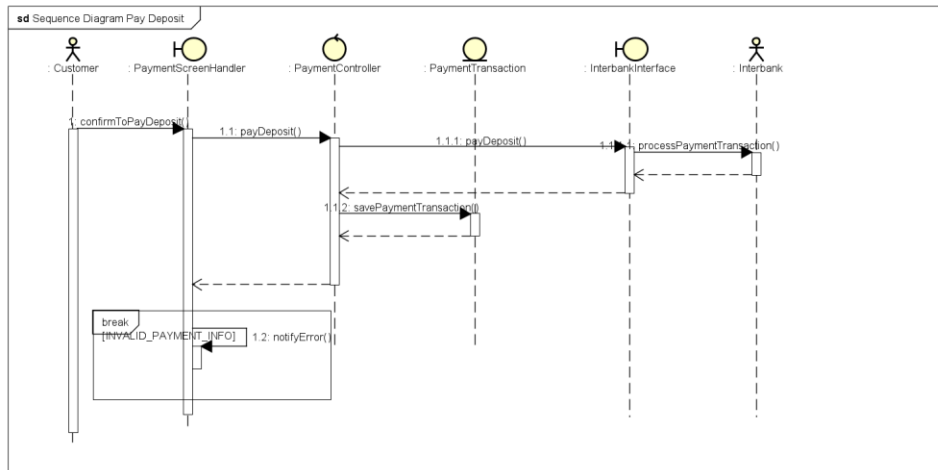




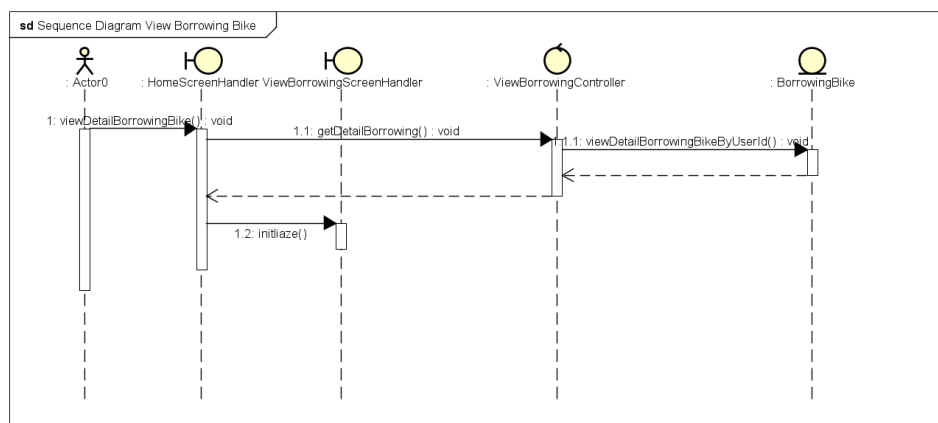
3.2.3 Usecase Thuê xe

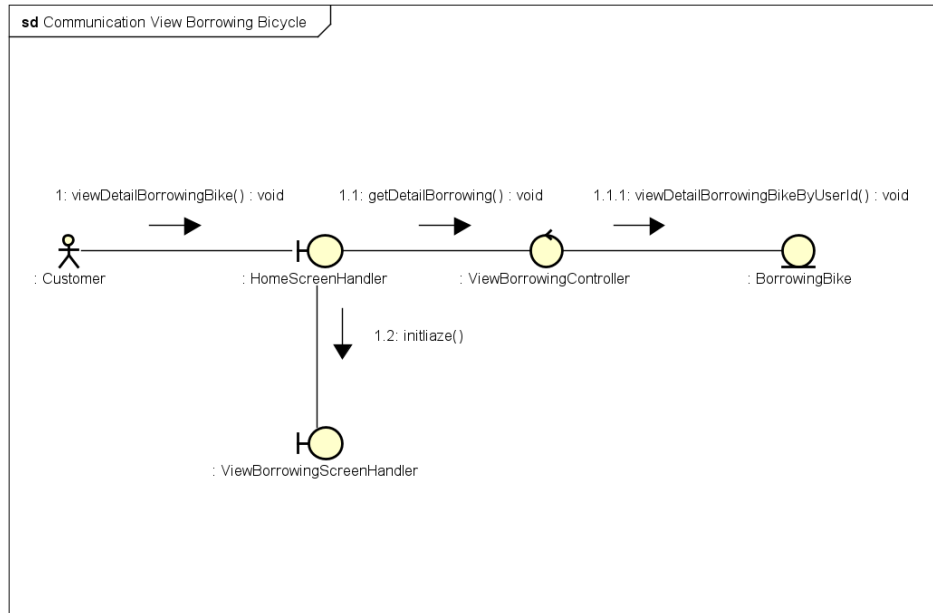


3.2.4 Usecase Thanh toán giao dịch

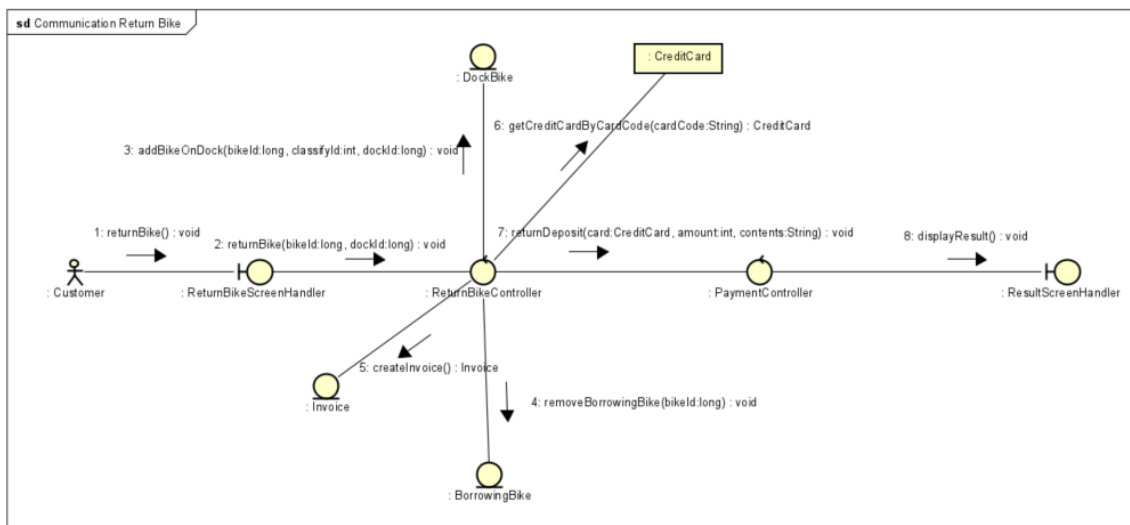
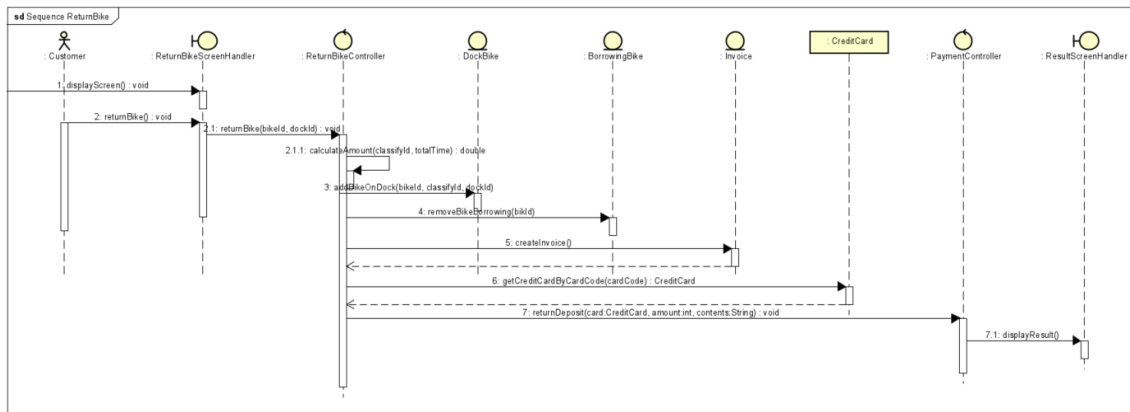


3.2.5 Usecase Xem xe đang thuê



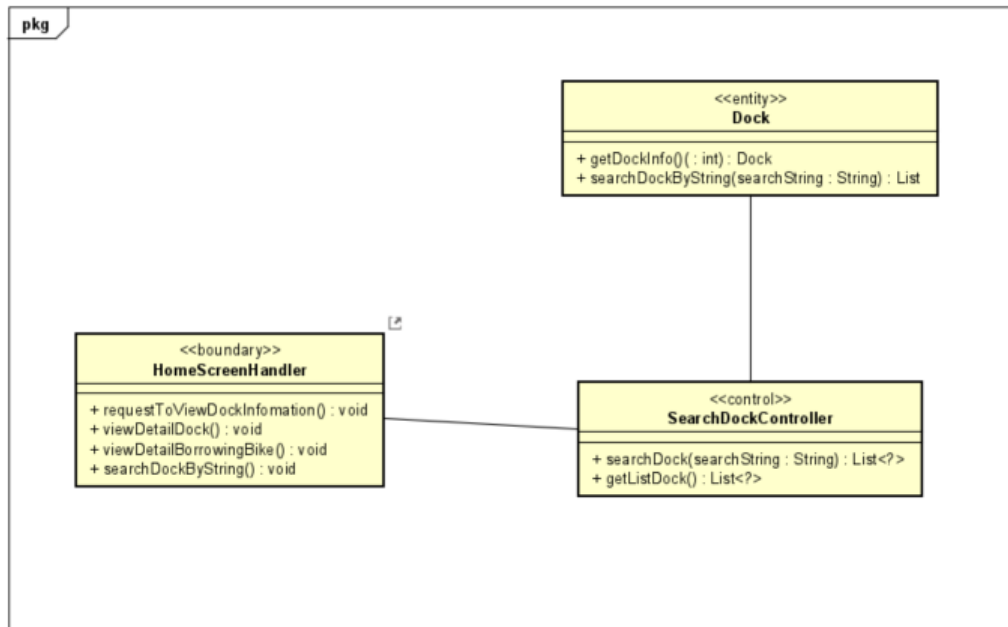


3.2.6 Usecase Trả xe

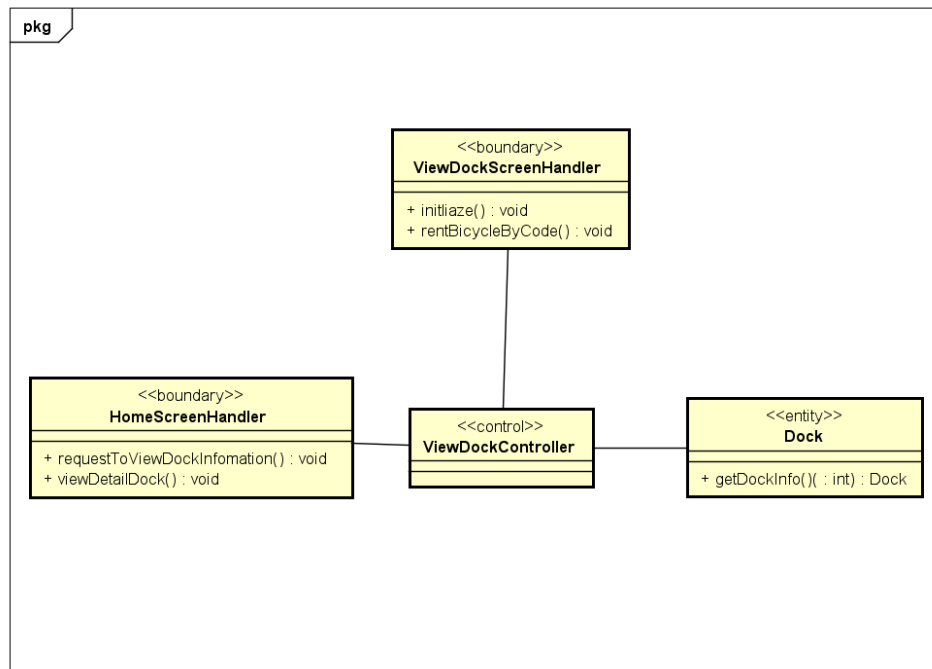


3.3 Analysis Class Diagrams

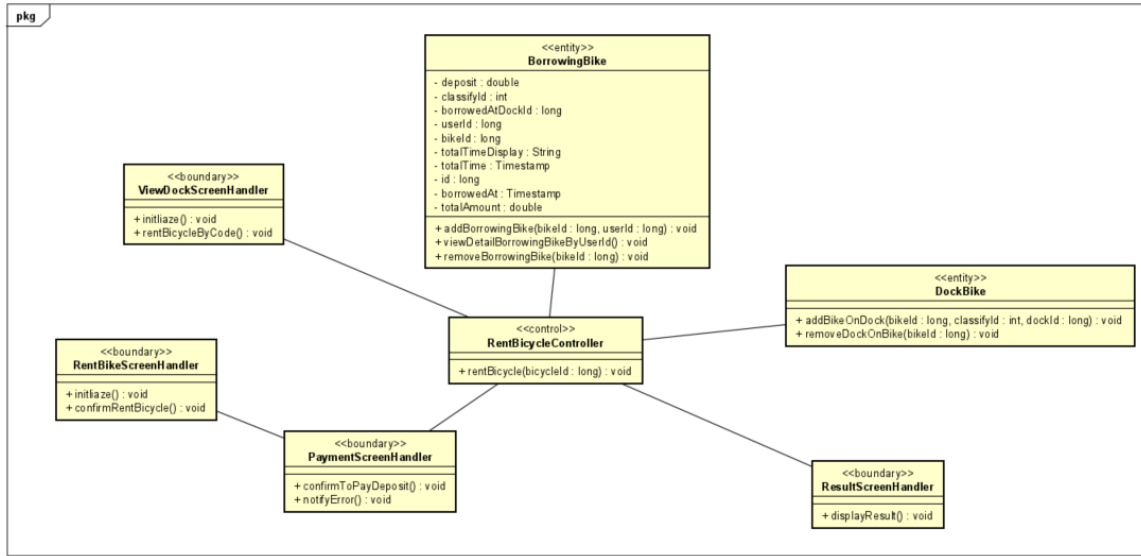
3.3.1 Usecase Tìm kiếm bãi xe



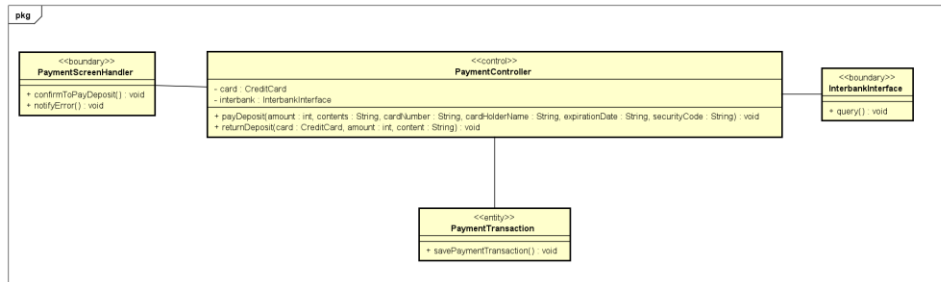
3.3.2 Usecase Xem chi tiết bãi xe



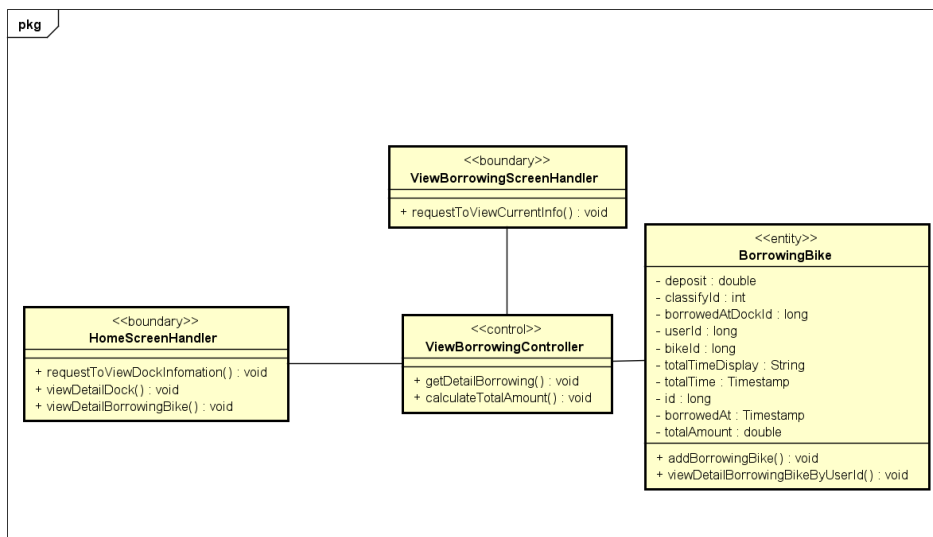
3.3.3 Usecase Thuê xe



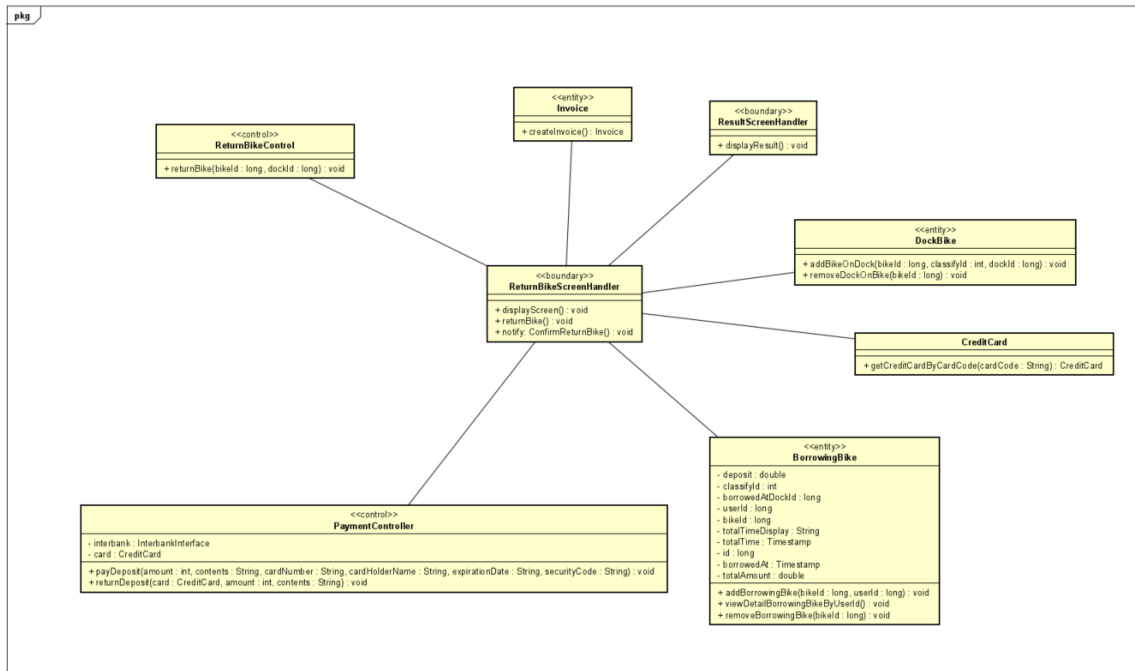
3.3.4 Usecase Thanh toán giao dịch



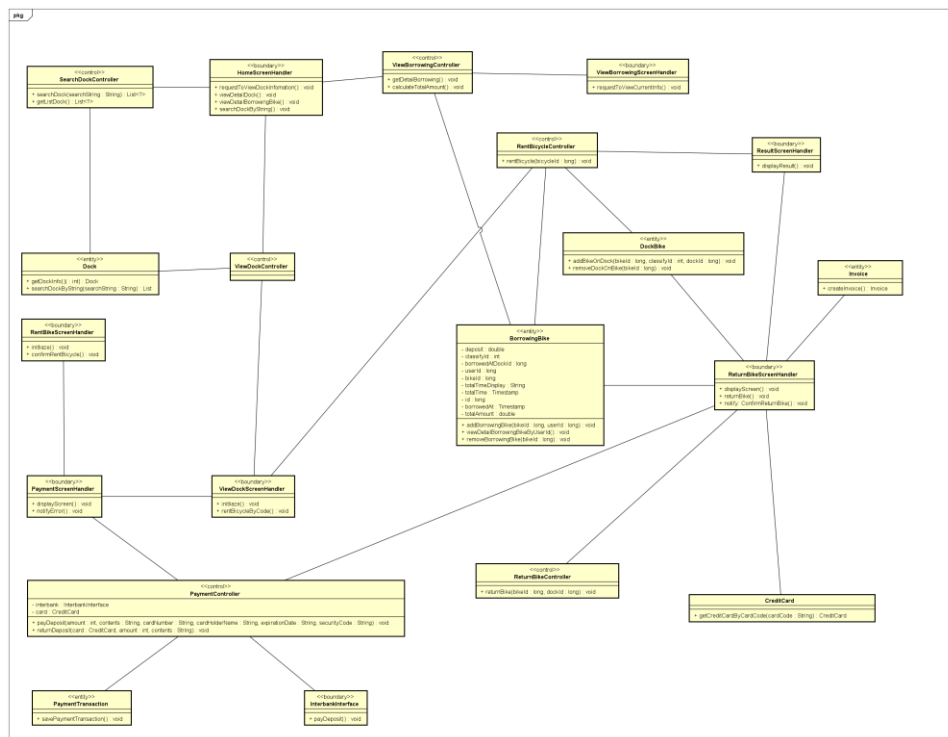
3.3.5 Usecase Xem xe đang thuê



3.3.6 Usecase Trả xe



3.4 Unified Analysis Class Diagram



3.5 Security Software Architecture

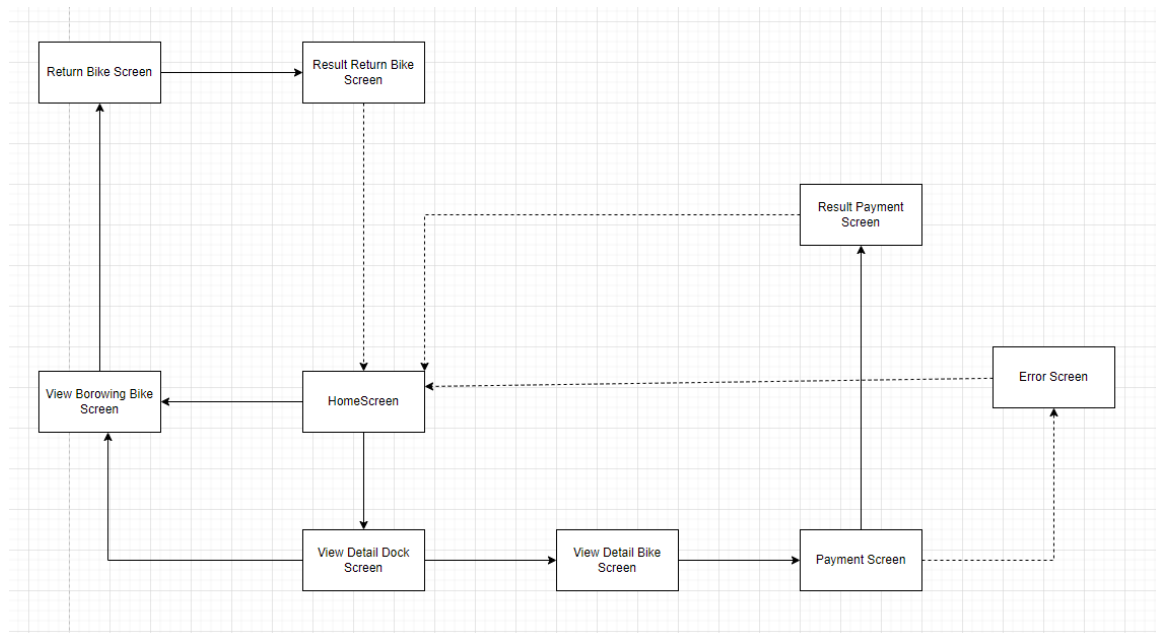
4 Detailed Design

4.1 User Interface Design

4.1.1 Screen Configuration Standardization

Kích thước giao diện: 1366x768

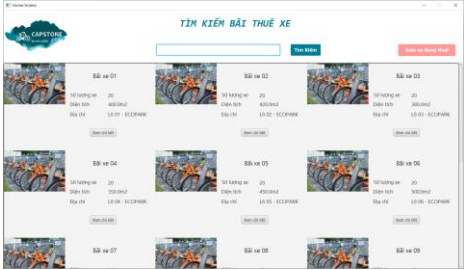
4.1.2 Screen Transition Diagrams



4.1.3 Screen Specifications

4.1.3.1 Màn trang chủ, tìm kiếm bãi xe

AIMS Software		Ngày tạo	Chấp nhận	Kiểm tra	Phụ trách
Screen Specification	Search Dock Screen	13/11/2021			Trần Hải Trung
		Control	Operation	Function	
		Text field tìm kiếm	Khởi tạo	Nhập thông tin tìm kiếm	

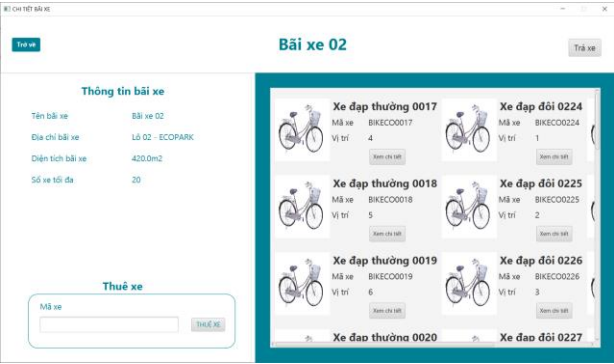
	Nút tìm kiếm	Click	Tìm kiếm bãi xe
	Nút xem xe đang thuê	Click	Hiển thị màn hình thông tin về xe đang thuê
	Khu vực bãi xe	Khởi tạo	Hiển thị danh sách các bãi xe
	Nút trả xe	Click	Chuyển về màn trả xe

Định nghĩa các trường thuộc tính

Items name	Number of digits	Type	Field attribute	Remarks
Tên bãi xe	100	Numerical		Căn giữa
Ô tìm kiếm	100	Numerical	Text Field	
Ảnh bãi xe		Image		
Số lượng xe	10	Number		Căn phải
Diện tích	10	Number		Căn phải
Địa chỉ	100	Numerical		Căn phải

4.1.3.2 Màn xem chi tiết bãi xe

AMIS Software	Date of creation	Approved by	Reviewed by	Person in charge
---------------	------------------	-------------	-------------	------------------

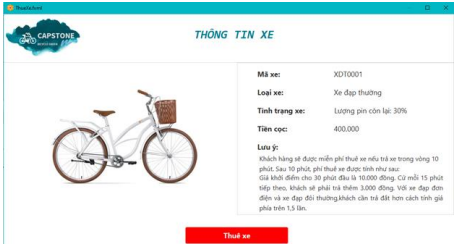
Screen specification	View homepage screen	07/11/2021			Trần Ngọc Phiên
	Control	Operation	Function		
	Vùng hiển thị thông tin bãi xe	Initial	Hiển thị các thông tin của bãi xe		
	Vùng hiển thị danh sách các xe	Initial	Hiển thị các xe có trong bãi xe		
	Button trả xe	Click	Hiện giao diện trả xe		
	TextFiel d nhập mã xe	Press	Nhập mã xe		
	Button thuê xe	Click	Hiện giao diện thuê xe		

Định nghĩa các trường thuộc tính

Screen name	View cart			
Item name	Number of digits(bytes)	Type	Field attribute	Remarks
Tên bãi xe	50	Character	Blue	right-justified
Địa chỉ	50	Character	Blue	Right-justified
Diện tích bãi xe	10	Numeral	Blue	Right-justified, định dạng x,xxxm2

Số xe tối đa	10	Numeral	Blue	Right-justified
Tên xe	50	Numeral	Black	Left-justified
Vị trí xe	10	Character	Black	Left-justified

4.1.3.3 Màn xem thông tin xe, thuê xe


AIMS Software		Ngày tạo	Chấp nhận	Kiểm tra	Phụ trách
Screen Specification	Order Bike Screen	15/11/2021			Trần Hải Trung
		Control	Operation	Function	
		Khu vực ảnh xe	Khởi tạo	Hình ảnh của chiếc xe khách muốn thuê	
		Khu vực thông tin xe	Khởi tạo	Hiển thị thông tin của chiếc xe khách muốn thuê và cách tính phí thuê xe	
		Nút thuê xe	Click	Xác nhận chuyển tới giao diện thanh toán	

Định nghĩa các trường thuộc tính

Items name	Number of digits	Type	Field attribute	Remarks
Mã xe	10	Character	black	Left-justified
Loại xe	20	Character	black	Left-justified
Tình trạng xe	50	Character	black	Left-justified

Tiền cọc	10	Number	black	Left-justified
----------	----	--------	-------	----------------

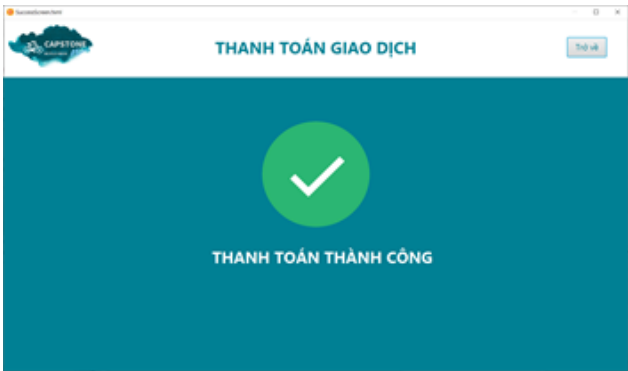
4.1.3.4 Màn nhập thông tin thẻ

AMIS Software		Date of creation	Approved by	Reviewed by	Person in charge
Screen specification	Payment Information Screen	07/11/2021			Trần Ngọc Phiên
		Control	Operation	Function	
		Vùng hiển thị ô TextField	Press	Người dùng nhập thông tin thẻ	
		Button thanh toán	Click	Yêu cầu thanh toán	


Định nghĩa các trường thuộc tính

Items name	Number of digits	Type	Field attribute	Remarks
Tên chủ thẻ	50	Character		
Mã thẻ	20	Character		
Ngày hết hạn	10	Character		
Mã bảo mật	10	Number		
Nội dung	200	Character		

4.1.3.5 Màn kết quả thanh toán

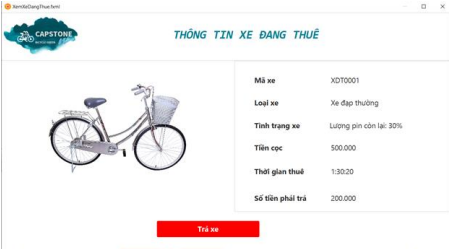
AMIS Software		Date of creation	Approved by	Reviewed by	Person in charge
Screen specification	Success payment	07/11/2021			Trần Ngọc Phiên
		Control	Operation	Function	
		Vùng hiển thị kết quả thanh toán	Initial	Hiển thị thanh toán thành công	

Màn Lỗi

AMIS Software		Date of creation	Approved by	Reviewed by	Person in charge
Screen specification	Fail Payment	07/11/2021			Trần Ngọc Phiên
		Control	Operation	Function	
		Vùng hiển thị kết quả thanh toán	Initial	Hiển thị lỗi thanh toán	

4.1.3.6 Màn xem xe đang thuê

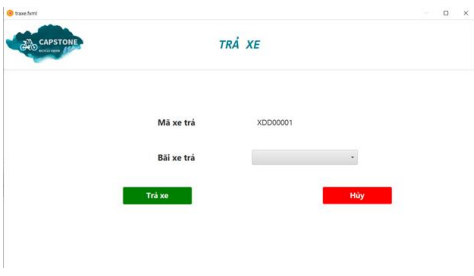
AIMS Software	Ngày tạo	Chấp nhận	Kiểm tra	Phụ trách
---------------	----------	-----------	----------	-----------

Screen Specification	Return Bike Screen	13/11/2021			Trần Hải Trung
		Control	Operation	Function	
		Khu vực ảnh xe	Khởi tạo	Hình ảnh của chiếc xe đang thuê	
		Khu vực thông tin xe	Khởi tạo	Hiển thị thông tin của chiếc xe đang thuê	
		Nút trả xe	Click	Xác chuyển tới giao diện trả xe	

Định nghĩa các trường thuộc tính

Items name	Number of digits	Type	Field attribute	Remarks
Mã xe		Numerical		
Loại xe		Numerical		
Tình trạng xe		Numerical		
Tiền cọc		Number		
Thời gian thuê		Time		
Số tiền phải trả		Number		

4.1.3.7 Màn trả xe

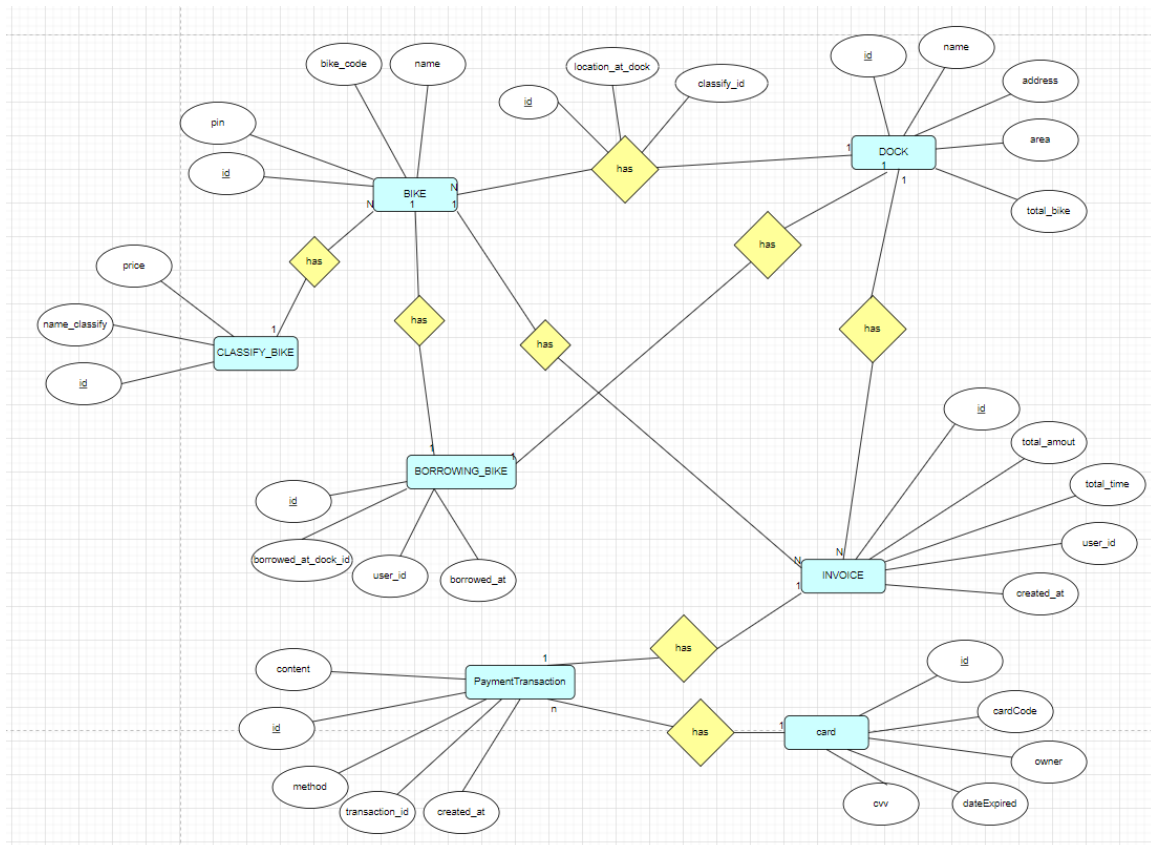
AIMS Software		Ngày tạo	Chấp nhận	Kiểm tra	Phụ trách
Screen Specification	Return Bike Screen	13/11/2021			Trần Hải Trung
		Control	Operation	Function	
		Combobox	Khởi tạo	Chọn bãi trả xe	
		Nút trả xe	Click	Xác nhận trả xe	
		Nút hủy	Click	Hủy trả xe, trở về màn hình trước đó	

Items name	Number of digits	Type	Field attribute	Remarks
Mã xe trả		Numeral		
Bãi xe trả		Combobox	Combobox	

4.2 Data Modeling

4.2.1 Conceptual Data Modeling

<E-R Diagram image and description of entities and relationships>



4.2.2 Database Design

4.2.2.1 Database Management System

<Specify what is the decision of Database Management System (DBMS) and give some description of the DBMS>

Trong bài tập này, chúng em sử dụng cơ sở dữ liệu MySQL. Chúng em sử dụng hệ quản trị cơ sở dữ liệu này bởi vì đây là hệ quản trị cơ sở dữ liệu hoàn toàn miễn phí, mọi người đều có thể cài đặt từ trang chủ. MySQL là cơ sở dữ liệu tốc độ cao, ổn định hoạt động trên nhiều hệ điều hành, cung cấp một hệ thống lớn các hàm tiện ích.

Một vài mô tả về hệ quản trị cơ sở dữ liệu MySQL:

- MySQL là một hệ thống quản trị cơ sở dữ liệu mã nguồn mở (gọi tắt là RDBMS) hoạt động theo mô hình client-server. Với RDBMS là viết tắt của Relational Database Management System. MySQL được tích hợp apache, PHP
- MySQL thích hợp với các ứng dụng có truy cập cơ sở dữ liệu trên Internet nhờ vào tốc độ cũng như tính bảo mật cao.

Ưu điểm:

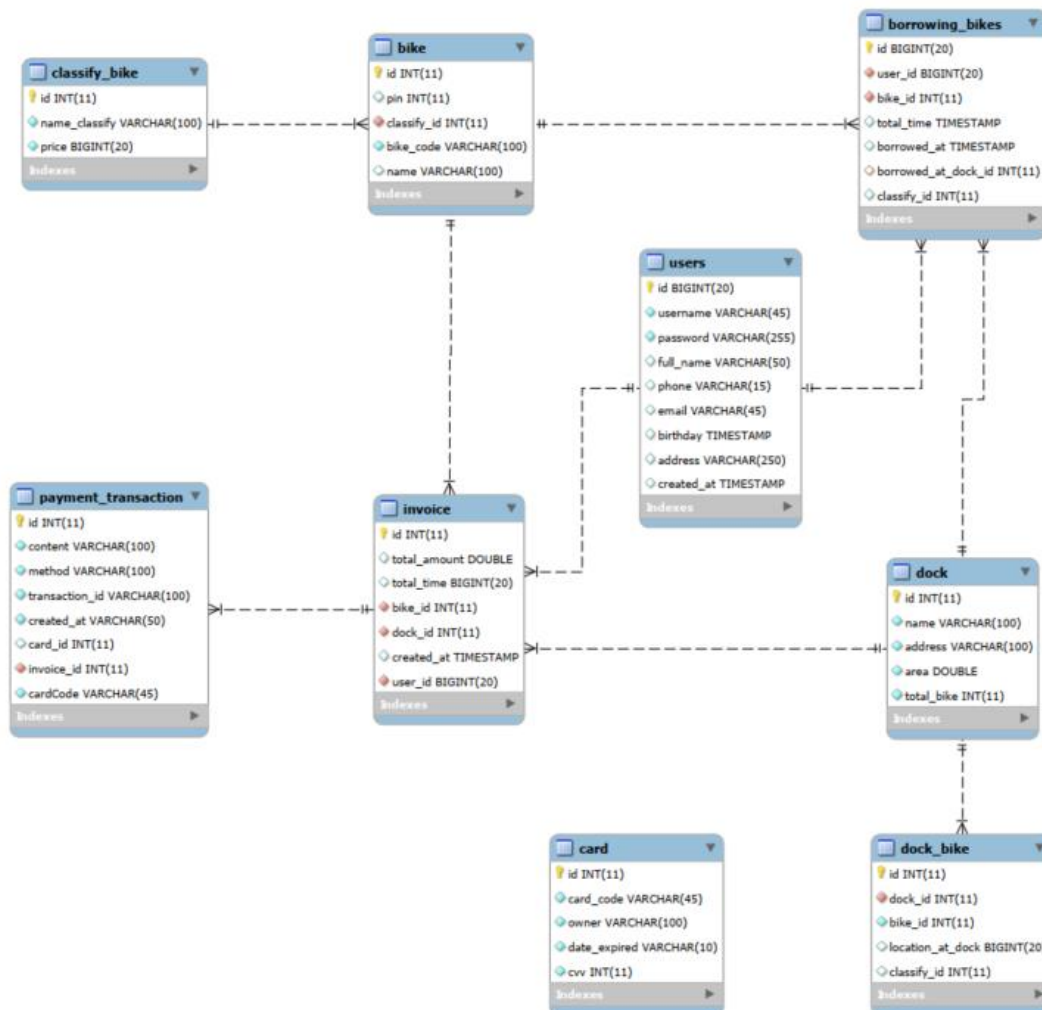
- Sử dụng dễ dàng

- Độ bảo mật cao
- Đa dạng tính năng
- Khả năng mở rộng và mạnh mẽ
- Nhanh chóng
- Sử dụng miễn phí

Nhược điểm:

- MySQL có thể bị khai thác để chiếm quyền điều khiển
- MySQL không được tích hợp để sử dụng cho các hệ thống lớn cần quản lý lượng dữ liệu khổng lồ. Ví dụ như các hệ thống siêu thị trên toàn quốc, ngân hàng, quản lý thông tin dân số cả nước,...

4.2.2.2 Database Diagram



4.2.2.3 Database Detail Design

- Classify_bike

#	PK	FK	Column name	Data type	Default value	Mandatory	Description
1	x		Id	INTEGER		Yes	ID, auto increment
2			name_classification	VARCHAR(45)		Yes	Tên loại xe
3			price	BIGINT		Yes	Giá tiền cọc

- **Bike**

#	PK	FK	Column name	Data type	Default value	Mandatory	Description
1	X		Id	INTEGER		Yes	ID của xe, auto increment
2			pin	INTEGER		No	Thời lượng pin(với xe đạp điện)
3		x	classify_id	INTEGER		Yes	id của loại xe
4			name	VARCHAR(45)		Yes	Tên xe
5			Bike_code	VARCHAR(50)		Yes	Mã xe

- **Dock**

#	PK	FK	Column name	Data type	Default value	Mandatory	Description
1	X		Id	INTEGER		Yes	ID của bãi xe
2			name	VARCHAR(45)		Yes	Tên bãi xe
3			address	VARCHAR(45)		Yes	Địa chỉ bãi xe
4			area	DOUBLE		Yes	Diện tích bãi xe

5			total_bike	INT		Yes	Tổng số xe trong bãi xe
---	--	--	------------	-----	--	-----	-------------------------

- **Dock_bike**

#	PK	FK	Column name	Data type	Default value	Mandatory	Description
1	x		Id	INTEGER		Có	ID của bảng dock_bike
2		x	dock_id	INTEGER		Có	ID của bãi xe
3		x	bike_id	INTEGER		Có	ID của xe
4			location_dock	INTEGER		Yes	Vị trí của xe trong bãi xe
5			Classify_id	INTEGER		yes	Vị trí của loại xe nào

- **invoice**

#	PK	FK	Column name	Data type	Default value	Mandatory	Description
1	x		Id	INTEGER		Yes	ID của hóa đơn
2			total_amount	INT		Yes	Tổng số tiền
3			total_time	BIGINT		Yes	Tổng thời lượng mượn xe dạng milisenconds
4		x	bike_id	INTEGER		Yes	ID của xe
5		x	dock_id	INTEGER		Yes	ID của bãi xe
6		x	User_id	BIGINT		Yes	Id của khách hàng thuê xe
7			createdAt	DATETIME		Yes	Thời gian tạo

- **Card**

#	PK	FK	Column name	Data type	Default value	Mandatory	Description
1	x		Id	INTEGER		Yes	ID, auto increment
2			CardCode	VARCHAR(45)		Yes	Số thẻ
3			Owner	VARCHAR(45)		Yes	Chủ sở hữu thẻ
4			cvvCode	VARCHAR(3)		Yes	CVV code
5			dateExpired	VARCHAR(4)		Yes	Ngày hết hạn

- **PaymentTransaction**

#	PK	FK	Column name	Data type	Default value	Mandatory	Description
1	x		Id	INTEGER		Yes	ID, auto increment
2			CreatedAt	DATETIME		Yes	Ngày tạo giao dịch
3			Content	VARCHAR(45)		Yes	Nội dung giao dịch
4			Method	VARCHAR(45)		Yes	Phương thức thanh toán
5			CardCode	VARCHAR(45)		Yes	Mã số thẻ
6		x	InvoiceId	INTEGER		Yes	InvoiceID
7			transactionId	VARCHAR(100)		Yes	Mã giao dịch

- **BorrowingBike**

#	PK	FK	Column name	Data type	Default value	Mandatory	Description
1	x		Id	BIGINT		Yes	ID, auto increment
2		x	User_id	BIGINT		Yes	Id người mượn xe
3		x	Bike_id	INT		Yes	Id Xe đang mượn
4			Total_time	BIGINT			Tổng thời gian mượn tới thời điểm hiện tại
5			Borrowed_At	TIMESTAMP			Thời điểm mượn xe
6		x	Borrowed_at_dock_id	INTEGER		Yes	Mã id của bãi xe mượn

- USERS

#	PK	FK	Column name	Data type	Default value	Mandatory	Description
1	x		Id	BIGINT		Yes	ID, auto increment
2			username	VARCHAR(45)		Yes	Tên tài khoản đăng nhập
3			password	VARCHAR(255)		Yes	Mật khẩu đăng nhập
4			Full_name	VARCHAR(50)			Họ và tên user
5			phone	VARCHAR(15)			Số điện thoại

6			Email	VARCHAR(45)			Email
7			Birthday	TIMESTAMP			Ngày sinh
8			address	VARCHAR(250)			Địa chỉ
9			Created_at	TIMESTAMP			Thời gian tại tài khoản

Database script

- Bảng `BIKE`

```
CREATE TABLE `bike` (
  `id` int(11) NOT NULL,
  `pin` int(11) DEFAULT NULL,
  `classify_id` int(11) NOT NULL,
  `bike_code` varchar(100) COLLATE utf8_bin NOT NULL,
  `name` varchar(100) COLLATE utf8_bin DEFAULT NULL,
  PRIMARY KEY (`id`),
  KEY `id` (`id`),
  KEY `bike_ibfk_2` (`classify_id`),
  CONSTRAINT `bike_ibfk_2` FOREIGN KEY (`classify_id`) REFERENCES
`classify_bike` (`id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_bin;
```

- Bảng `borrowing_bikes`

```
CREATE TABLE `borrowing_bikes` (
  `id` bigint(20) NOT NULL AUTO_INCREMENT,
  `user_id` bigint(20) NOT NULL,
  `bike_id` int(11) NOT NULL,
```

```

`total_time` bigint(20) DEFAULT NULL,
`borrowed_at` timestamp NULL DEFAULT NULL,
`borrowed_at_dock_id` int(11) DEFAULT NULL,
`classify_id` int(11) DEFAULT NULL,
PRIMARY KEY (`id`),
KEY `fk_borrowing_user_id` (`user_id`),
KEY `fk_borrowing_bike_id` (`bike_id`),
KEY `fk_borrowing_dock_id` (`borrowed_at_dock_id`),
CONSTRAINT `fk_borrowing_bike_id` FOREIGN KEY (`bike_id`) REFERENCES
`bike` (`id`),
CONSTRAINT `fk_borrowing_dock_id` FOREIGN KEY (`borrowed_at_dock_id`)
REFERENCES `dock` (`id`),
CONSTRAINT `fk_borrowing_user_id` FOREIGN KEY (`user_id`) REFERENCES
`users` (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=50 DEFAULT CHARSET=utf8
COLLATE=utf8_bin;

```

- Bảng `Card`

```

CREATE TABLE `card` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `card_code` varchar(45) COLLATE utf8_bin NOT NULL,
  `owner` varchar(100) COLLATE utf8_bin NOT NULL,
  `date_expired` varchar(10) COLLATE utf8_bin NOT NULL,
  `cvv` int(11) NOT NULL,
  PRIMARY KEY (`id`),
  UNIQUE KEY `card_code_UNIQUE` (`card_code`)
) ENGINE=InnoDB AUTO_INCREMENT=21 DEFAULT CHARSET=utf8
COLLATE=utf8_bin;

```

- Bảng `classify_bike`

```
CREATE TABLE `classify_bike` (
  `id` int(11) NOT NULL,
  `name_classify` varchar(100) COLLATE utf8_bin NOT NULL,
  `price` bigint(20) NOT NULL,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_bin;
```

- **Bảng `Dock`**

```
CREATE TABLE `dock` (
  `id` int(11) NOT NULL,
  `name` varchar(100) COLLATE utf8_bin NOT NULL,
  `address` varchar(100) COLLATE utf8_bin NOT NULL,
  `area` double NOT NULL,
  `total_bike` int(11) NOT NULL,
  PRIMARY KEY (`id`),
  KEY `id` (`id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_bin;
```

- **Bảng `dock_bike`**

```
CREATE TABLE `dock_bike` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `dock_id` int(11) NOT NULL,
  `bike_id` int(11) NOT NULL,
  `location_at_dock` bigint(20) DEFAULT NULL,
  `classify_id` int(11) DEFAULT NULL,
  PRIMARY KEY (`id`),
  KEY `dock_bike_ibfk_1` (`dock_id`),
  CONSTRAINT `dock_bike_ibfk_1` FOREIGN KEY (`dock_id`) REFERENCES `dock`
  (`id`)
```

```
) ENGINE=InnoDB AUTO_INCREMENT=2753 DEFAULT CHARSET=utf8  
COLLATE=utf8_bin;
```

- **Bảng `invoice`**

```
CREATE TABLE `invoice` (  
  `id` int(11) NOT NULL AUTO_INCREMENT,  
  `total_amount` double DEFAULT NULL,  
  `total_time` bigint(20) DEFAULT NULL,  
  `bike_id` int(11) NOT NULL,  
  `dock_id` int(11) NOT NULL,  
  `created_at` timestamp NULL DEFAULT NULL,  
  `user_id` bigint(20) NOT NULL,  
  PRIMARY KEY (`id`),  
  KEY `invoice_ibfk_1` (`dock_id`),  
  KEY `invoice_ibfk_2` (`bike_id`),  
  KEY `fk_user_id` (`user_id`),  
  CONSTRAINT `fk_user_id` FOREIGN KEY (`user_id`) REFERENCES `users` (`id`),  
  CONSTRAINT `invoice_ibfk_1` FOREIGN KEY (`dock_id`) REFERENCES `dock`  
  (`id`),  
  CONSTRAINT `invoice_ibfk_2` FOREIGN KEY (`bike_id`) REFERENCES `bike`  
  (`id`)  
) ENGINE=InnoDB AUTO_INCREMENT=74 DEFAULT CHARSET=utf8  
COLLATE=utf8_bin;
```

- **Bảng `payment_transaction`**

```
CREATE TABLE `payment_transaction` (  
  `id` int(11) NOT NULL AUTO_INCREMENT,  
  `content` varchar(100) COLLATE utf8_bin NOT NULL,  
  `method` varchar(100) COLLATE utf8_bin NOT NULL,  
  `transaction_id` varchar(100) COLLATE utf8_bin NOT NULL,
```



```

`created_at` varchar(50) COLLATE utf8_bin NOT NULL,
`card_id` int(11) DEFAULT NULL,
`invoice_id` int(11) NOT NULL,
`cardCode` varchar(45) COLLATE utf8_bin NOT NULL,
PRIMARY KEY (`id`),
KEY `invoice_id` (`invoice_id`),
CONSTRAINT `fk_invoice_id` FOREIGN KEY (`invoice_id`) REFERENCES
`invoice` (`id`) ON DELETE NO ACTION ON UPDATE NO ACTION
) ENGINE=InnoDB AUTO_INCREMENT=16 DEFAULT CHARSET=utf8
COLLATE=utf8_bin;

```

- **Bảng `users`**

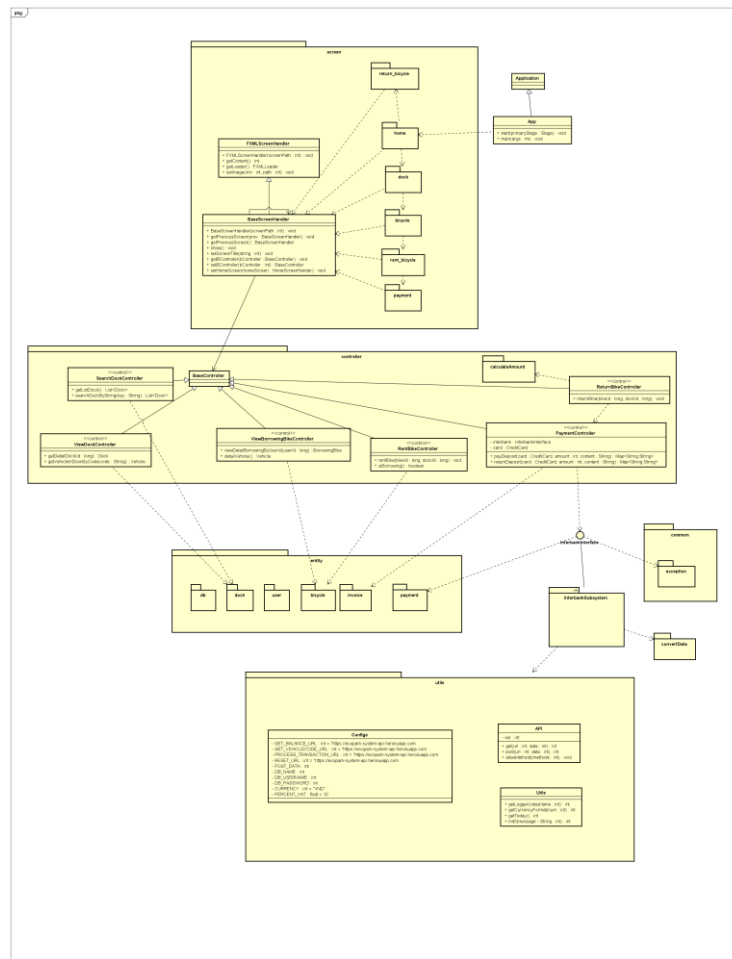
```

CREATE TABLE `users` (
`id` bigint(20) NOT NULL,
`username` varchar(45) COLLATE utf8_bin NOT NULL,
`password` varchar(255) COLLATE utf8_bin NOT NULL,
`full_name` varchar(50) COLLATE utf8_bin DEFAULT NULL,
`phone` varchar(15) COLLATE utf8_bin DEFAULT NULL,
`email` varchar(45) COLLATE utf8_bin DEFAULT NULL,
`birthday` timestamp NULL DEFAULT NULL,
`address` varchar(250) COLLATE utf8_bin DEFAULT NULL,
`created_at` timestamp NULL DEFAULT NULL,
PRIMARY KEY (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=18 DEFAULT CHARSET=utf8
COLLATE=utf8_bin;

```

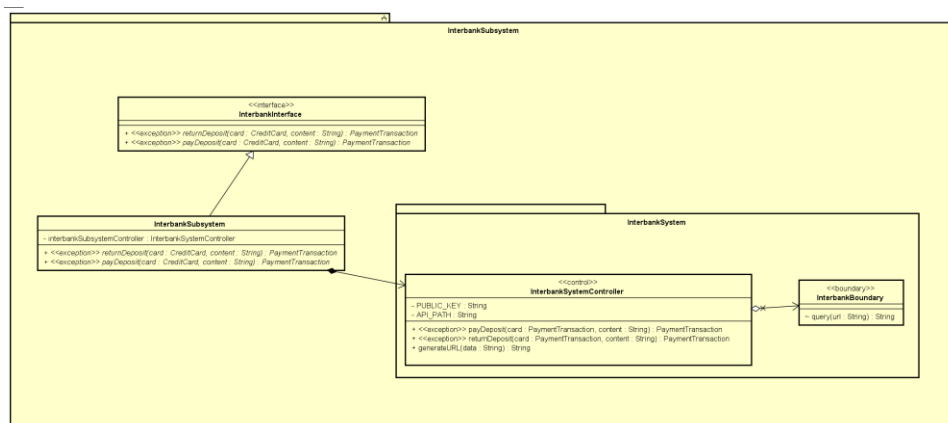
4.3 Class Design

4.3.1 General Class Diagram

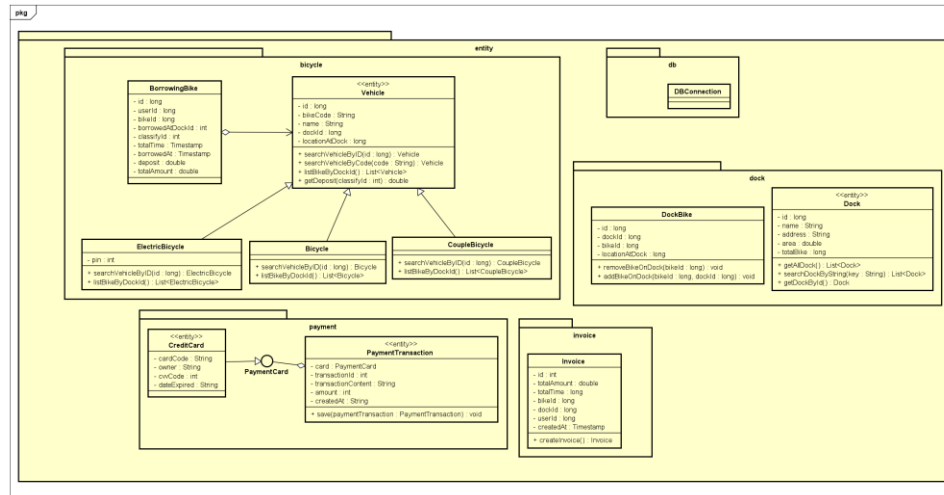


4.3.2 Class Diagrams

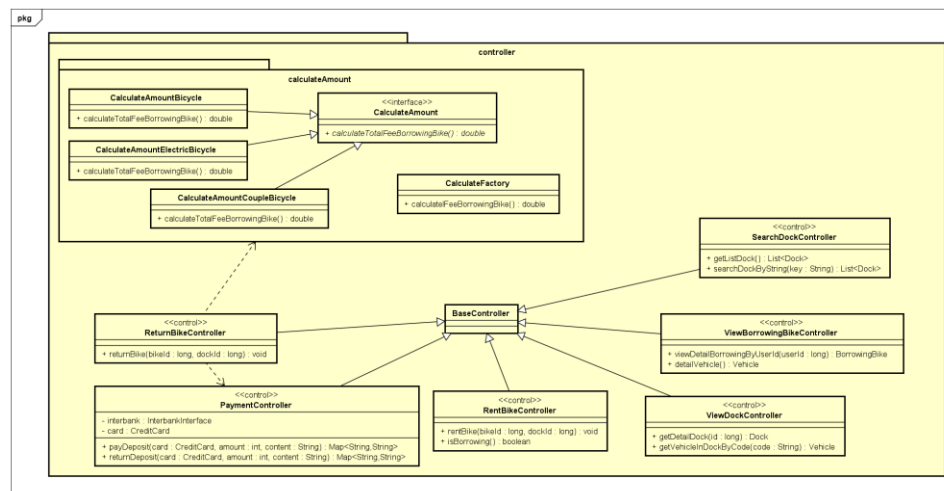
4.3.2.1 Class Diagram for Subsystem Interbank



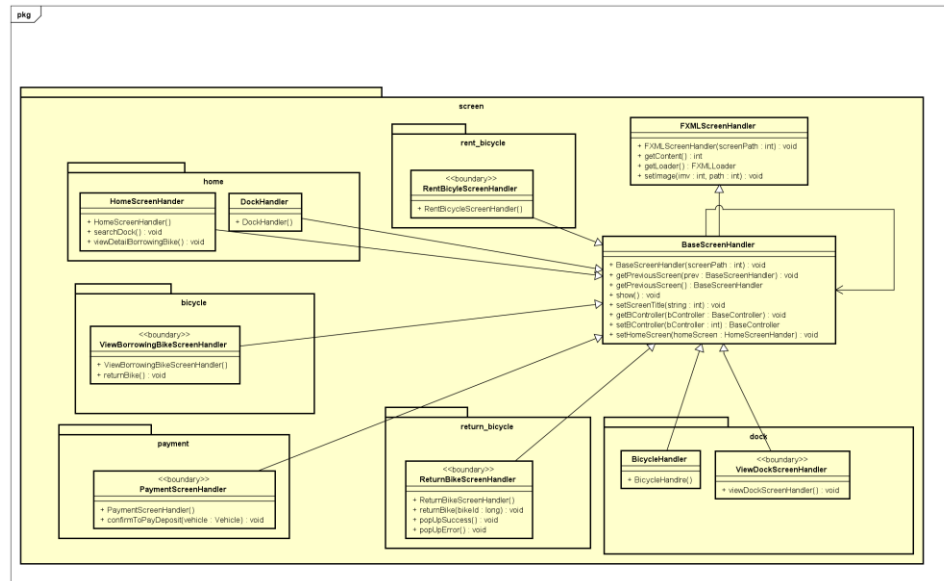
4.3.2.2 Class Diagram for package entity



4.3.2.3 Class Diagram for package controller



4.3.2.4 Class Diagram for package screen



4.3.3 Class Design

<Detail design for each class>

4.3.3.1 Class “SearchDockController”

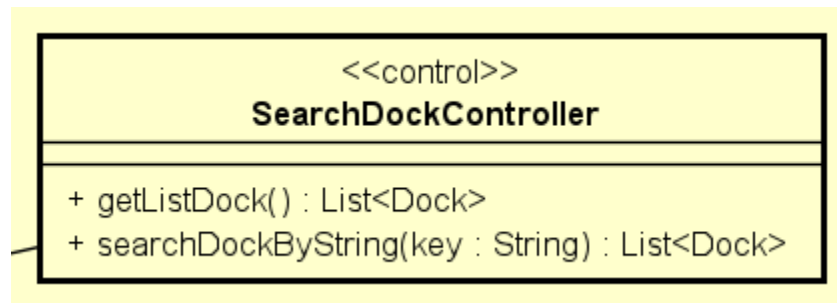


Table 1. Example of operation design

#	Name	Return type	Description (purpose)
1	getListDock	List<Dock>	Lấy thông tin của tất cả các bãi xe
2	searchDockByString	List<Dock>	Tìm các bãi xe phù hợp với nội dung từ khóa

Parameter:

- key: Từ khóa tìm kiếm bãi xe

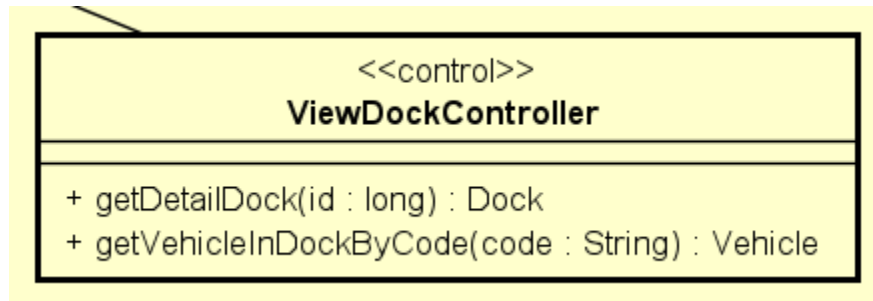
Method

Không

State

Không

4.3.3.2 Class “ViewDockController”



Operation

#	Name	Return type	Description (purpose)
1	getDetailDock	Dock	Lấy thông tin của 1 bãi xe dựa vào id
2	getVehicleInDockByCode	Vehicle	Lấy thông tin của 1 xe trong bãi xe bằng mã xe

Parameter:

- id: id của bãi xe
- code: code của xe

Exception:

- Không

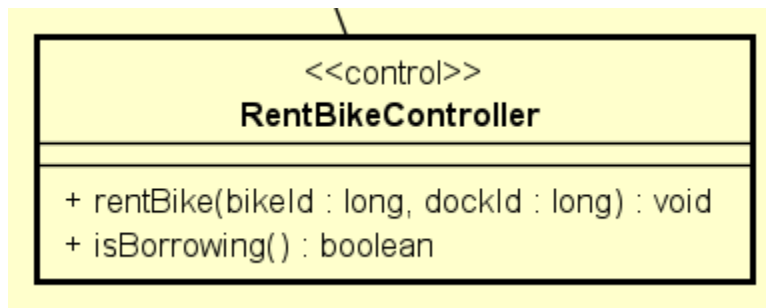
Method

Không

State

Không

4.3.3.3 Class “RentBikeController”



Operation

#	Name	Return type	Description (purpose)
1	rentBike	void	Thuê xe
2	isBorrowing	boolean	Kiểm tra đang có xe mượn hay không

Parameter:

- bikeId: id của xe
- dockId: id của bãi xe

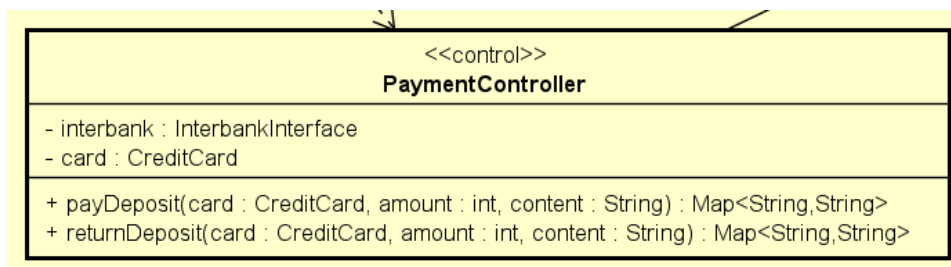
Method

Không

State

Không

4.3.3.4 Class “PaymentController”



Attribute

#	Name	Data type	Default value	Description
1	Card	CreditCard	NULL	Thẻ để thanh toán

2	Interbank	InterbankInterface	NULL	Interbank subsystem
---	-----------	--------------------	------	---------------------

Operation

#	Name	Return type	Description (purpose)
1	payDeposit	Map<String,String>	Thanh toán tiền đặt cọc và trả về kết quả
2	returnDeposit	Map<String,String>	Hoàn tiền đặt cọc và trả về kết quả

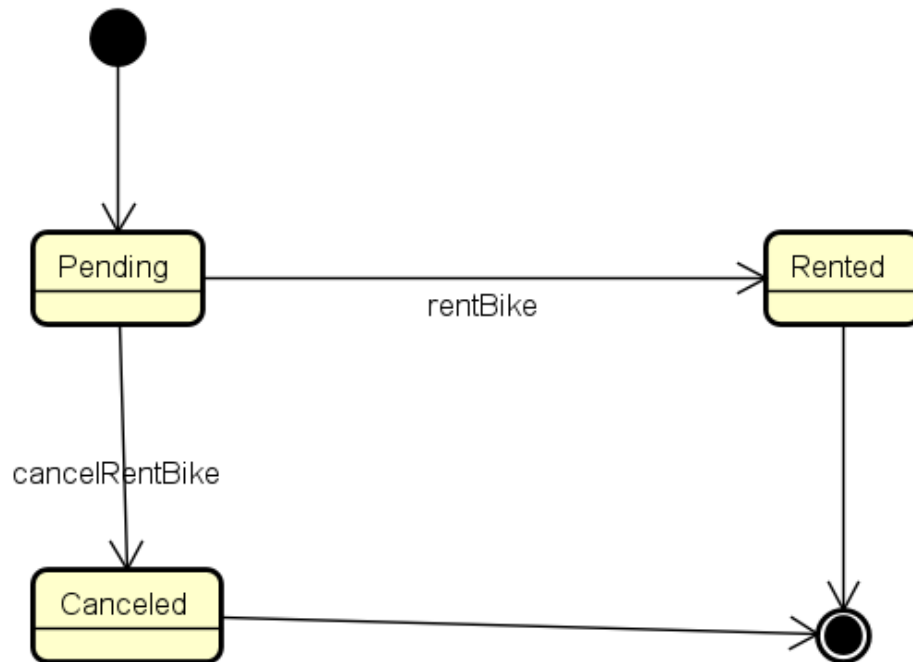
Parameter:

- Card – Thẻ dùng để giao dịch
- Contents – Nội dung giao dịch
- Amount – Số tiền giao dịch

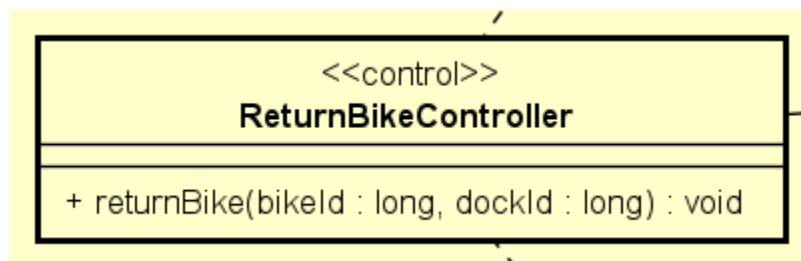
Exception:

- Không

State cho đối tượng Bicycle.



4.3.3.5 Class “ReturnBikeController”



Operation

#	Name	Return type	Description (purpose)
1	returnBike	void	Trả xe vào bãi xe

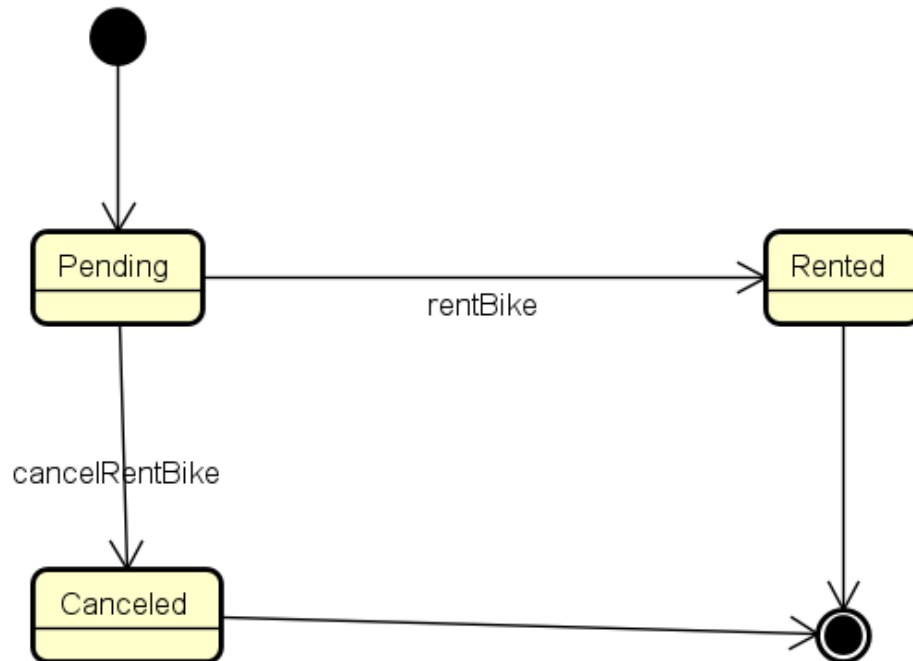
Parameter:

- bikeId– Id của xe
- dockId– Id của bãi xe (bãi xe dùng để trả xe)

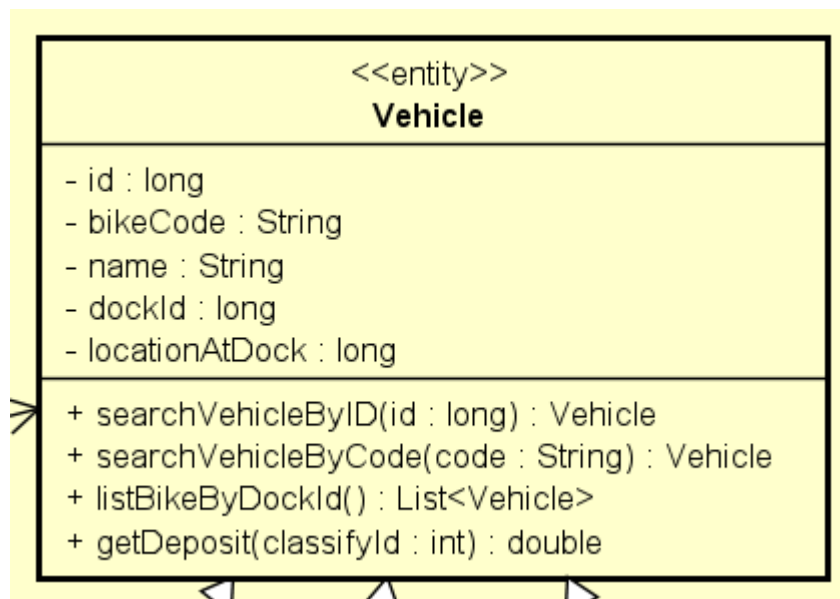
Exception:

- Không

State cho đối tượng Bicycle.



4.3.3.6 Class “Vehicle”



Example of attribute design

#	Name	Data type	Default value	Description
1	id	long	null	ID của xe
2	bikeCode	String	“	Mã xe
3	name	String	“”	Tên xe
4	dockId	long	0	ID của bãi xe
5	locationAtDock	long	0	Vị trí trong bãi xe

Table 2. Example of operation design

#	Name	Return type	Description (purpose)
1	searchVehicleById	Vehicle	Tìm kiếm xe theo ID
2	searchVehicleByCode	Vehicle	Tìm kiếm xe theo mã xe
3	listBikeByDockId	List<Vehicle>	Lấy các xe trong bãi xe
4	getDeposit	double	Lấy tiền đặt cọc

Parameter:

- BikeID : mã id xe
- Code: mã xe
- DockID: id của bãi xe
- ClassifyId: id của loại xe

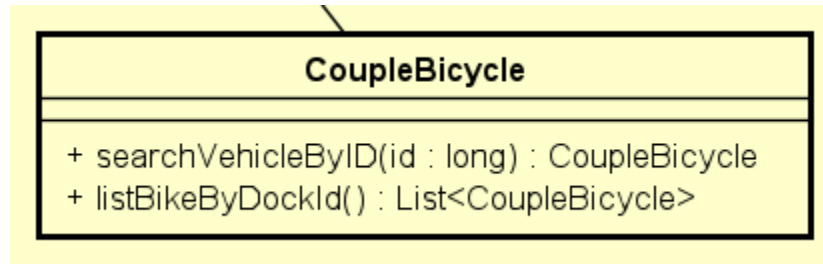
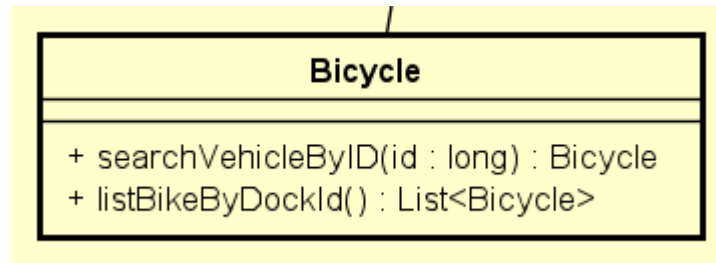
Method

Không

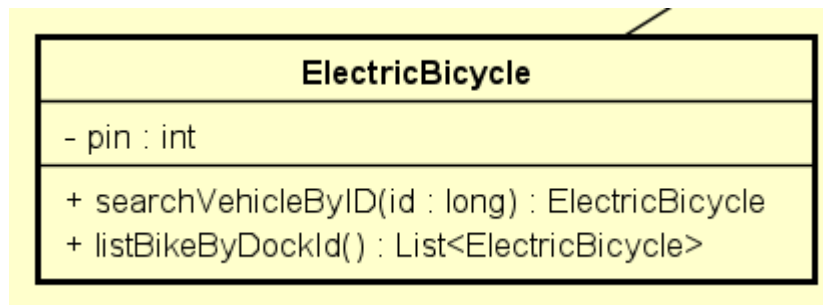
State

Không

4.3.3.7 Class “Bicycle, CoupleBicycle” tương tự như Vehicle



4.3.3.8 Class “ElectricBicycle”



#	Name	Data type	Default value	Description
1	pin	integer	null	Pin của xe

Table 3. Example of operation design

#	Name	Return type	Description (purpose)
1	searchVehicleById	Vehicle	Tìm kiếm xe theo ID
3	listBikeByDockId	List<Vehicle>	Lấy các xe trong bãi xe

Parameter:

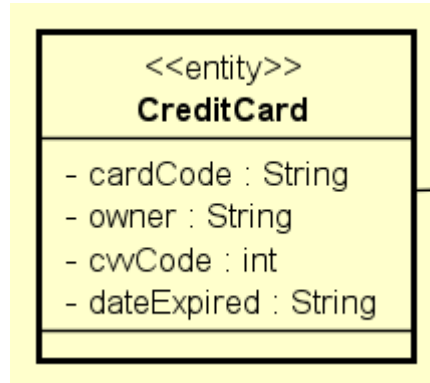
- BikeID : mã id xe

Method

Không

State

4.3.3.9 Class “CreditCard”



Attribute

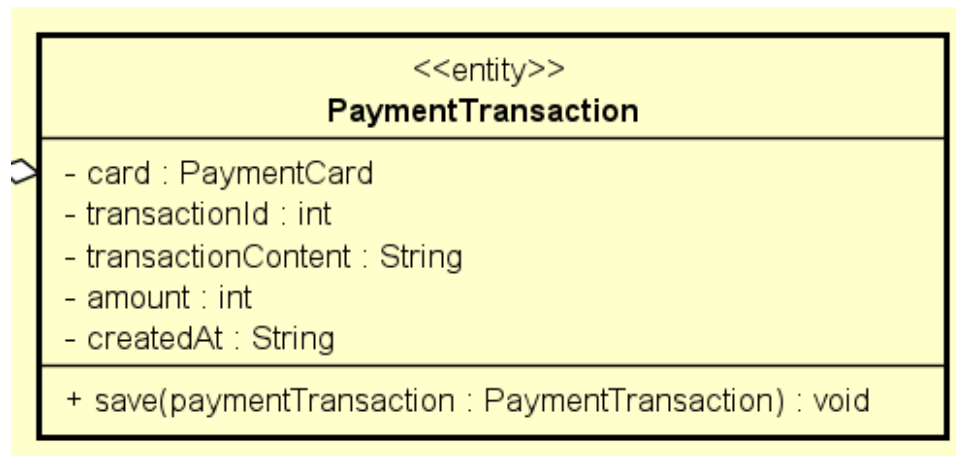
#	Name	Data type	Default value	Description
1	cardNumber	String		Số thẻ
2	cardHolder Name	String		Tên chủ thẻ
3	issuingBank	String		Ngân hàng phát hành
4	expiration Date	Timestamp		Thời hạn hết hạn
5	cardSecurityCode	int		Mã số bảo mật.

Operation

Method

State

4.3.3.10 Class “PaymentTransaction”



Attribute

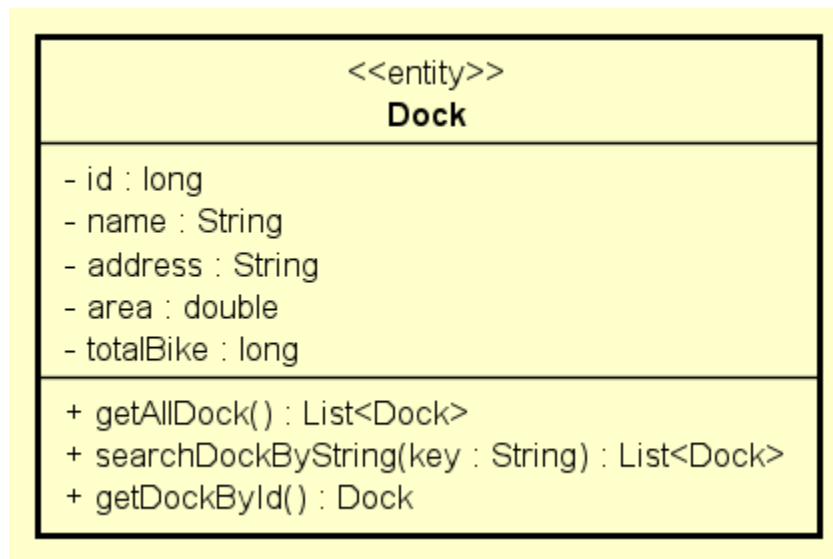
#	Name	Data type	Default value	Description
1	card	PaymentCard		Thẻ dùng để thanh toán
2	transaction ID	int		Id giao dịch
3	transaction Content	String		Nội dung giao dịch
4	amount	int		Số tiền giao dịch
5	createdAt	Timestamp		Thời gian tạo giao dịch

Operation

Method

- savePaymentTransaction: Lưu lại thông tin giao dịch

4.3.3.11 Class “Dock”



Example of attribute design

#	Name	Data type	Default value	Description
1	id	long	null	ID của bãi xe
2	name	String	“	Tên bãi xe
3	address	String	“”	Địa chỉ bãi xe
4	area	double	0	Diện tích bãi xe
5	totalBike	long	0	Tổng số xe trong bãi xe

Table 4. Example of operation design

#	Name	Return type	Description (purpose)
1	GetAllDock()	List<Dock>	Lấy thông tin tất cả bãi xe
2	searchDockByString	List<Dock>	Tìm kiếm bãi xe
3	getDockById	Dock	Lấy thông tin bãi xe theo ID của bãi xe

Parameter:

- Key: từ khóa tìm kiếm

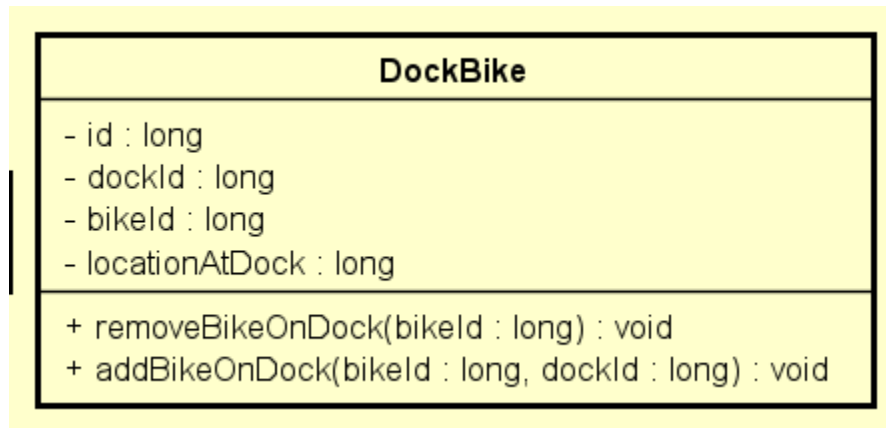
Method

Không

State

Không

4.3.3.12 Class “DockBike”



Example of attribute design

#	Name	Data type	Default value	Description
1	id	long	null	ID của dock bike
2	dockID	long	“	ID của bãi xe
3	name	String	“”	Tên xe
4	dockId	long	0	ID của bãi xe
5	locationAtDock	long	0	Vị trí trong bãi xe

Table 5. Example of operation design

#	Name	Return type	Description (purpose)
1	removeBikeOnDock	void	Xóa xe khỏi bãi xe
2	addBikeOnDock	void	Thêm xe vào bãi xe

Parameter:

- BikeID : mã id xe
- DockID: id của bãi xe

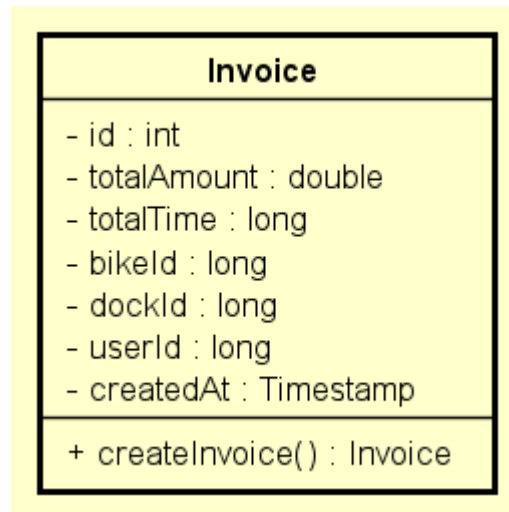
Method

Không

State

Không

4.3.3.13 Class “Invoice”



Example of attribute design

#	Name	Data type	Default value	Description
1	id	long	null	ID của hóa đơn
2	totalAmount	double	“	Tổng tiền
3	totalTime	long	“”	Tổng thời gian
4	bikeId	long	0	Id xe
5	dockID	long	0	Id bãi xe
6	userId	long		ID của người dùng
7	createdAt	Timestamp		Thời gian tạo hóa đơn

Table 6. Example of operation design

#	Name	Return type	Description (purpose)
1	createInvoice	Invoice	Tạo hóa đơn
2	addBikeOnDock	void	Thêm xe vào bãi xe

Parameter:

- BikeID : mã id xe
- DockID: id của bãi xe

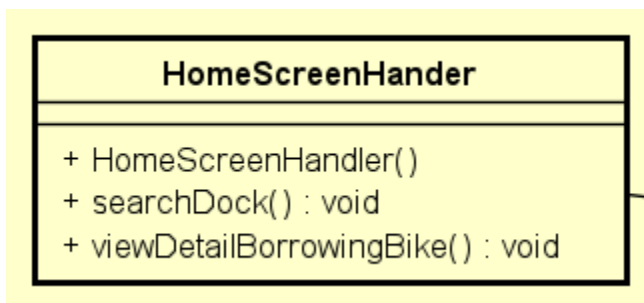
Method

Không

State

Không

4.3.3.14 Class “HomeScreenHandler”



Attribute

Table 7. Example of operation design

#	Name	Return type	Description (purpose)
1	HomeScreenHandler		Constructor khởi tạo, lấy dữ liệu các bãi xe từ db

2	searchDock	void	Tìm kiếm bãi xe
3	viewDetailBorrowingBike	void	Hiển thị xem xe đang thuê

Parameter:

- BikeID : mã id xe
- DockID: id của bãi xe

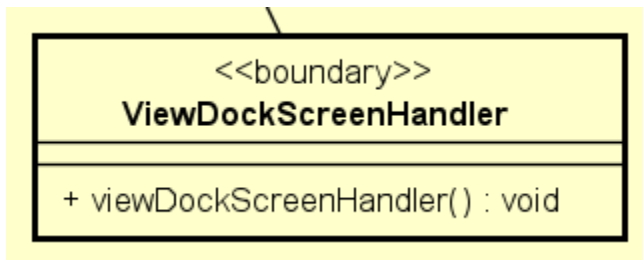
Method

Không

State

Không

4.3.3.15 Class “ViewDockScreenHandler”



Attribute

Table 8. Example of operation design

#	Name	Return type	Description (purpose)
1	ViewDockScreenHandler		Constructor khởi tạo, lấy dữ liệu các xe trong bãi xe

Parameter:

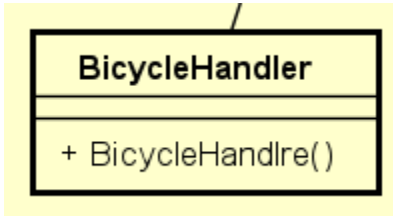
Method

Không

State

Không

4.3.3.16 Class “BicycleHandler”



Attribute

Table 9. Example of operation design

#	Name	Return type	Description (purpose)
1	BicycleHandler		Constructor khởi tạo, lấy dữ liệu xe và hiển thị lên giao diện
2			

Parameter:

- BikeID : mã id xe
- DockID: id của bãi xe

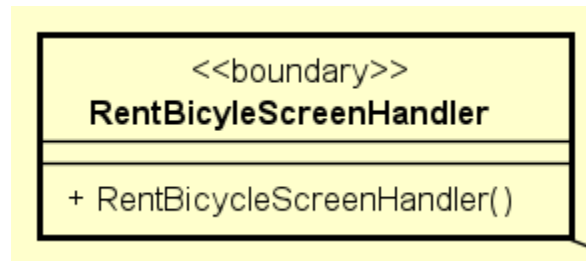
Method

Không

State

Không

4.3.3.17 Class “RentBicycleScreenHandler”



Attribute

Table 10. Example of operation design

#	Name	Return type	Description (purpose)
1	RentBicycleScreenHandler		Constructor khởi tạo, Hiển thị thông tin xe
2	requestRentBike		Hiển thị màn hình nhập thông tin thẻ để thanh toán

Parameter:

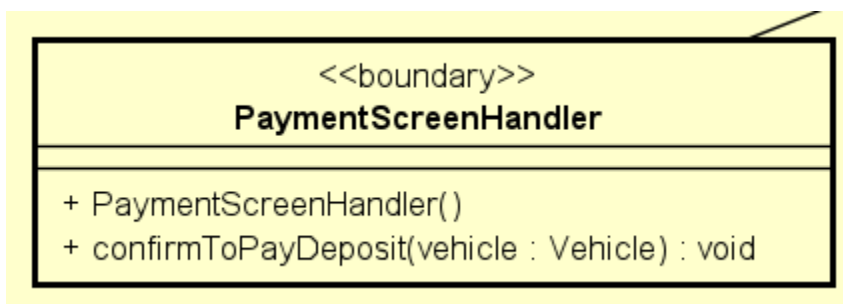
Method

Không

State

Không

4.3.3.18 Class “PaymentScreenHandler”



Attribute

Table 11. Example of operation design

#	Name	Return type	Description (purpose)
1	PaymentScreenHandler		Khởi tạo giao diện
2	confirmToPayDeposit	void	Xác nhận thanh toán

Parameter:

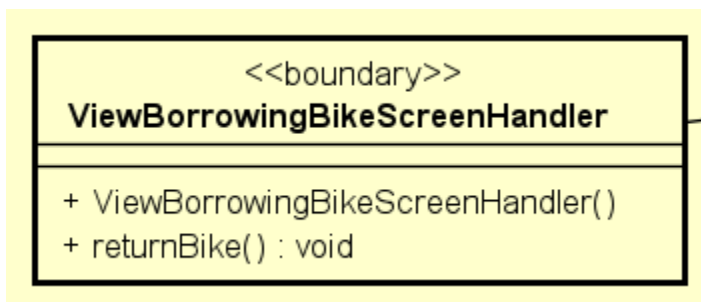
Method

Không

State

Không

4.3.3.19 Class “ViewBorrowingScreenHandler”



Attribute

Table 12. Example of operation design

#	Name	Return type	Description (purpose)
1	ViewBorrowingBicycleScreenHandler		Khởi tạo màn hình giao diện chứa thông tin xe đang thuê
2	returnBike	void	Hiển thị màn hình trả xe

Parameter:

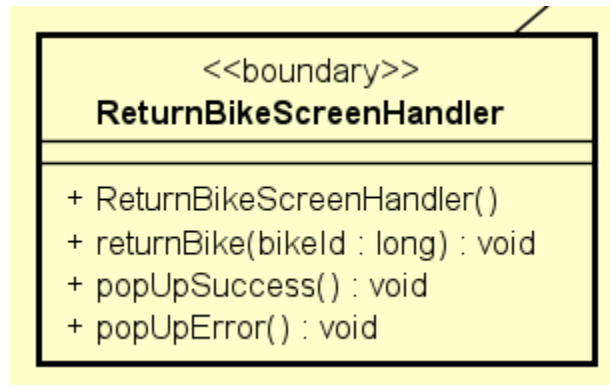
Method

Không

State

Không

4.3.3.20 Class “ReturnBikeScreenHandler”



Attribute

Table 13. Example of operation design

#	Name	Return type	Description (purpose)
1	ReturnBikeScreenHandler		Hiển thị màn hình trả xe
2	returnBike	void	Trả xe
3	popUpSuccess	void	Hiển thị màn hình thanh toán thành công
4	popUpError	Void	Hiển thị màn hình thanh toán thất bại

Parameter:

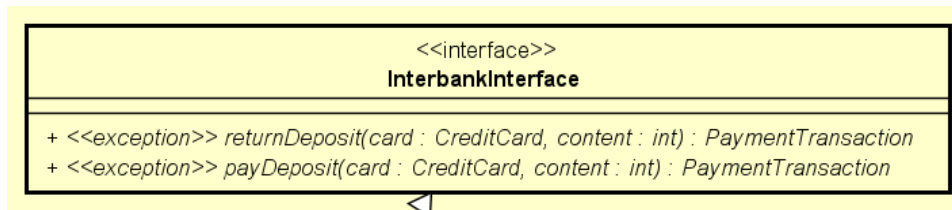
- BikeID : mã id xe

Method

Không

State

Không

4.3.3.21 Class “InterbankInterface”**Operation**

#	Name	Return type	Description (purpose)
1	payDeposit	PaymentTransaction	Trả phí đặt cọc và trả về giao dịch thanh toán
2	returnDeposit	PaymentTransaction	Hoàn lại tiền đặt cọc và trả về giao dịch thanh toán

Parameter:

- Card – thẻ tín dụng để giao dịch
- Amount – số tiền giao dịch
- Contents – nội dung giao dịch

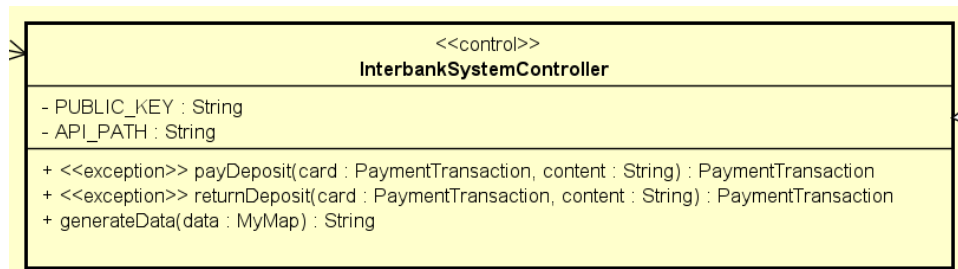
Method

Không

State

Không

4.3.3.22 Class “InterbankSubsystemController”



Attribute

Operation

#	Name	Return type	Description (purpose)
1	payDeposit	PaymentTransaction	Trả tiền đặt cọc và trả về giao dịch thanh toán
2	returnDeposit	PaymentTransaction	Hoàn lại tiền đặt cọc và trả về giao dịch thanh toán
3			

Parameter:

- 5 card: thông tin thẻ dùng để thanh toán
- 6 amount: tổng số tiền thanh toán
- 7 contents: nội dung giao dịch thanh toán

Exception:

- 8 InternalServerErrorException
- 9 InvalidCardException
- 10 NotEnoughBalanceException
- 11 InvalidDeliveryException
- 12 InvalidVersionException
- 13 InvalidTransactionAmountException

Method

- GenerateURL: chuyển dữ liệu từ dạng MyMap sang json

State

13.1.1.1 Class “InterbankSubsystem”

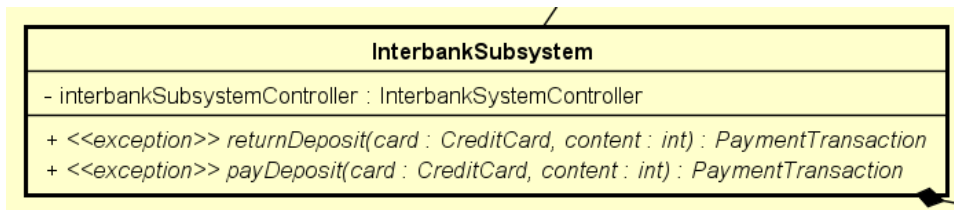


Table 14. Example of attribute design

#	Name	Data type	Default value	Description
1	Interbank...	InterbankSubsystemController	null	Interbank Subsystem Controller dùng để thực hiện các giao dịch
2				

Table 15. Example of operation design

#	Name	Return type	Description (purpose)
1	payDeposit	PaymentTransaction	Trả tiền đặt cọc
2	returnDeposit	PaymentTransaction	Hoàn lại tiền đặt cọc

Parameter:

- card: thông tin thẻ dùng để thanh toán
- amount: tổng số tiền thanh toán
- contents: nội dung giao dịch thanh toán

Exception:

- InternalServerErrorException
- InvalidCardException
- NotEnoughBalanceException
- InvalidDeliveryException
- InvalidVersionException
- InvalidTransactionAmountException
-

Method

Không

State

Không

14 Design Considerations

<Describe issues which need to be addressed or resolved before attempting to devise a complete design solution. Remember that, you have to refactor your source code to strictly follow the final design>

14.1 Goals and Guidelines

Mục tiêu: Thiết kế hệ thống có thể dễ dàng nâng cấp, bảo trì. Do các hệ thống phần mềm thường xuyên thay đổi yêu cầu, hệ thống sẽ phải thêm các tính năng mới phù hợp với yêu cầu người dùng. Một bản thiết kế phần mềm tốt sẽ giúp việc thêm các tính năng này dễ dàng hơn.

Để thiết kế hệ thống có thể dễ dàng bảo trì, mở rộng nhóm tập trung áp dụng các nguyên tắc thiết kế đã được học trong bài giảng. Do lượng kiến thức còn hạn chế nên bản thiết kế vẫn chưa hoàn hảo.

14.2 Architectural Strategies

Ngôn ngữ sử dụng: Java – do java là một ngôn ngữ hướng đối tượng, dễ dàng mở rộng, ngôn ngữ này giúp lập trình giảm thiểu các vi phạm coupling.

Vận dụng các kiến thức đã học, tăng tính cohesion trong một module, giảm sự liên kết giữa các module với nhau từ đó sẽ tạo ra ít sự thay đổi hơn. Nếu một module thay đổi thì các module khác do ít phụ thuộc với nhau nên ít cần phải thay đổi từ đó mở rộng thêm các chức năng khác hơn.

Việc vận dụng nguyên lý thiết kế hướng đối tượng SOLID giúp code dễ đọc hơn, dễ hiểu và dễ dàng bảo trì.

14.3 Cohesion

5.3.1. Coupling

5.3.1.1. Content coupling

Related modules	Description	Improvement
Không có		

5.3.1.2. Common coupling

Related modules	Description	Improvement
-----------------	-------------	-------------

Không có		
----------	--	--

5.3.1.3. Control coupling

Related modules	Description	Improvement
ReturnBikeController	Phương thức tính toán phí thuê xe đang truyền vào classifyId thông qua câu lệnh if else xác định cách tính tiền	Chuyển cách tính tiền sang 1 class khác, sử dụng factory pattern

5.3.1.4. Stamp coupling

Related modules	Description	Improvement
Không có		

5.3.1.5. Data coupling

Related modules	Description	Improvement
Tất cả các module còn lại	Sự giao tiếp giữa các controller với entity hay giữa handler với controller đều giao tiếp thông qua các method, không trở trực tiếp vào các attributes của object. Các method chỉ truyền vào đủ các param dùng tới, không truyền thừa nên các module này có thể coi là data coupling	Không

5.3.1.6. Uncoupling

Related modules	Description	Improvement
Không có		

5.3.2. Cohesion

5.3.2.1. Coincidental cohesion

Related modules	Description	Improvement
Không có		

5.3.2.2. Logical cohesion

Related modules	Description	Improvement
Không có		

5.3.2.3. Temporal cohesion

Related modules	Description	Improvement
Không có		

5.3.2.4. Procedural cohesion

Related modules	Description	Improvement
Không có		

5.3.2.5. Communicational cohesion

Related modules	Description	Improvement
InterbankSubsytemController	Các thành phần trong class này để xử lý và trả về cùng 1 kiểu dữ liệu PaymentTransaction	Không có
InterbankSubsystem	Cũng nhận dữ liệu đầu vào giống nhau và trả về kiểu PaymentTransaction	Không

5.3.2.6. Sequential cohesion

Related modules	Description	Improvement
PaymentController	Output của hàm getExpiredDate là đầu vào của payOrder	Không có

5.3.2.7. Informatin cohesion

Related modules	Description	Improvement
Không có		

5.3.2.8. Function cohesion

Related modules	Description	Improvement
Không có		

14.4 Design Principles

5.4.1. Single Responsibility Principle

#	Related modules	Description	Improvement
1	ReturnBikeController	Ngoài việc điều hướng thì controller này còn phải thực hiện tính toán phí thuê xe	Tách phương thức tính toán phí thuê xe ra class khác
2	PaymentController	Ngoài việc điều hướng thanh toán thì controller này còn phải thực hiện convert dữ liệu về dạng chuẩn mà API mong muốn	Tách việc xử lý, convert dữ liệu sang 1 class riêng

5.4.2. Open/Closed Principle

#	Related modules	Description	Improvement
1	ReturnBikeController	Phương thức tính toán phí thuê xe calculateTotalAmount sẽ phải thay đổi khi ta muốn thêm cách thức tính tiền hoặc thêm loại xe mới	Tạo 1 interface CalculateAmount, khi muốn mở rộng thay đổi thì tạo class implement interface này, hoặc sửa class cần thay đổi

5.4.3. Liskov Subsitution Principle

#	Related modules	Description	Improvement
1	Không có vi phạm		

5.4.4. Interface Segregation Principle

#	Related modules	Description	Improvement
1	InterbankInterface	Khi có thêm hệ thống ngân hàng khác có thêm hay không dùng thức 2 phương thức payOrder hay refund lúc này interface đã vi phạm tính này	Tạo 1 interface cha để giao tiếp với các ngân hàng, tạo các interface con chứa các phương thức của ngân hàng đó. Các class sẽ implement các interface các chức năng mà ngân hàng đó cung cấp.

5.4.5. Dependency Inversion Principle

#	Related modules	Description	Improvement
1	PaymentTransaction	PaymentTransaction đang phụ thuộc chặt chẽ vào CreditCard, khi mở rộng hệ thống sẽ có thêm thanh toán bằng hình thức khác thì sẽ không thể sử dụng CreditCard	Tạo 1 interface PaymentCard để khi mở rộng thanh toán ta chỉ việc implemt lại interface này

14.5 Design Patterns

Trong bản thiết kế này, chúng em đã sử dụng tới Factory Pattern để xác định tính toán chi phí thuê xe. Để tránh việc vi phạm control coupling, em sử dụng Factory Pattern để điều khiển luồng dữ liệu. Tạo 1 interface CalculateAmount, các cách tính phí sẽ implement interface này. Tạo 1 class CalculateFactory, tạo 1 phương thức có input truyền vào loại xe, output sẽ là 1 instance tương ứng với loại xe truyền vào. Từ đối tượng trả về chỉ cần gọi tới hàm tính toán phí xe sẽ được kết quả mong muốn.