HANOI UNIVERSITY OF SCIENCE AND TECHNOLOGY

School of Information and communications technology

Software Design Document

Version 1.0

Eco Bike Rental Software

Subject: ITSS Software Development

Group 18

Nguyễn Mạnh Cường – 20183874

Hoàng Hải Đăng – 20183877

Nguyễn Văn Chiến – 20183870

Nguyễn Đào Duy Kiên - 20183935

Hanoi, January, 2022

Table of Contents

1 Introduction 3

1.1 Objective 3

1.2 Scope 3

1.3 Glossary 3

1.4 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

2.2.3 Risks 4

3 System Architecture and Architecture Design 5

3.1 Architectural Patterns 5

3.2 Interaction Diagrams 5

3.2.1 View Dock and Bike Info 5

3.2.2 Rent Bike 7

3.2.3 Return Bike 10

3.3 Analysis Class Diagrams 12

3.3.1 Docks and Bikes 12

3.3.2 Rent Bike 12

3.3.3 Return Bike 13

3.4 Unified Analysis Class Diagram 13

3.5 Security Software Architecture 13

4 Detailed Design 14

4.1 User Interface Design 14

4.1.1 Screen Configuration Standardization 14

4.1.2 Screen Transition Diagrams 15
4.1.3 Screen Specifications 15
4.2 Data Modeling 27
4.2.1 Conceptual Data Modeling 27
4.2.2 Database Design 28
4.3 Non-Database Management System Files 39
4.4 Class Design 40
4.4.1 General Class Diagram 40
4.4.2 Class Diagrams 40
4.4.3 Class Design 42
5 Design Considerations 108
5.1 Goals and Guidelines 108
5.2 Architectural Strategies 108
5.3 Coupling and Cohesion 108
5.4 Design Principles 108

1 Introduction

1.1 Objective

Mục đích của tài liệu này là trình bày mô tả chi tiết về các thiết kế của Eco Bike Rental. Tài liệu này dành cho nhóm 18, sử dụng các thiết kế làm hướng dẫn để thực hiện dự án. Mặt khác, tài liệu này được TS.Nguyễn Thị Thu Trang và trợ giảng xem xét như một phần của khóa học.

1.2 Scope

Mục đích của phần mềm Eco Bike Rental là cung cấp dịch vụ cho thuê xe đạp cho khách hàng.

Trong phần mềm này, người dùng sẽ được cung cấp khả năng tìm kiếm các bãi xe và xem thông tin chi tiết của nó. Tại mỗi bãi xe, người dùng có thể xem thông tin về những chiếc xe đạp đang đỗ tại đó. Cho thuê và trả xe đạp là những chức năng chính của hệ thống này.

Liên ngân hàng sẽ có khả năng thực hiện các giao dịch thanh toán. Liên ngân hàng sẽ diễn ra quá trình xác nhận, thêm tiền và khấu trừ tiền.

Tài liệu này mô tả chi tiết về kiến trúc phần mềm của hệ thống kiểm kê. Nó chỉ định cấu trúc và thiết kế của một số mô-đun được thảo luận trong SRS. Nó cũng cung cấp các sơ đồ tuần tự và hoạt động cho một số trường hợp sử dụng. Biểu đồ lớp cho thấy cách nhóm lập trình sẽ triển khai mô-đun cụ thể.

1.3 Glossary

1.4 References

2 Overall Description

2.1 General Overview

Hệ thống EBR là một mô phỏng cho dịch vụ cho thuê xe đạp. Đây là một phần mềm máy tính để bàn chạy trên Java Runtime Environment. Phần mềm này được thiết kế bằng cách sử dụng mô hình thiết kế kiến trúc MVC.

2.2 Assumptions/Constraints/Risks

2.2.1 Assumptions

Thiết kế được mô tả trong tài liệu này dựa trên các yêu cầu mô phỏng thay vì các yêu cầu thực tế.

Phần mềm này được thiết kế để chạy trên bất kỳ hệ điều hành nào có JRE.

2.2.2 Constraints

Hệ thống được xây dựng chỉ có thể truy cập thông qua máy tính cá nhân. Hệ thống được triển khai bằng Java và sử dụng JavaFX để tạo giao diện người dùng.

Để lưu trữ dữ liệu, sử dụng PostgreSQL.

2.2.3 Risks

3 System Architecture and Architecture Design

3.1 Architectural Patterns

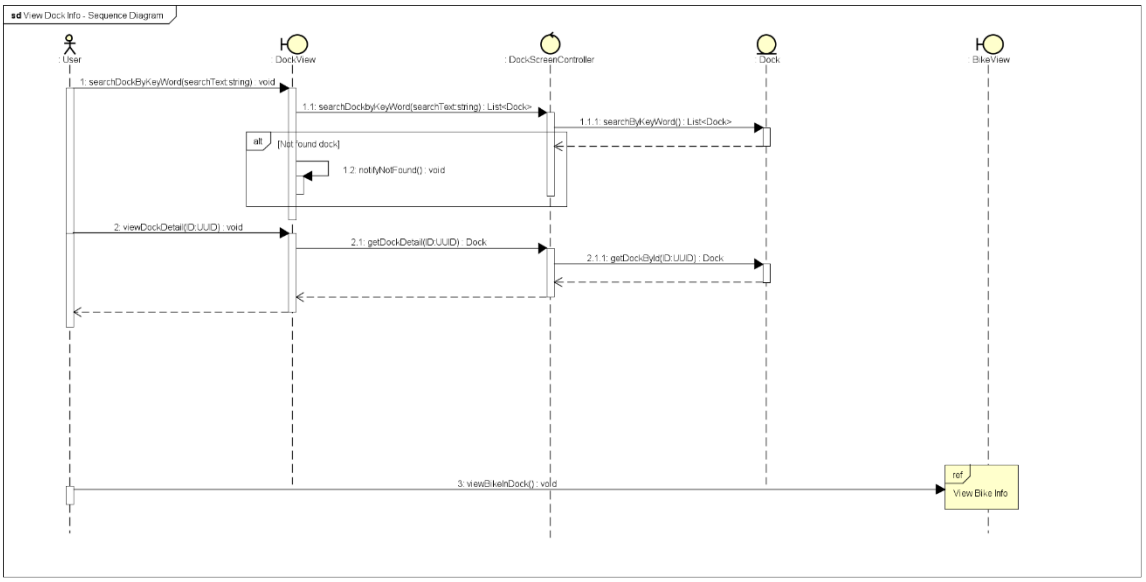
Model-View-Controller (MVC) là mẫu thiết kế được chọn.

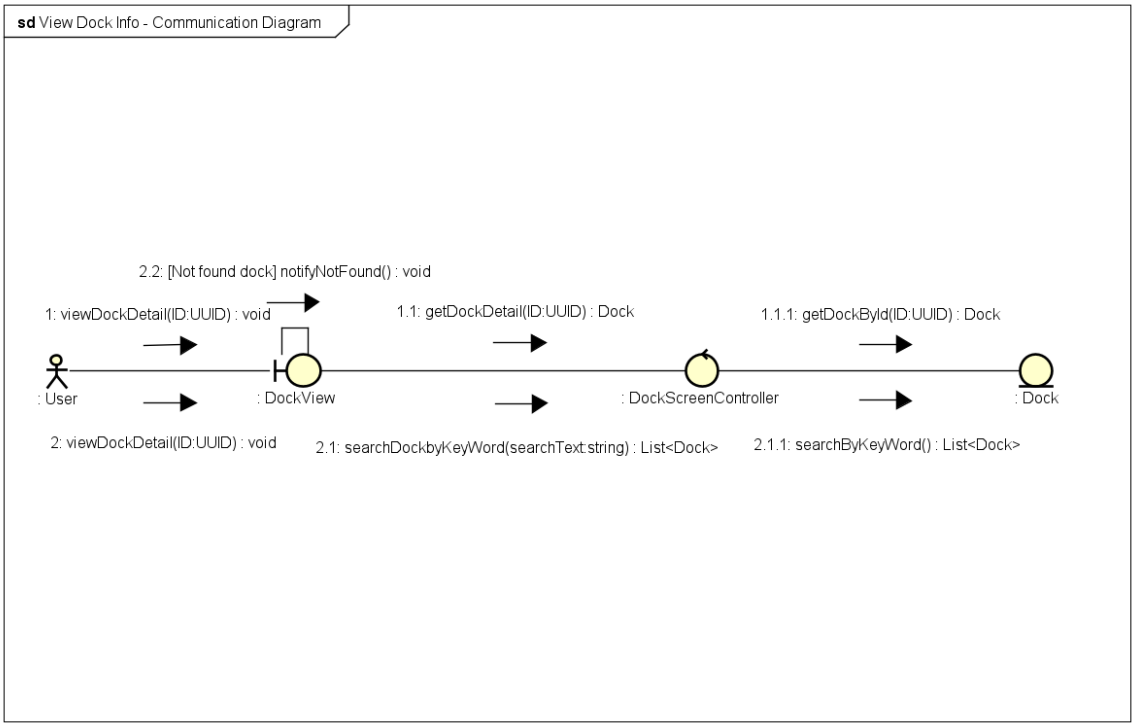
Các lợi ích của MVC:

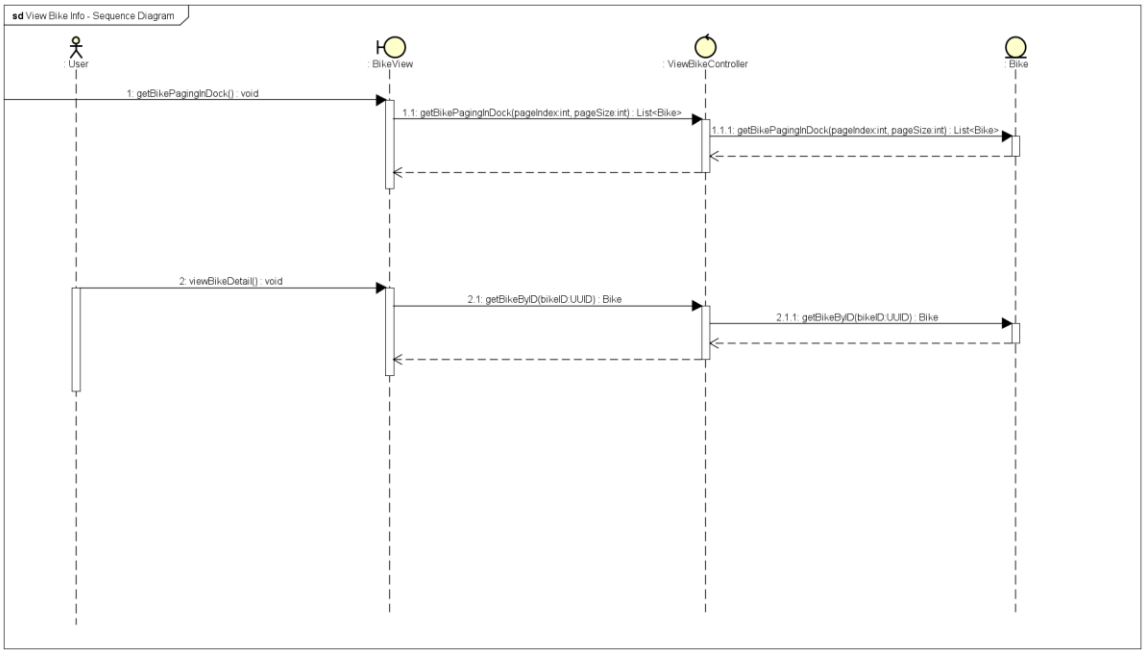
- Phát triển đồng thời
- High cohesion
- Low coupling
- Dễ dàng sửa đổi, phát triển trong tương lai

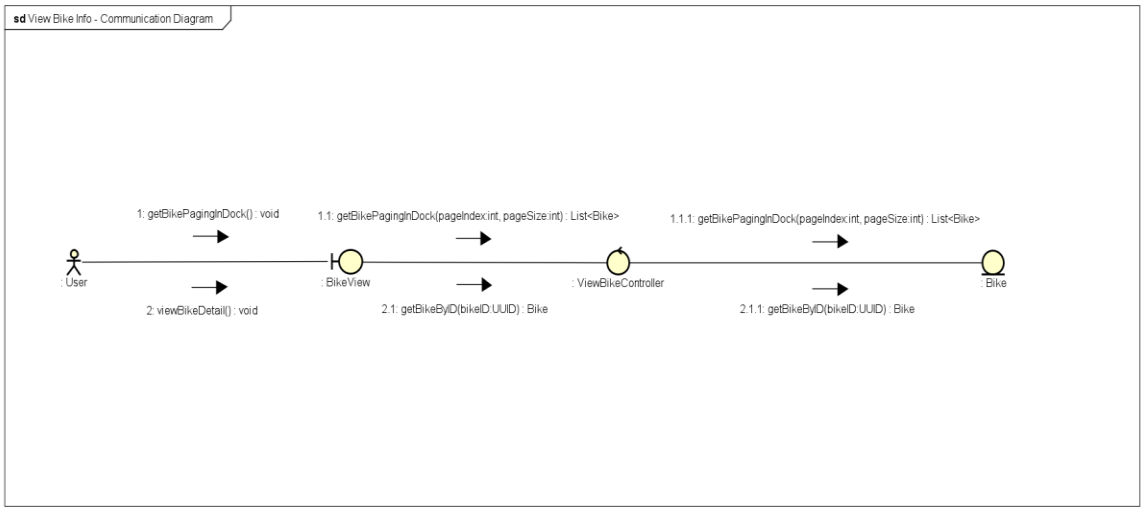
3.2 Interaction Diagrams

3.2.1 View Dock and Bike Info

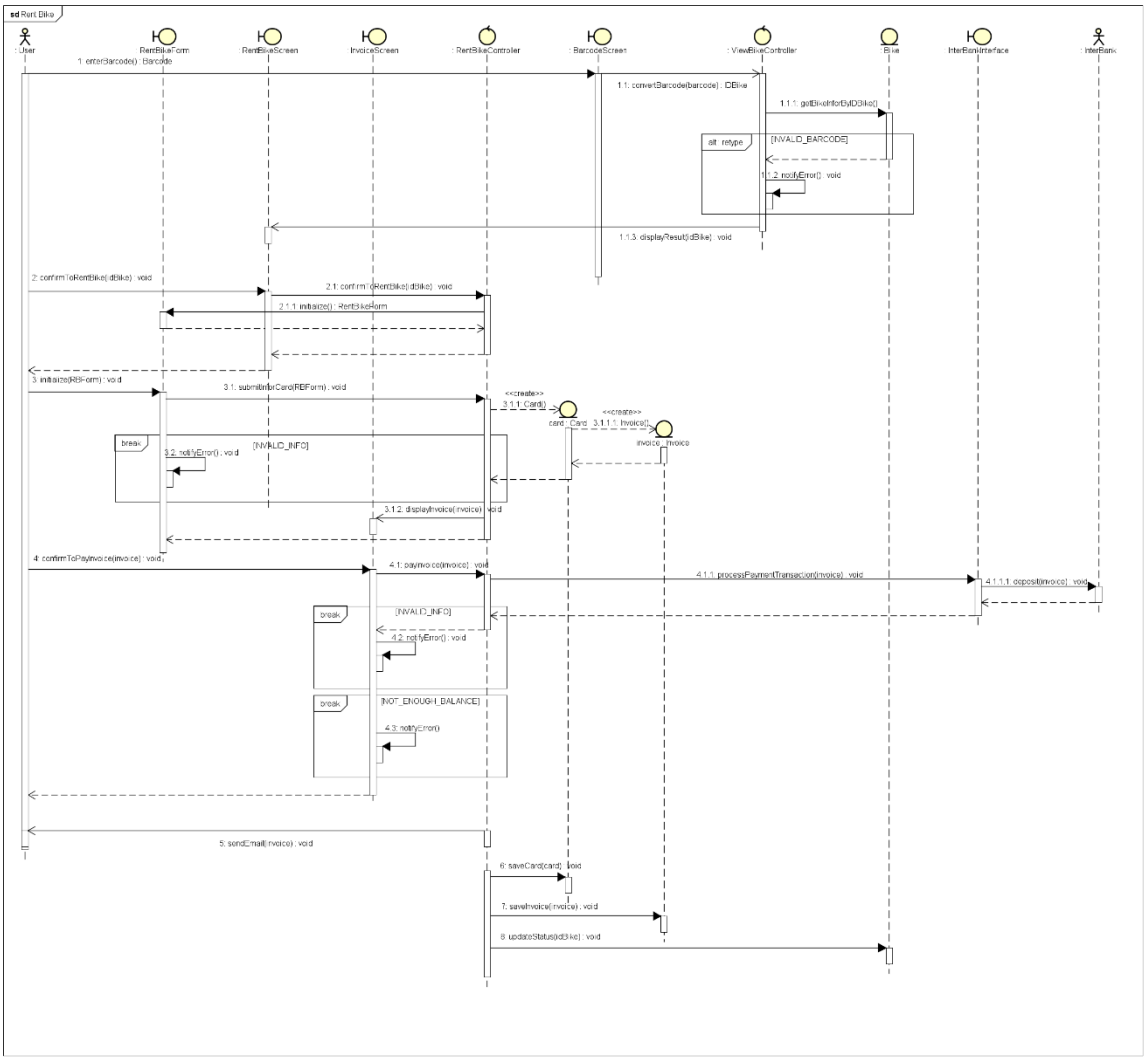


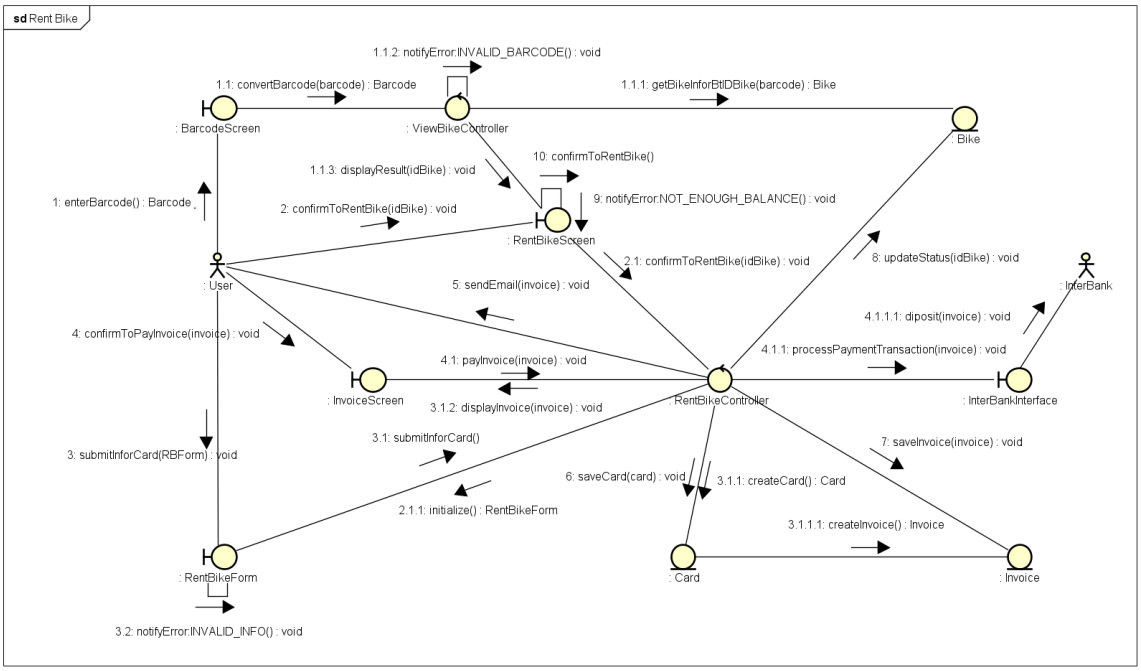

sd View Dock Info - Communication Diagram
2.2: [Not found dock] notifyNotFound() : void
1: viewDockDetail(ID:UUID) : void
1.1: getDockDetail(ID:UUID) : Dock
1.1.1: getDockById(ID:UUID) : Dock
: User
: DockView
: DockScreenController
: Dock
2: viewDockDetail(ID:UUID) : void
2.1: searchDockbyKeyWord(searchText:string) : List<Dock>
2.1.1: searchByKeyWord() : List<Dock>


sd View Bike Info - Sequence Diagram
: User
: BikeView
: ViewBikeController
: Bike
1: getBikePagingInDock() : void
1.1: getBikePagingInDock(pageIndex:int, pageSize:int) : List<Bike>
1.1.1: getBikePagingInDock(pageIndex:int, pageSize:int) : List<Bike>
2: viewBikeDetail() : void
2.1: getBikeByID(bikeID:UUID) : Bike
2.1.1: getBikeByID(bikeID:UUID) : Bike

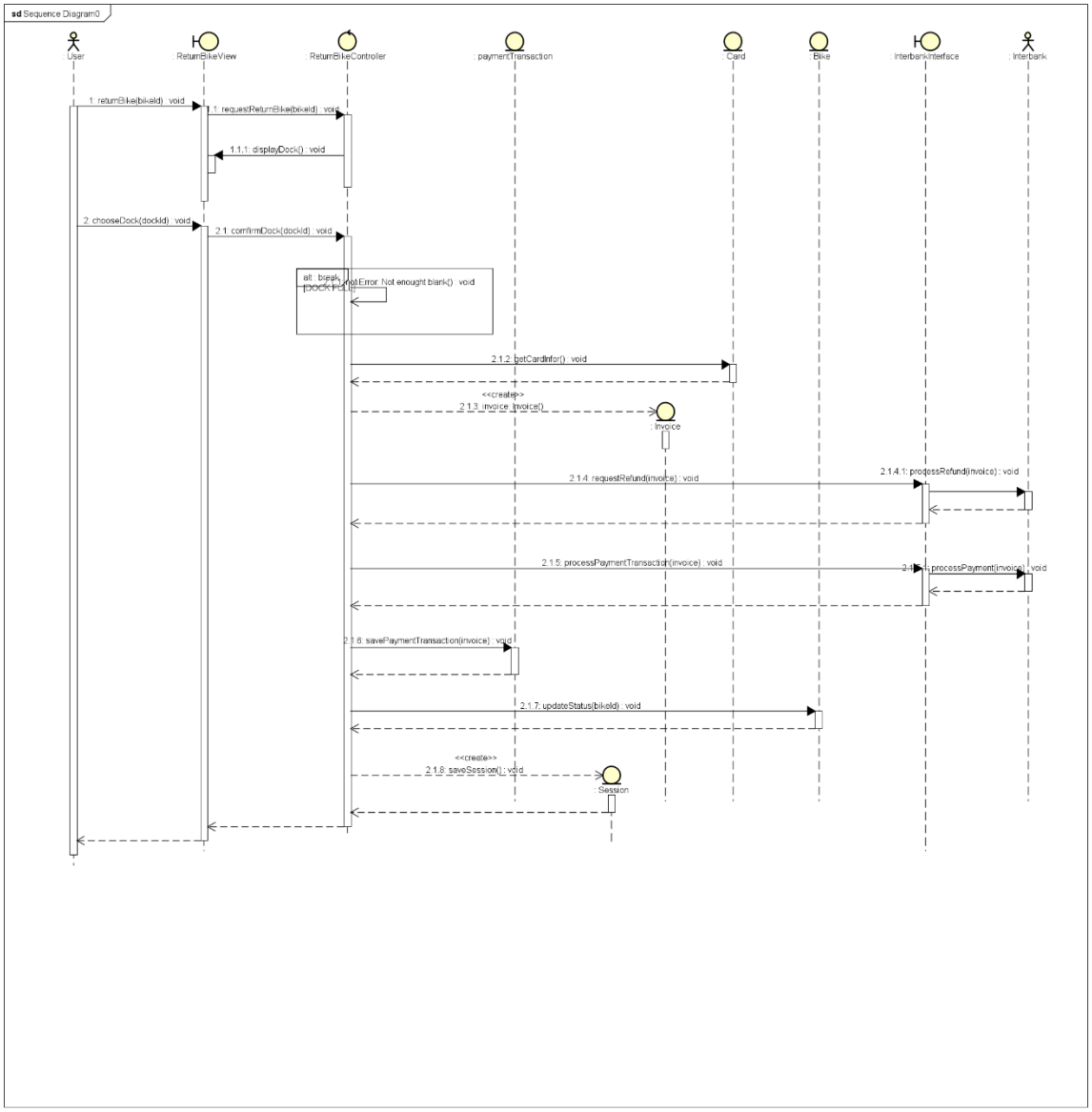


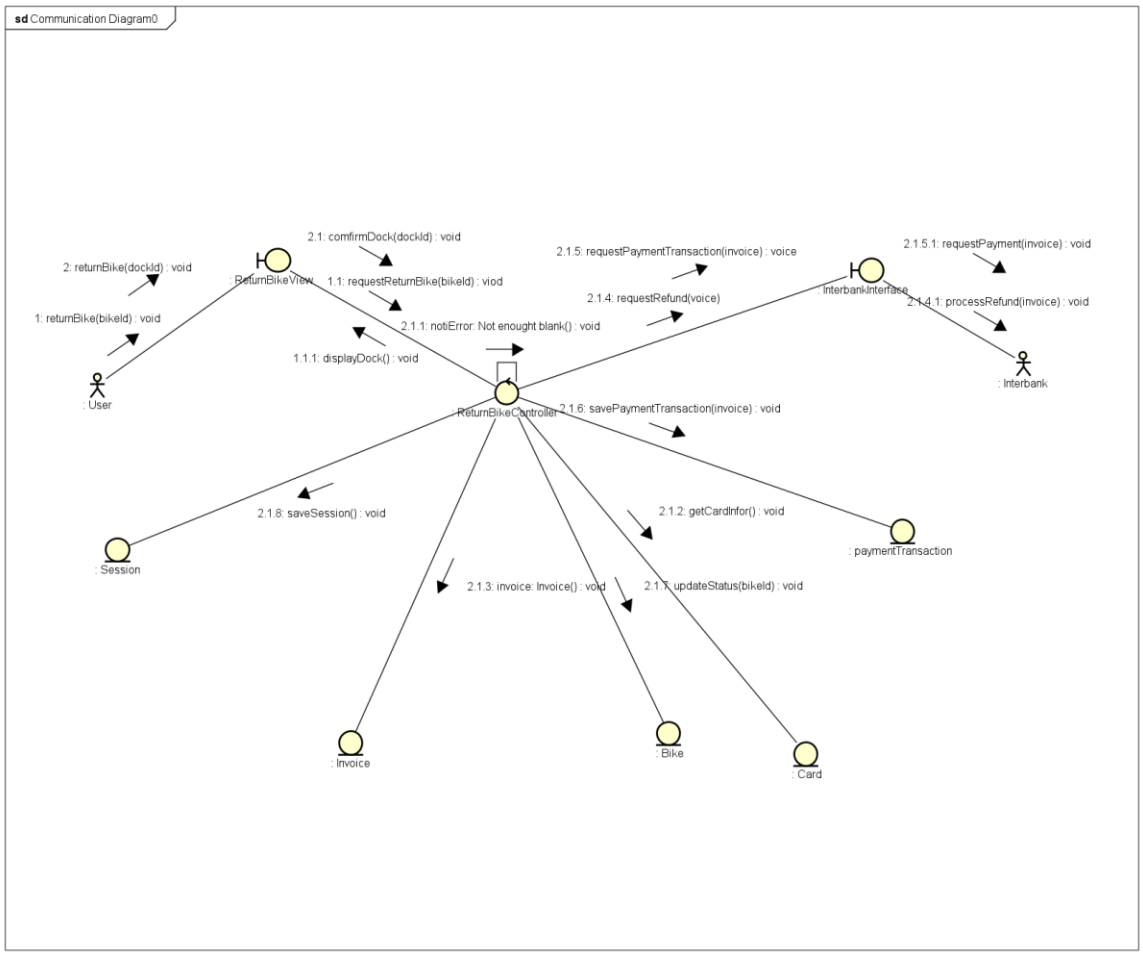
3.2.2 Rent Bike


sd Rent Bike
: User
: RentBikeForm
: RentBikeScreen
: InvoiceScreen
: RentBikeController
: BarcodeScreen
: ViewBikeController
: Bike
: InterBankInterface
: InterBank
1: enterBarcode() : Barcode
1.1: convertBarcode(barcode) : IDBike
1.1.1: getBikeInforByIDBike()
alt retype
[INVALID_BARCODE]
1.1.2 notifyError() : void
1.1.3: displayResult(dBike) :void
2: confirmToRentBike(dBike) : void
2.1: confirm(dRentBike):dBike) :void
2.1.1: initialize() : RentBikeForm
3: initialize(RBForm) : void
3.1: submitInforCard(RBForm) : void
<<create>>
3.1.1 Card()
card : Card
3.1.1.1: Invoice()
<<create>>
invoice : Invoice
break
[INVALID_INFO]
3.2: notifyError() : void
3.1.2: displayInvoice(invoice) :void
4: confirmToPayInvoice(invoice) : void
4.1: payInvoice(invoice) : void
4.1.1: processPaymentTransaction(invoice) : void
4.1.1.1: deposit(invoice) : void
break
[INVALID_INFO]
4.2: notifyError() : void
break
[NOT_ENOUGH_BALANCE]
4.3: notifyError()
5: sendEmail(invoice) : void
6: saveCard(card) :void
7: saveInvoice(invoice) : void
8: updateStatus(dBike) : void


sd Rent Bike
1.1.2: notifyError:INVALID_BARCODE() : void
1.1: convertBarcode(barcode) : Barcode
1.1.1: getBikeInforBtIDBike(barcode) : Bike
: BarcodeScreen
: ViewBikeController
: Bike
10: confirmToRentBike()
1.1.3: displayResult(idBike) : void
1: enterBarcode() : Barcode
2: confirmToRentBike(idBike) : void
9: notifyError:NOT_ENOUGH_BALANCE() : void
: RentBikeScreen
: User
2.1: confirmToRentBike(idBike) : void
8: updateStatus(idBike) : void
: InterBank
5: sendEmail(invoice) : void
4.1.1.1: diposit(invoice) : void
4: confirmToPayInvoice(invoice) : void
4.1.1: processPaymentTransaction(invoice) : void
4.1: payInvoice(invoice) : void
: InvoiceScreen
3.1.2: displayInvoice(invoice) : void
RentBikeController
: InterBankInterface
3.1: submitInforCard()
7: saveInvoice(invoice) : void
3: submitInforCard(RBForm) : void
6: saveCard(card) : void
3.1.1: createCard() : Card
2.1.1: initialize() : RentBikeForm
3.1.1.1: createInvoice() : Invoice
: RentBikeForm
: Card
: Invoice
3.2: notifyError:INVALID_INFO() : void

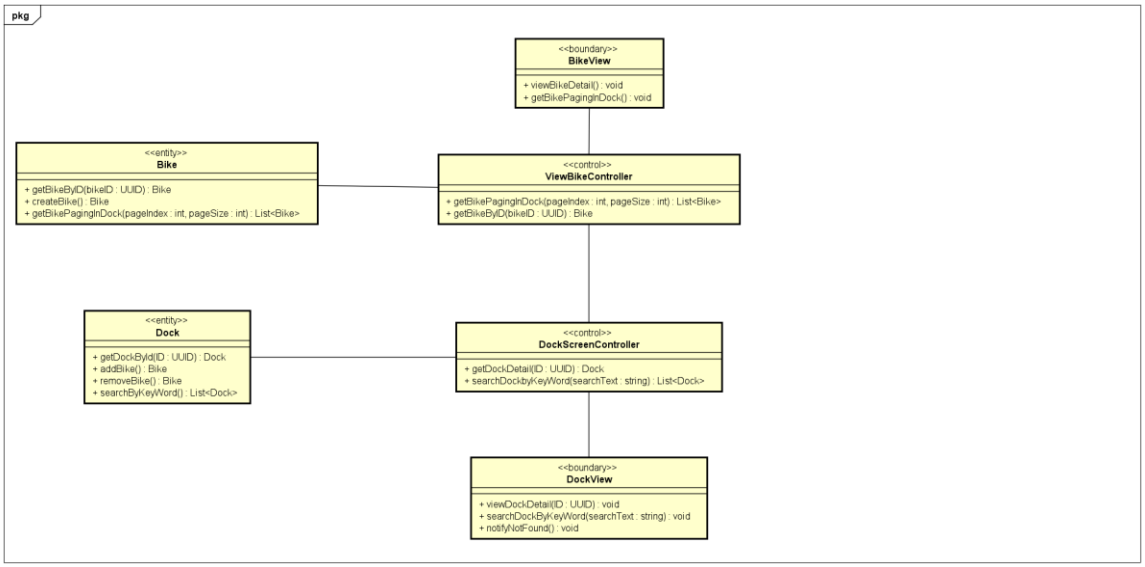
3.2.3 Return Bike



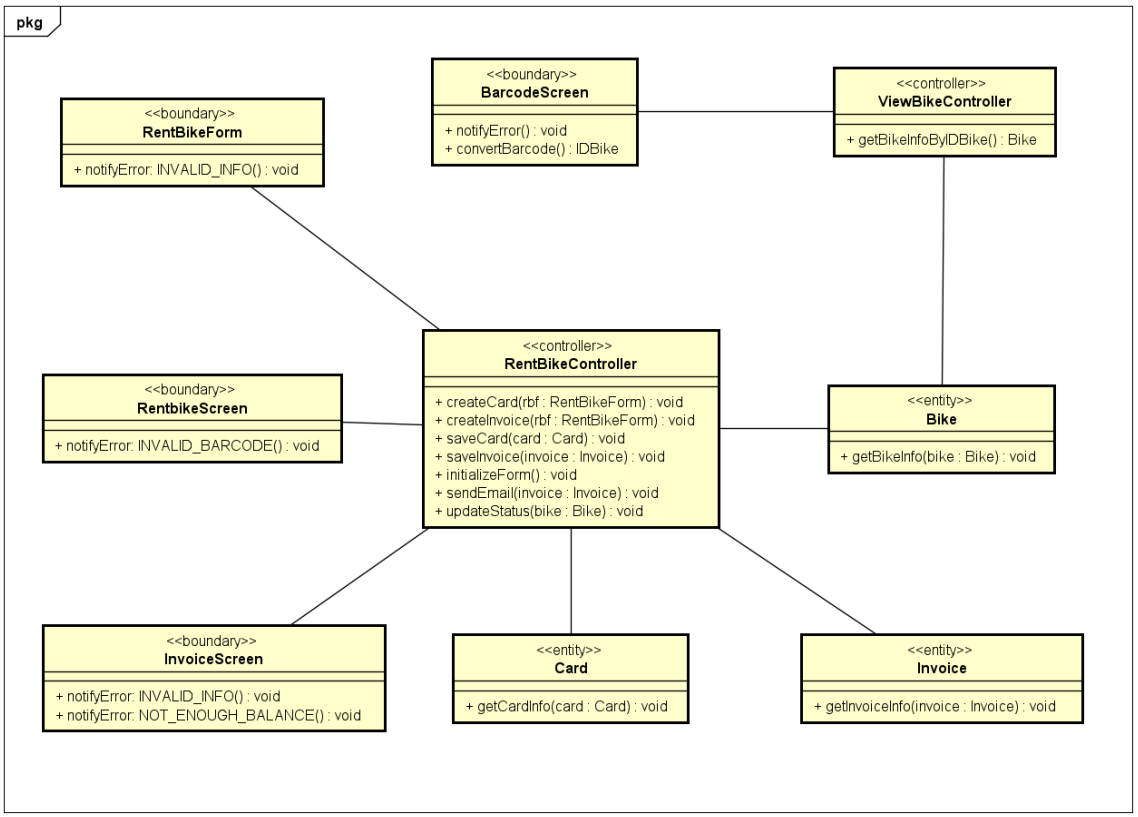

sd Communication Diagram0
2.1: comfirmDock(dockId) : void
2: returnBike(dockId) : void
: ReturnBikeView
1.1: requestReturnBike(bikeId) : viod
2.1.5: requestPaymentTransaction(invoice) : voice
2.1.5.1: requestPayment(invoice) : void
: InterbankInterface
2.1.4: requestRefund(voice)
2.1.4.1: processRefund(invoice) : void
1: returnBike(bikeId) : void
2.1.1: notiError: Not enought blank() : void
1.1.1: displayDock() : void
: Interbank
: User
: ReturnBikeController
2.1.6: savePaymentTransaction(invoice) : void
2.1.8: saveSession() : void
2.1.2: getCardInfor() : void
: paymentTransaction
: Session
2.1.3: invoice: Invoice() : void
2.1.7: updateStatus(bikeId) : void
: Invoice
: Bike
: Card

3.3 Analysis Class Diagrams

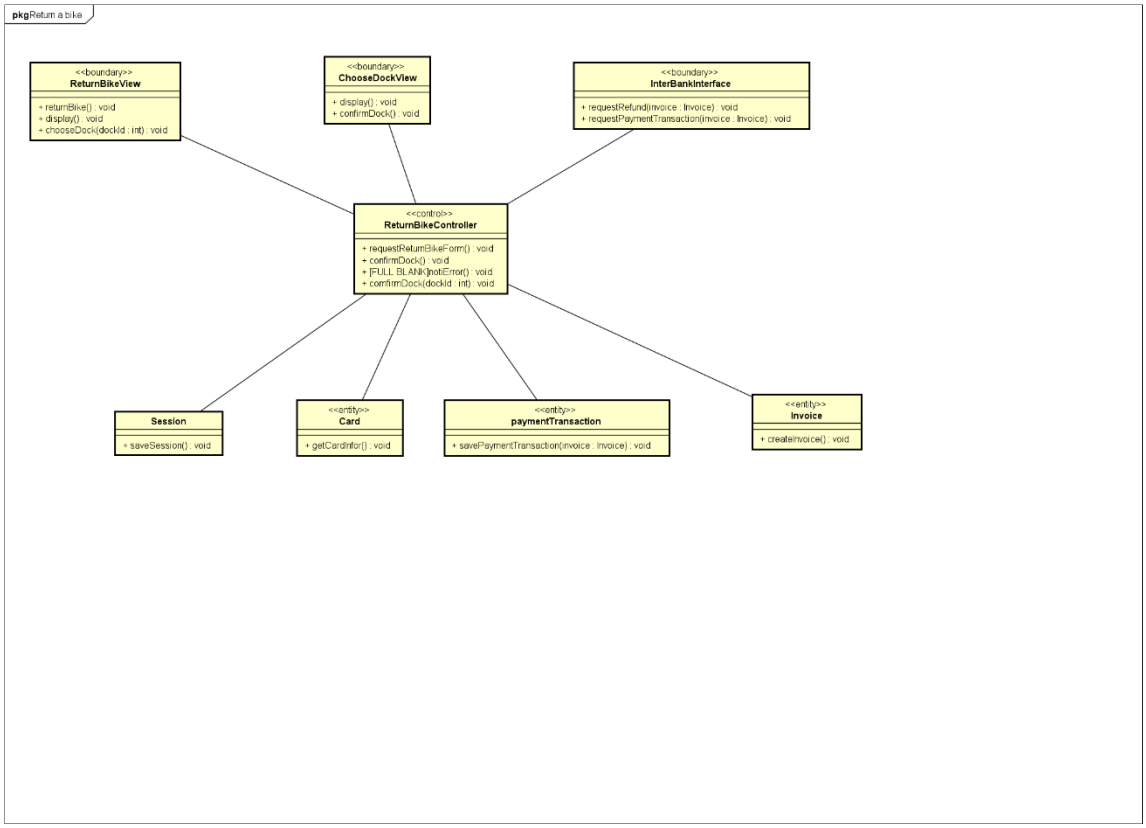
3.3.1 Docks and Bikes



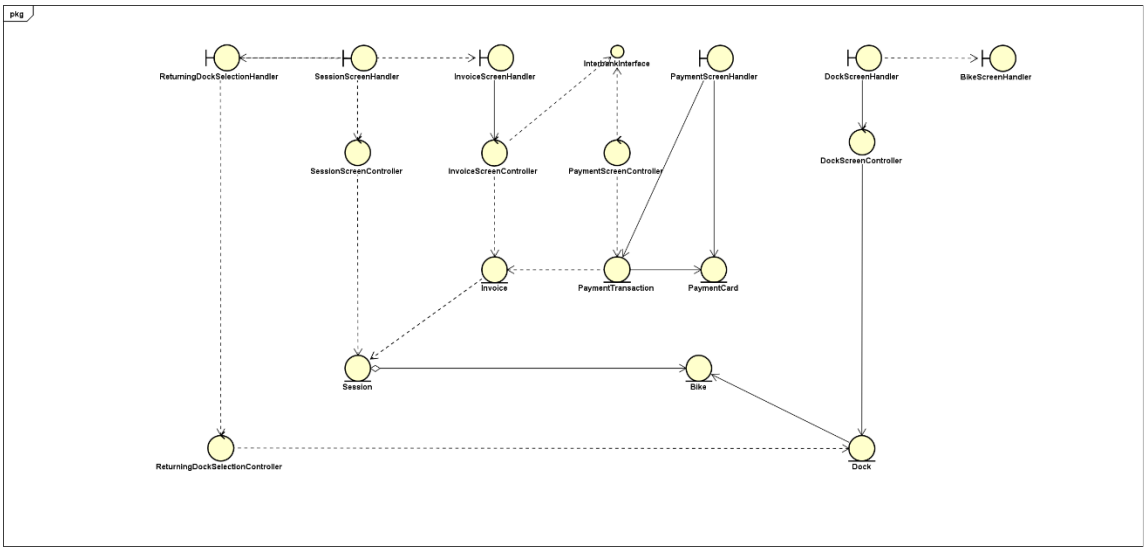
3.3.2 Rent Bike



3.3.3 Return Bike



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

Display

Số lượng màu được hỗ trợ: 16,777,216 màu

Độ phân giải: 1366 × 768 pixels

Screen

Vị trí của của button: Ở dưới cùng (theo chiều dọc) và ở giữa (theo chiều ngang) của khung.

Vị trí thông điệp: Ở giữa trung tâm màn hình

Vị trí Tiêu đề màn hình: Đặt góc bên trái màn hình

Sự nhất quán trong hiển thị chữ số: dấu phẩy để phân cách hàng nghìn và chuỗi chỉ bao gồm các ký tự, chữ số, dấu phẩy, dấu chấm, dấu cách, dấu gạch dưới và ký hiệu gạch nối

Control

Kích thước chữ:

- Title and header text: font: System, style: bold, size: 48px, color: #36C0AA
  (medium-dark purple)
- Text in button: font: System, style: black, size: 24px, color: #FFFFFF
- Other text: font: System, style: regular, size: 14px, color: #000000

Xử lí kiểm tra đầu vào: Nên kiểm tra xem input có empty hay không. Tiếp theo, kiểm tra xem input có đúng format hay không.

Dịch chuyển màn hình: Không có các khung chồng lên nhau. Các màn hình được tách biệt. Tuy nhiên, hướng dẫn sử dụng được xem như là 1 popup message vì màn hình chính ở dưới sẽ không thể thao tác trong khi màn hình hướng dẫn sử dụng đang được hiển thị. Ban đầu khi app khởi chạy thì màn hình splash screen (màn hình chớp) sẽ được hiện lên và sau đó màn hình đầu tiên(Home Screen) sẽ xuất hiện.

Thứ tự các màn hình trong hệ thống:

1. Splash screen (first screen)
2. Home screen – Hiện danh sách bãi xe

3. Barcode Popup – Nhập Barcode
4. Dock View Screen – Xem thông tin bãi xe vào danh sách xe trong bãi
5. Bike View Screen – Xem thông tin chi tiết xe
6. Payment Screen – Điền thông tin thẻ
7. Transaction Result Popup – Hiện kết quả giao dịch
8. Renting Session – Thông tin xe đang thuê
9. Invoice screen – Xem chi tiết hóa đơn
10. Dock Selection Screen – Hiển thị để chọn bãi xe cho việc trả xe

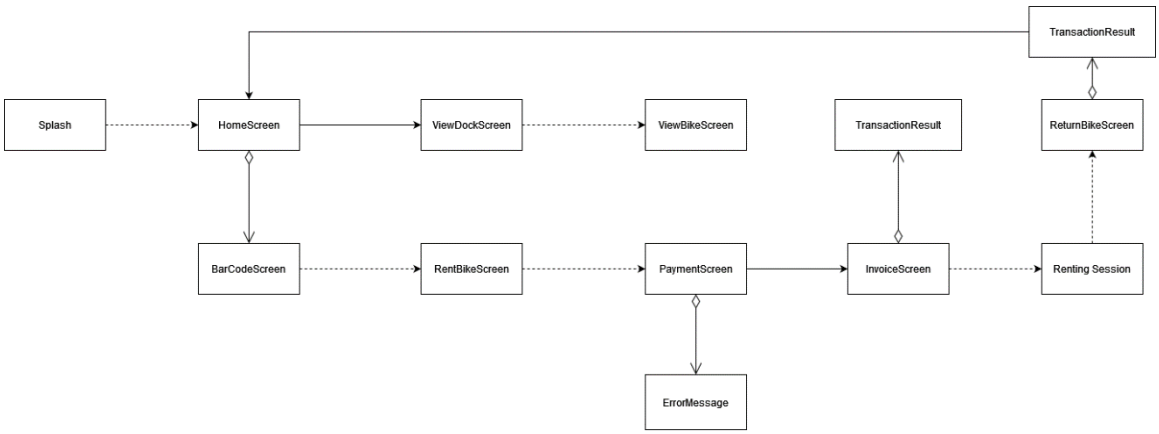
Nhập đầu vào từ bàn phím

Sẽ không có phím tắt. Có các button quay lại để quay lại các màn hình trước đó. Ngoài ra button "X" nằm ở thanh tiêu đề bên phải để đóng screen

Error

Một thông điệp sẽ được hiện lên để thông báo cho người dùng biết vấn đề đang gặp phải là gì.

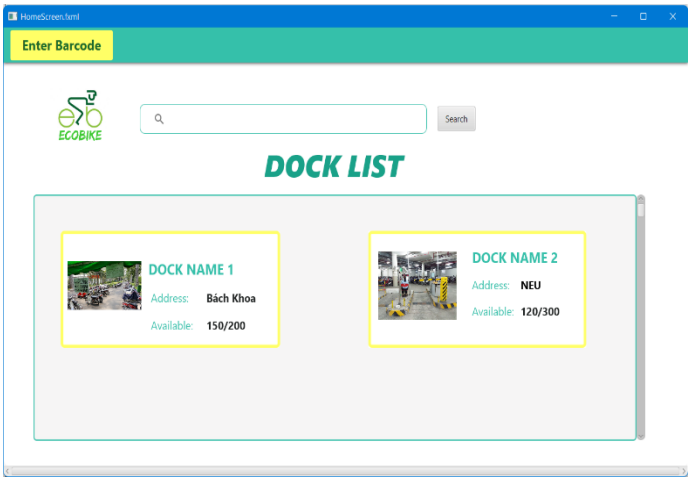
4.1.2 Screen Transition Diagrams



4.1.3 Screen Specifications

4.1.3.1 Home Screen

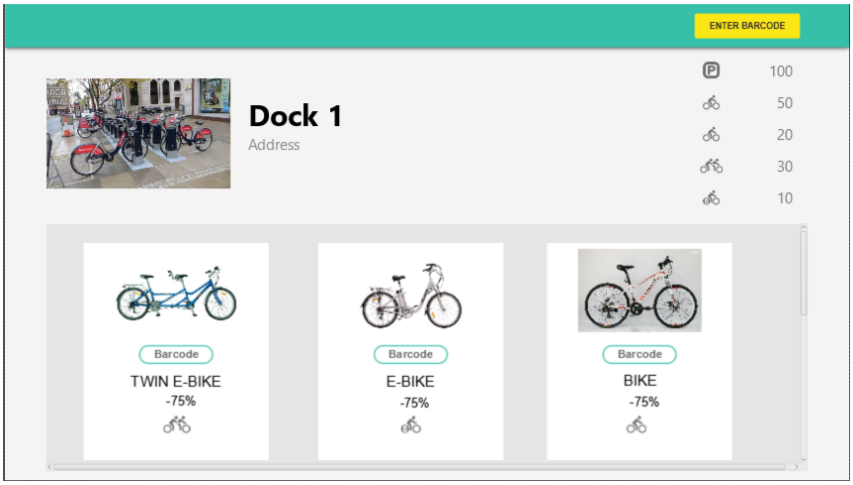
EBR Software		Date of creation	Approved by	Reviewed by	Person in charge
Screen specification	Home Screen	17/11/2021			Nguyễn Mạnh Cường



Control	Operation	Function
"Enter Barcode" Button	Click	Display Barcode Popup Screen
Area for entering search	Initial	Receive Text for Searching
"Search" Button	Click	Searching Dock

Screen name	Home Screen			
Item name	Number of digits (bytes)	Type	Field attribute	Remarks
Search field	50	String	Black	Left justified
Dock's name	50	String	Green	Left justified
Dock's address	50	String	Black	Left justified
Available	10	String	Black	Left justified

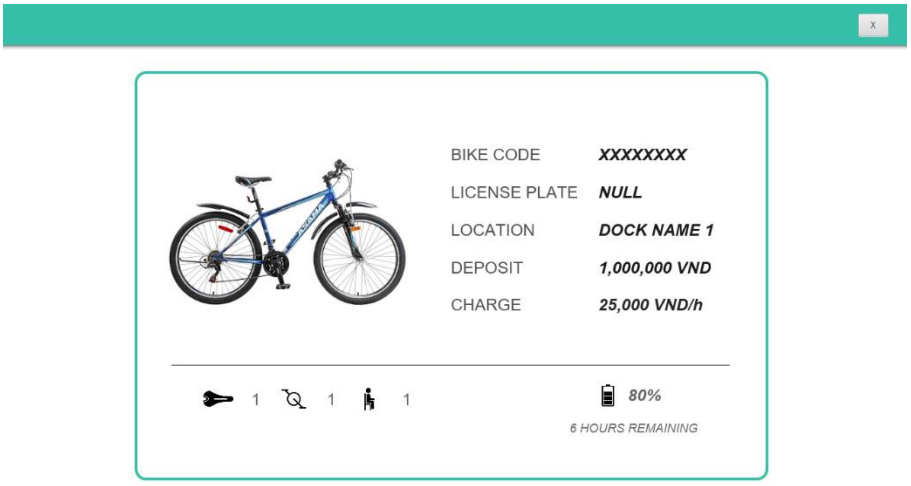
4.1.3.2 Dock View Screen

<table>
<tr><td colspan="2">EBR Software</td><td>Date of creation</td><td>Approved by</td><td>Reviewed by</td><td>Person in charge</td></tr>
<tr><td>Screen specification</td><td>Dock View Screen</td><td>18/11/2021</td><td></td><td></td><td>Hoàng Hải Đăng</td></tr>
<tr><td colspan="2" rowspan="8"> 
</td><td>Control</td><td>Operation</td><td colspan="2">Function</td></tr>
<tr><td>Enter Barcode Button</td><td>Click</td><td colspan="2">Display Barcode Popup Screen</td></tr>
<tr><td>Back Button</td><td>Click</td><td colspan="2">Back to Home Screen</td></tr>
<tr><td>Rent Bike Button</td><td>Click</td><td colspan="2">Display Bike View Screen</td></tr>
<tr><td>EBR Home Screen Button</td><td>Click</td><td colspan="2">Display Home Screen</td></tr>
<tr><td>Area for Displaying Number of Spots and Bikes</td><td>Initial</td><td colspan="2">Displaying Number of Spots and Bikes</td></tr>
<tr><td>Area for Displaying Bike in the Dock</td><td>Initial</td><td colspan="2">Displaying Bike in the Dock</td></tr>
</table>

Screen name	Dock View Screen			
Item name	Number of digits (bytes)	Type	Field attribute	Remarks
Number of Spot	3	Numeral	Black	Right justified
Number of Standard Bike	3	Numeral	Black	Right justified
Number of Twin Bike	3	Numeral	Black	Right justified
Number of E-Bike	3	Numeral	Black	Right justified
Bike Barcode	10	String	Black	Center justified
Battery Left	4	String	Green	Center justified
Bike Type	6	Numeral	Black	Center justified

4.1.3.3 View Bike Screen

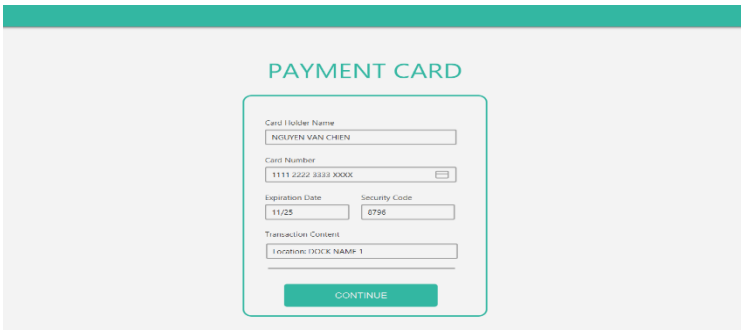
I. EBR Software		Date of creation	Approved by	Reviewed by	Person in charge
Screen specification	View Bike Screen	18/11/2021			Hoàng Hải Đăng



Control	Operation	Function
"x" Button	Click	Display Home Screen
"Return Bike" Button	Click	Display Bike Screen

Screen name	Invoice Screen			
Item name	Number of digits (bytes)	Type	Field attribute	Remarks
Bike code	20	String	Black	Left justified
License Plate	8	String	Black	Left justified
Location	30	String	Black	Left justified
Battery	4	String	Black	Left justified
Deposit	20	String	Black	Left justified
Charge	30	String	Black	Left justified
Pedals	1	Numeric	Black	Left justified
Saddle	1	Numeric	Black	Left justified
Rear Seat	1	Numeric	Black	Left justified

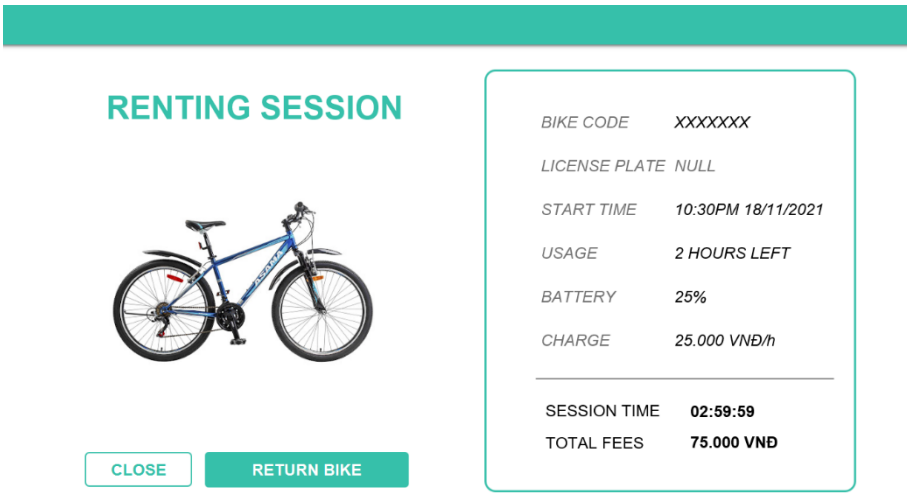
4.1.3.4 Payment Screen

EBR Software		Date of creation	Approved by	Reviewed by	Person in charge
Screen specification	Payment Screen	11/18/2021			Nguyễn Văn Chiến
		Control	Operation	Function	
		Area for display Payment Information	Initial	Display the Card Holder Name, Card Number, Expiration Date, Security Number, Transaction Contents	
		"Continue" Button	Click	Display Invoice Screen	

Screen name	Payment Screen			
Item name	Number of digits (bytes)	Type	Field attribute	Remarks
Card Number	20	String	Black	Left justified
Card Holder Name	20	String	Black	Left justified
Expiration Date	5	String	Black	Left justified
Security Code	4	Numeric	Black	Left justified
Transaction Contents	100	String	Black	Left justified

4.1.3.5 Renting Session Screen

EBR Software		Date of creation	Approved by	Reviewed by	Person in charge
Screen specification	Session Screen	11/18/2021			Nguyễn Văn Chiến

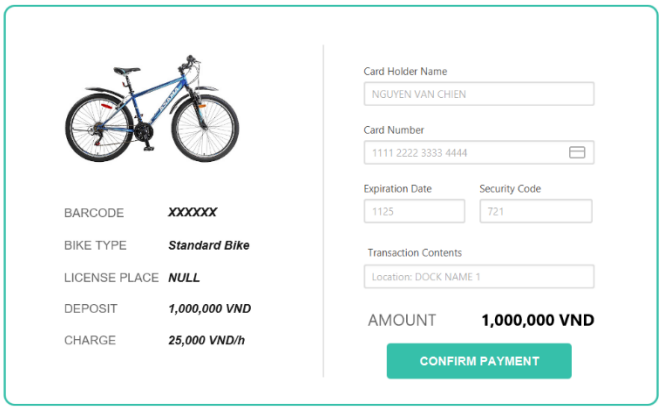


Control	Operation	Function
"Close" Button	Click	Display Home Screen
"Return Bike" Button	Click	Display Bike Screen

Screen name	Invoice Screen			
Item name	Number of digits (bytes)	Type	Field attribute	Remarks
Bike code	20	String	Black	Left justified
License Plate	8	String	Black	Left justified
Start time	30	String	Black	Left justified
Battery	4	String	Black	Left justified
Total Fees	30	String	Black	Left justified
Charge	30	String	Black	Left justified
Barcode	30	String	Black	Left justified
Session Time	8	String	Black	Left justified

4.1.3.6 Invoice Screen

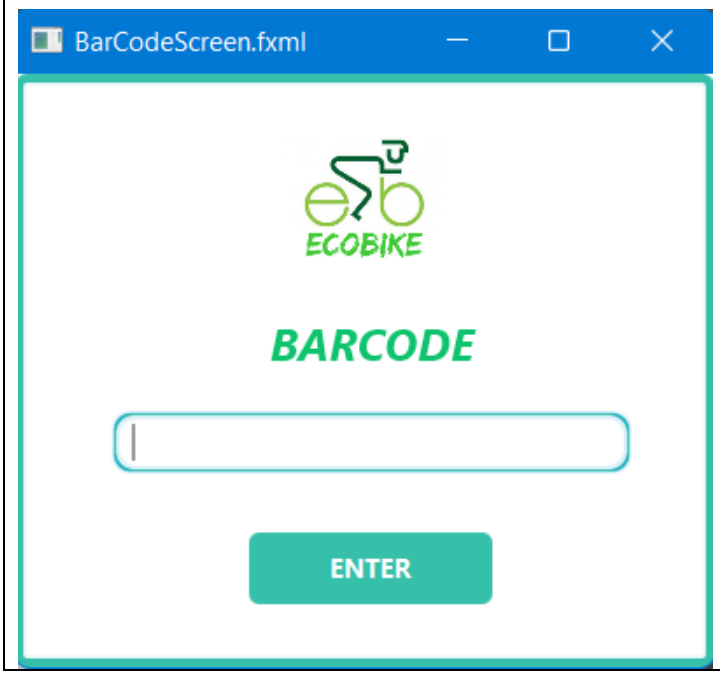
EBR Software		Date of creation	Approved by	Reviewed by	Person in charge
Screen specification	Invoice Screen	11/18/2021			Nguyễn Văn Chiến
		Control	Operation	Function	
		"Confirm Payment" Button	Click	Display Transaction Result Screen	



Screen name	Invoice Screen			
Item name	Number of digits (bytes)	Type	Field attribute	Remarks
Card Number	20	String	Black	Left justified
Card Holder Name	20	String	Black	Left justified
Expiration Date	5	String	Black	Left justified
Security Code	4	Numeric	Black	Left justified
Bike Type	30	String	Black	Left justified
Amount	30	String	Black	Left justified
Charge	30	String	Black	Left justified
Deposit	30	String	Black	Left justified
Barcode	30	String	Black	Left justified
License plate	8	String	Black	Left justified
Transaction Contents	100	String	Black	Left justified

4.1.3.7 Barcode Screen

EBR Software		Date of creation	Approved by	Reviewed by	Person in charge
Screen specification	Barcode Screen	17/11/2021			Nguyễn Mạnh Cường

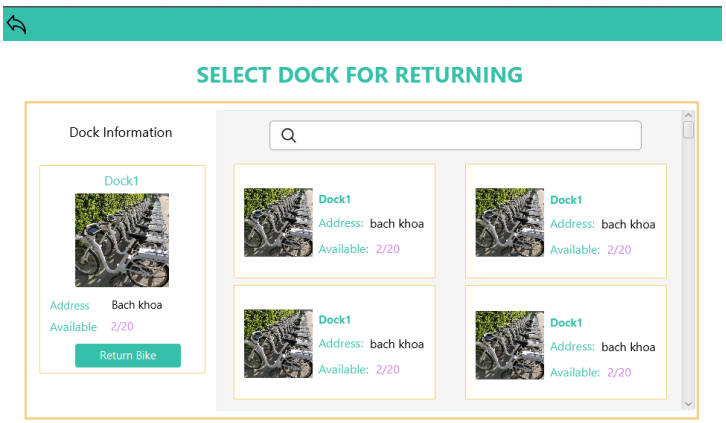


Control	Operation	Function
Area for entering Barcode	Initial	Receive barcode for renting bike
"Enter" Button	Click	Display Renting Bike Screen

Screen name	Barcode Screen			
Item name	Number of digits (bytes)	Type	Field attribute	Remarks
Barcode	8	String	Black	Left justified

4.1.3.8 Select Dock Screen

EBR Software		Date of creation	Approved by	Reviewed by	Person in charge
Screen specification	Dock selection	18/11/2021			Nguyễn Đào Duy Kiên

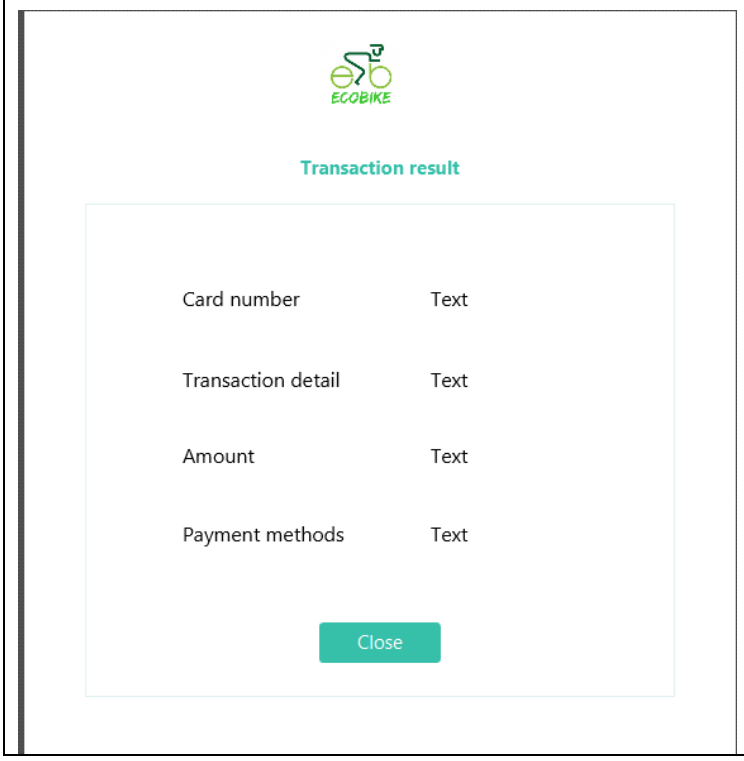


Control	Operation	Function
"Return Bike" Button	Click	Display Payment Screen
Area for entering search	Initial	Receive Text for Searching
"Search" Button	Click	Searching Dock
"Item Dock"	Click	Display item dock in left dock information

Screen name	Home Screen			
Item name	Number of digits (bytes)	Type	Field attribute	Remarks
Search field	50	String	Black	Left justified
Dock's name	50	String	Green	Left justified
Dock's address	50	String	Black	Left justified
Available	10	String	Black	Left justified

4.1.3.9 Transaction Result Screen

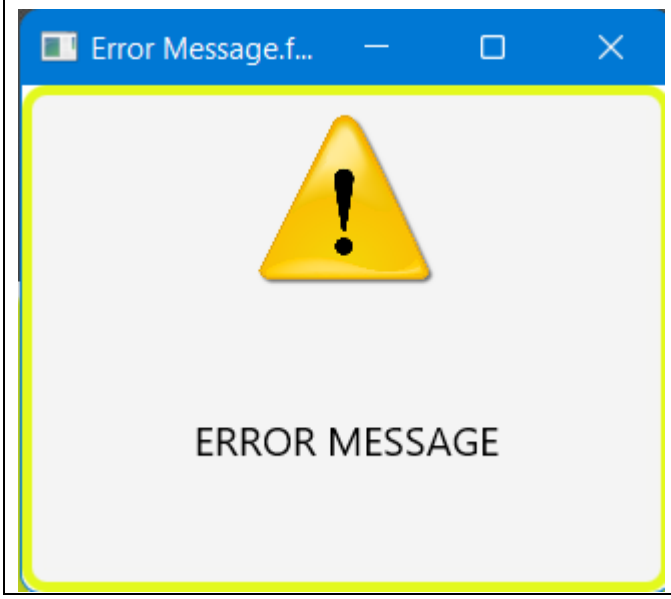
EBR Software		Date of creation	Approved by	Reviewed by	Person in charge
Screen specification	Transaction result	18/11/2021			Nguyễn Đào Duy Kiên



Control	Operation	Function
Close Button	Click	Close popup
Area for display transaction Information	Initial	Display transaction result and transaction info

Screen name	Transaction result			
Item name	Number of digits (bytes)	Type	Field attribute	Remarks
Transaction details	50	number	Black	Left justified
Card number	50	number	Black	Left justified
Amout	50	number	Black	Left justified
Payment method	50	String	Black	Left justified

4.1.3.10 Error Message Screen

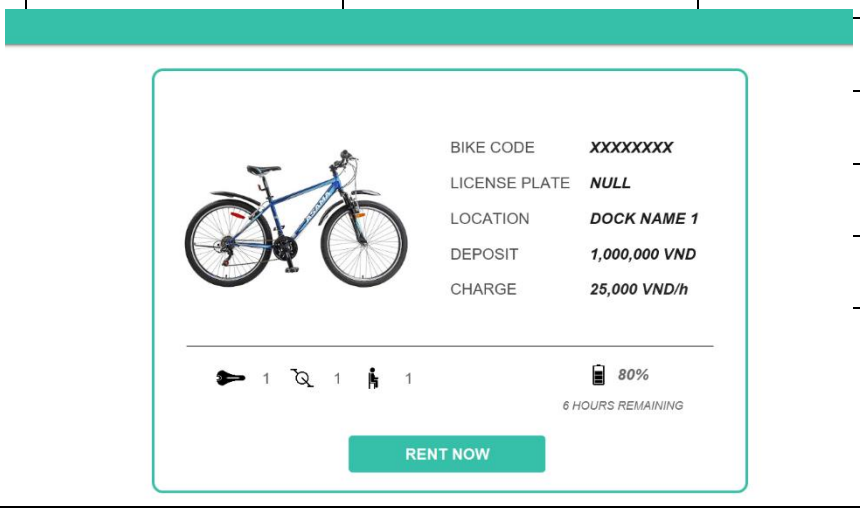
EBR Software		Date of creation	Approved by	Reviewed by	Person in charge
Screen specification	Error Message Screen	17/11/2021			Nguyễn Mạnh Cường
		**Control**	**Operation**	**Function**	
		Area for displaying error message	Initial	Display error message	

Screen name	Error Message Screen			
Item name	Number of digits (bytes)	Type	Field attribute	Remarks
Error Message	100	String	Black	Center

4.1.3.11 Rent Bike Screen

EBR Software		Date of creation	Approved by	Reviewed by	Person in charge
Screen specification	Rent Bike Screen	11/18/2021			Nguyễn Văn Chiến
		Control	Operation	Function	

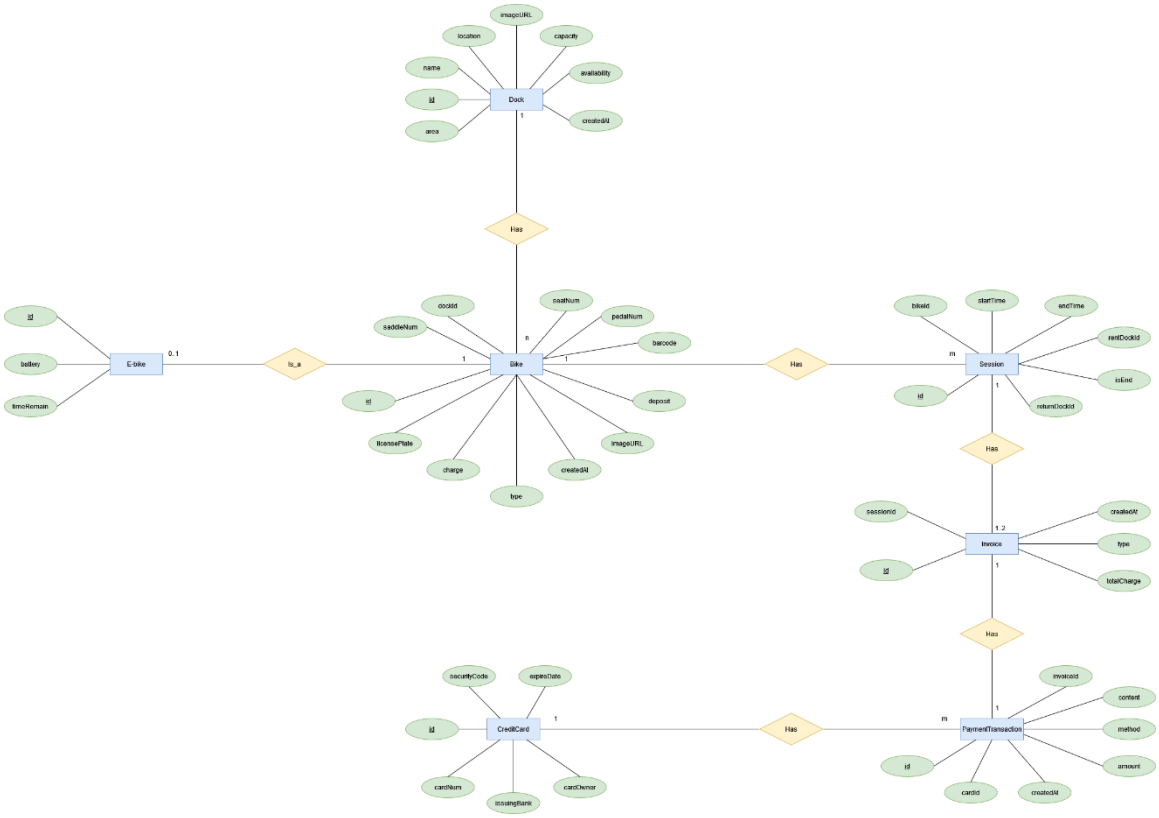
Screen name	Rent Bike Screen			
Item name	Number of digits (bytes)	Type	Field attribute	Remarks
Bike Code	8	String	Black	Left justified
			Black	Left justified
			Black	Left justified
			Black	Left justified
			Black	Left justified



	Enter "Rent Now" Button	Click		Display Payment Screen

4.2 Data Modeling

4.2.1 Conceptual Data Modeling

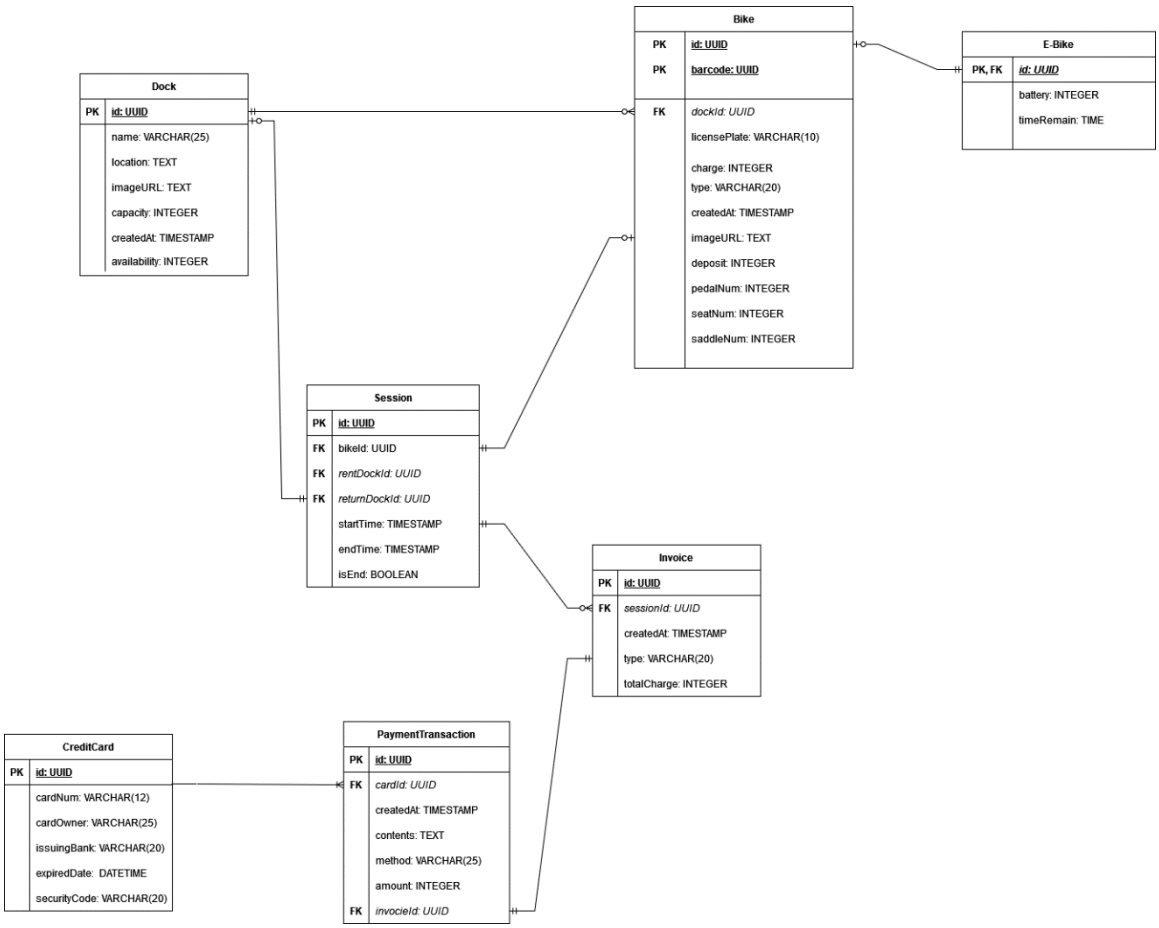


4.2.2 Database Design

4.2.2.1 Database Management Systems

Database Management System: PostgreSQL

4.2.2.2 Logical Data Model



4.2.2.3 Physical Data Model

Dock

#	PK	FK	Column Name	Data Type	Mandatory	Description
1	X		id	UUID	Yes	Dock ID
2			name	VARCHAR(25)	Yes	Name of the dock
3			location	TEXT	Yes	Address of the dock
4			imageURL	TEXT	Yes	URL of image of dock
5			capacity	INTEGER	Yes	Number of slots in the dock

6			createdAt	TIMESTAMP	Yes	Created Time
7			availability	INTEGER	Yes	Number of available bikes in dock

Bike

#	PK	FK	Column Name	Data Type	Mandatory	Description
1	X		id	UUID	Yes	Bike ID
2	X		barcode	UUID	Yes	Barcode for renting
3		X	dockId	UUID	Yes	Dock ID
4			licensePlate	VARCHAR(10)	Yes	License Plate
5			charge	INTEGER	Yes	Charge to rent bike per hour
6			type	VARCHAR(20)	Yes	Bike type
7			createdAt	TIMESTAMP	Yes	Created time
8			imageURL	TEXT	Yes	URL of image of bike
9			deposit	INTEGER	Yes	Deposit of renting
10			seatNum	INTEGER	Yes	Number of seats
11			pedalNum	INTEGER	Yes	Number of pedals
12			saddleNum	INTEGER	Yes	Number of saddles

E_Bike

#	PK	FK	Column Name	Data Type	Mandatory	Description
1	X	X	id	UUID	Yes	Bike ID
2			battery	INTEGER	Yes	Battery left percentage
3			timeRemain	TIME	Yes	Time remaining

Session

#	PK	FK	Column Name	Data Type	Mandatory	Description
1	X		id	UUID	Yes	Session ID
2		X	bikeId	UUID	Yes	Bike Id
3		X	rentDockId	UUID	Yes	Reting Dock ID
4		X	returnDockId	UUID	Yes	Returning Dock ID
5			start_time	TIMESTAMP	Yes	Start time of session
6			endTime	TIMESTAMP	No	End time of session
7			isEnd	BOOLEAN	Yes	Check session finished

Invoice

#	PK	FK	Column Name	Data Type	Mandatory	Description
1	X		id	UUID	Yes	Invoice ID
2		X	sessionId	UUID	Yes	Session ID
3			createdAt	TIMESTAMP	Yes	Created time
			type	VARCHAR(20)	Yes	Type of Invoice
5			totalCharge	INTEGER	Yes	Total charge money

PaymentTransaction

#	PK	FK	Column Name	Data Type	Mandatory	Description
1	X		id	UUID	Yes	Payment Transaction ID
2		X	cardId	UUID	Yes	Card ID

3			createAt	TIMESTAMP	Yes	Created transaction time
4			contents	TEXT	No	Transaction contents
5			method	VARCHAR(25)	Yes	Method of payment
6			amount	INTEGER	Yes	Transaction amount
7		X	invoiceId	UUID	Yes	Invoice ID

CreditCard

#	PK	FK	Column Name	Data Type	Mandatory	Description
1	X		id	UUID	Yes	Card ID
2			cardNum	VARCHAR(12)	Yes	Card number
3			cardOwner	VARCHAR(25)	Yes	Card holder name
4			issuingBank	VARCHAR(20)	Yes	Issuing Bank
5			expiredDate	DATETIME	Yes	Expired date
6			securityCode	VARCHAR(20)	Yes	Security code

SQL Script

SET statement_timeout = 0;

SET LOCK_TIMEOUT = 0;

SET idle_in_transaction_session_timeout = 0;

SET client_encoding = 'UTF8';

SET standard_conforming_strings = ON;

SELECT pg_catalog.set_config('search_path', '', false);

SET check_function_bodies = false;

SET xmloption = content;

SET client_min_messages = warning;

```
SET row_security = OFF;
SET default_tablespace = '';

CREATE EXTENSION IF NOT EXISTS adminpack WITH SCHEMA pg_catalog;
CREATE EXTENSION IF NOT EXISTS "uuid-ossp" WITH SCHEMA PUBLIC;
DROP TABLE IF EXISTS PUBLIC.dock;
DROP TABLE IF EXISTS PUBLIC.bike;
DROP TABLE IF EXISTS PUBLIC.e_bike;
DROP TABLE IF EXISTS PUBLIC.session;
DROP TABLE IF EXISTS PUBLIC.invoice;
DROP TABLE IF EXISTS PUBLIC.paymenttransaction;
DROP TABLE IF EXISTS PUBLIC.card;

CREATE TABLE PUBLIC.dock (
        id uuid DEFAULT PUBLIC.uuid_generate_v4() NOT NULL
        , name VARCHAR(25) NOT NULL
        , location TEXT NOT NULL
        , imageURL TEXT NOT NULL
        , capacity INTEGER NOT NULL DEFAULT 0
        , createdAt TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP
        , availability INTEGER NOT NULL
        );

ALTER TABLE PUBLIC.dock OWNER TO postgres;

CREATE TABLE PUBLIC.bike (
        id uuid NOT NULL DEFAULT PUBLIC.uuid_generate_v4()
        , barcode uuid NOT NULL DEFAULT PUBLIC.uuid_generate_v4()
        , dockId uuid NOT NULL
```

```
    , licensePlate VARCHAR(10) NOT NULL
    , charge INTEGER NOT NULL
    , type VARCHAR(20) NOT NULL
    , createdAt TIMESTAMP DEFAULT CURRENT_TIMESTAMP NOT NULL
    , imageURL TEXT NOT NULL
    , deposit INTEGER NOT NULL
    , seatNum INTEGER NOT NULL
    , pedalNum INTEGER NOT NULL
    , saddleNum INTEGER NOT NULL
    );

ALTER TABLE PUBLIC.bike OWNER TO postgres;

CREATE TABLE PUBLIC.e_bike (
    id uuid NOT NULL
    , battery INTEGER NOT NULL
    , timeRemain TIME NOT NULL
    );

ALTER TABLE PUBLIC.e_bike OWNER TO postgres;

CREATE TABLE PUBLIC.session (
    id uuid NOT NULL DEFAULT PUBLIC.uuid_generate_v4()
    , bikeId uuid NOT NULL
    , rentDockId uuid NOT NULL
    , returnDockId uuid NOT NULL
    , startTime TIMESTAMP NOT NULL
    , endTime TIMESTAMP
    , isEnd boolean NOT NULL
```

```
);

ALTER TABLE PUBLIC.session OWNER TO postgres;

CREATE TABLE PUBLIC.invoice (
    id uuid NOT NULL DEFAULT PUBLIC.uuid_generate_v4()
    ,sessionId uuid NOT NULL
    ,createAt TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP
    ,type VARCHAR(20) NOT NULL
    ,totalCharge INTEGER NOT NULL
    );

ALTER TABLE PUBLIC.invoice OWNER TO postgres;

CREATE TABLE PUBLIC.paymentTransaction (
    id uuid NOT NULL DEFAULT PUBLIC.uuid_generate_v4()
    ,cardId uuid NOT NULL
    ,createdAt TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP
    ,contents TEXT
    ,method VARCHAR(25) NOT NULL
    ,amount INTEGER NOT NULL
    ,invoiceId UUID NOT NULL
    );

ALTER TABLE PUBLIC.paymentTransaction OWNER TO postgres;

CREATE TABLE PUBLIC.card (
    id uuid NOT NULL DEFAULT PUBLIC.uuid_generate_v4()
    , cardNum VARCHAR(12) NOT NULL
```

```
    , cardOwner VARCHAR(25) NOT NULL
    , issuingBank VARCHAR(20) NOT NULL
    , expiredDate DATE NOT NULL
    , securityCode VARCHAR(20) NOT NULL
    );

ALTER TABLE PUBLIC.card OWNER TO postgres;

INSERT VALUES INTO SOME TABLES

-- DOCK
INSERT INTO PUBLIC.dock (
    name
    , location
    , imageURL
    , capacity
    , availability
    )
    VALUES (
        'ECO Bike Park 1'
        , 'Location 1'
        , ''
        , 200
        , 10
     )
    ,(
        'ECO Bike Park 2'
        , 'Location 2'
        , ''
```

```
        , 150
        , 15
    )
    ,(
        'ECO Bike Park 3'
        , 'Location 3'
        , ''
        , 100
        , 20
    );

-- BIKE
INSERT INTO PUBLIC.bike (
    dockId
    , licensePlate
    , charge
    , type
    , imageURL
    , deposit
    , seatNum
    , pedalNum
    , saddleNum
    )
    VALUES (
        'd33190a0-5674-4951-93ad-cbee666edb12'
        , '30MD1-1234'
        , 10000
        , 'Xe đạp điện'
        , ''
```

```
        , 200000
        , 2
        , 1
        , 1
    )
    ,(
        '70c784e0-3916-48b1-a309-3ff6853dd98c'
        , ''
        , 5000
        , 'Xe đạp'
        , ''
        , 50000
        , 1
        , 1
        , 1
    );

-- E_Bike
INSERT INTO PUBLIC.e_bike (
    id
    , battery
    , timeRemain
    )
    VALUES (
        'dff53949-73db-494b-a8eb-57e5738918bc'
        , 70
        , '08:00:00'
    );
```

-- ADD PRIMARY KEY

ALTER TABLE ONLY PUBLIC.dock ADD CONSTRAINT dock_pk PRIMARY KEY (id);

ALTER TABLE ONLY PUBLIC.bike ADD CONSTRAINT bike_pk PRIMARY KEY (id);

ALTER TABLE ONLY PUBLIC.e_bike ADD CONSTRAINT e_bike_pk PRIMARY KEY (id);

ALTER TABLE ONLY PUBLIC.session ADD CONSTRAINT session_pk PRIMARY KEY (id);

ALTER TABLE ONLY PUBLIC.invoice ADD CONSTRAINT invoice_pk PRIMARY KEY (id);

ALTER TABLE ONLY PUBLIC.paymenttransaction ADD CONSTRAINT payment_transaction_pk PRIMARY KEY (id);

ALTER TABLE ONLY PUBLIC.card ADD CONSTRAINT card_pk PRIMARY KEY (id);

-- ADD FOREIGN KEY

ALTER TABLE ONLY PUBLIC.bike ADD CONSTRAINT bike_fk FOREIGN KEY (dockid) REFERENCES PUBLIC.dock (id) ON DELETE CASCADE;

ALTER TABLE ONLY PUBLIC.e_bike ADD CONSTRAINT e_bike_fk FOREIGN KEY (id) REFERENCES PUBLIC.bike (id) ON DELETE CASCADE;

ALTER TABLE ONLY PUBLIC.session ADD CONSTRAINT session_fk1 FOREIGN KEY (bikeId) REFERENCES PUBLIC.bike (id);

ALTER TABLE ONLY PUBLIC.session ADD CONSTRAINT session_fk2 FOREIGN KEY (rentdockid) REFERENCES PUBLIC.dock (id);

ALTER TABLE ONLY PUBLIC.session ADD CONSTRAINT session_fk3 FOREIGN KEY (returndockid) REFERENCES PUBLIC.dock (id);

ALTER TABLE ONLY PUBLIC.invoice ADD CONSTRAINT invoice_fk FOREIGN KEY (sessionid) REFERENCES PUBLIC.session (id);

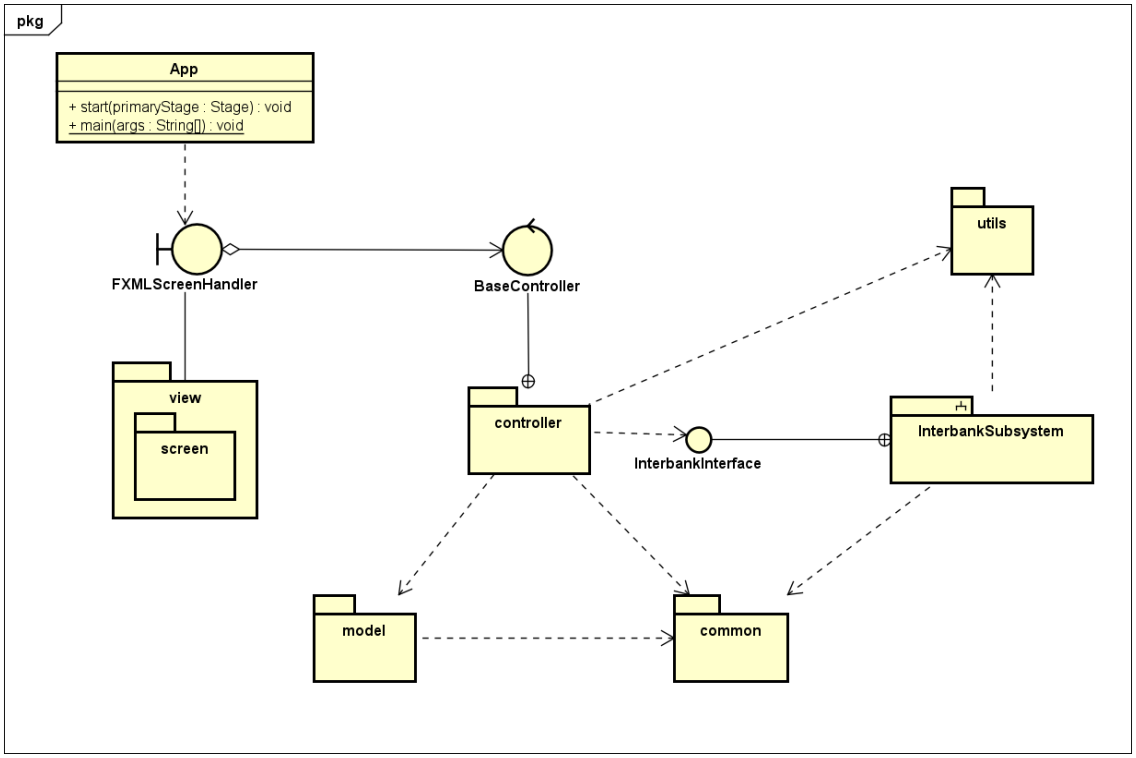
ALTER TABLE ONLY PUBLIC.paymenttransaction ADD CONSTRAINT payment_transaction_fk1 FOREIGN KEY (cardid) REFERENCES PUBLIC.card (id);

ALTER TABLE ONLY PUBLIC.paymenttransaction ADD CONSTRAINT payment_transaction_fk2 FOREIGN KEY (invoiceid) REFERENCES PUBLIC.invoice (id);

4.3 Non-Database Management System Files

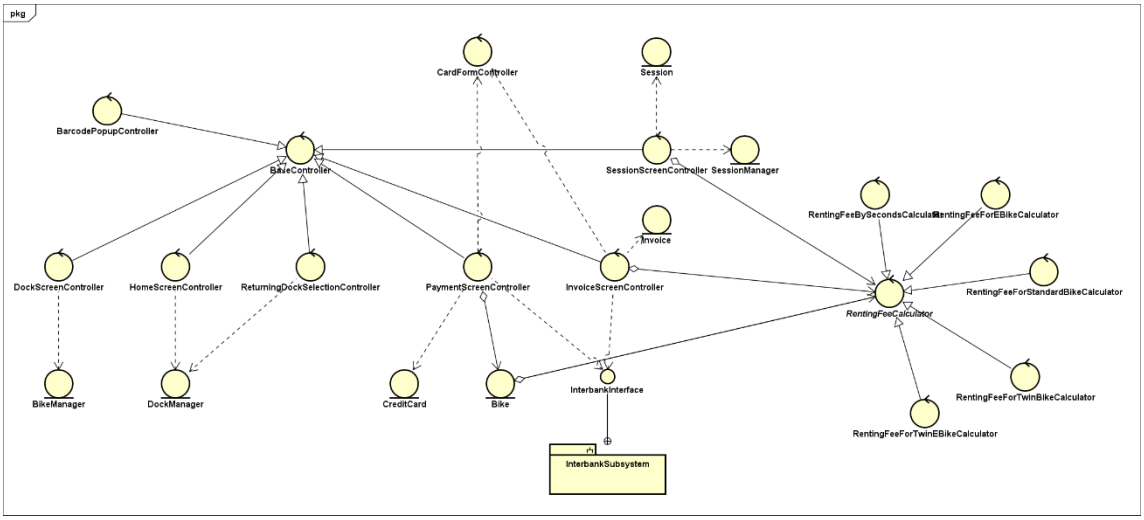
4.4 Class Design

4.4.1 General Class Diagram

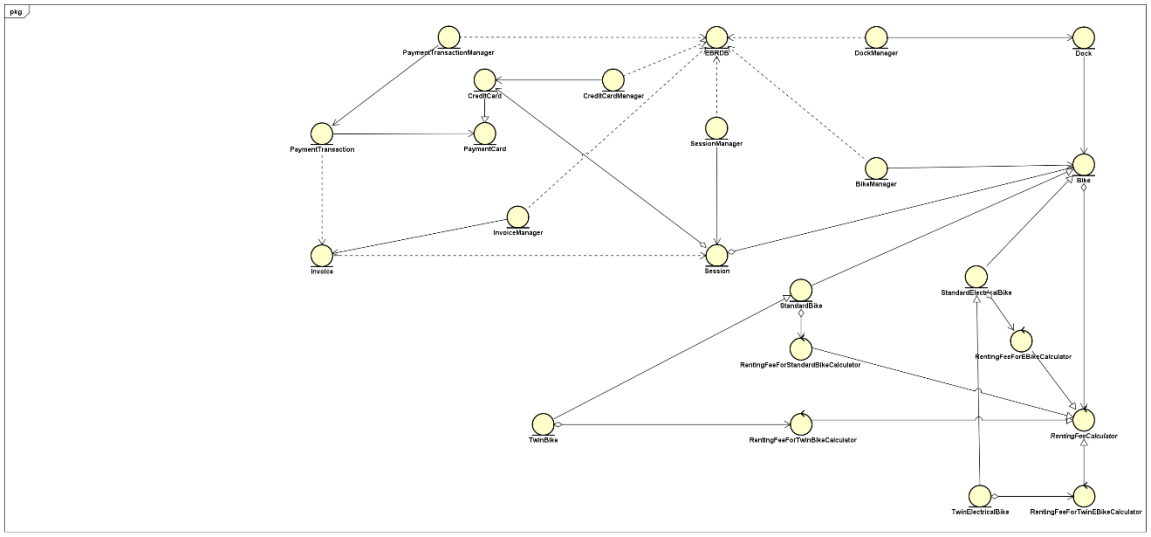


4.4.2 Class Diagrams

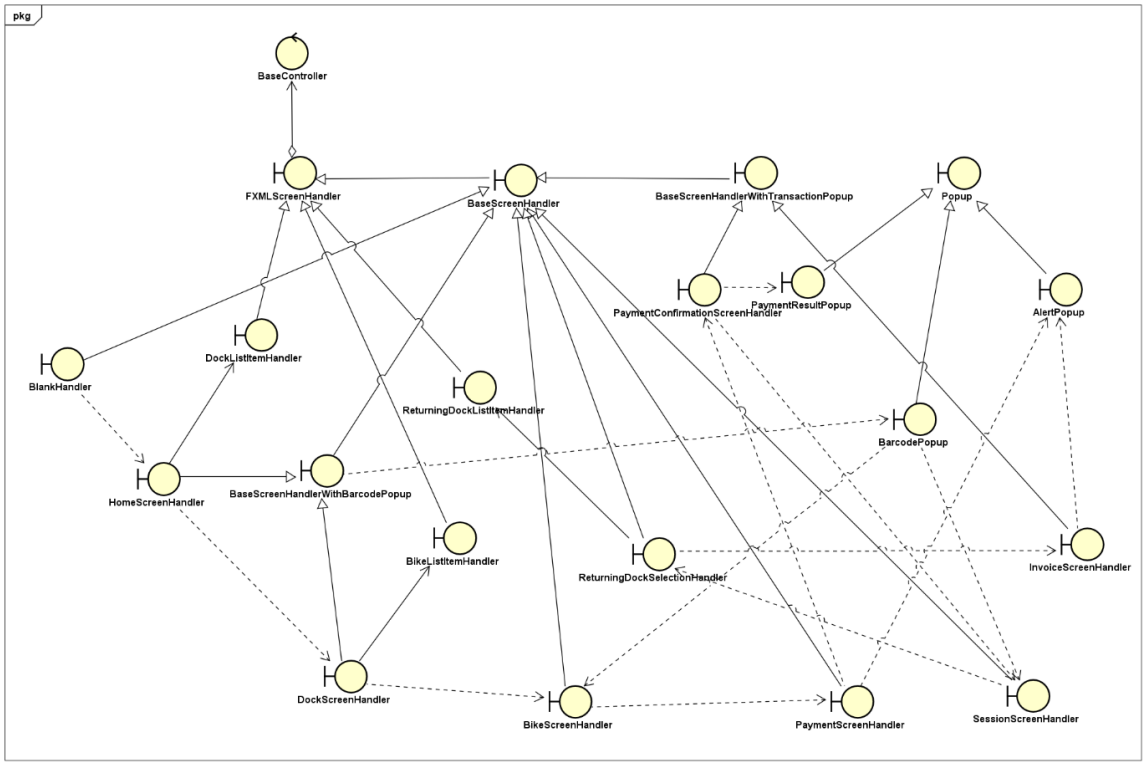
4.4.2.1 Class Diagram for Package "controller"



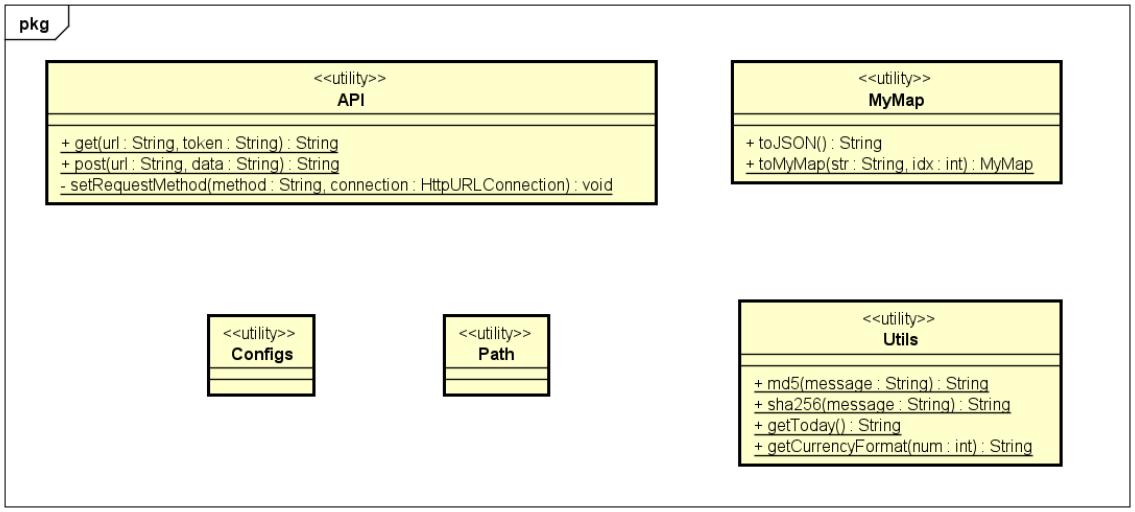
4.4.2.2 Class Diagram for Package “model”



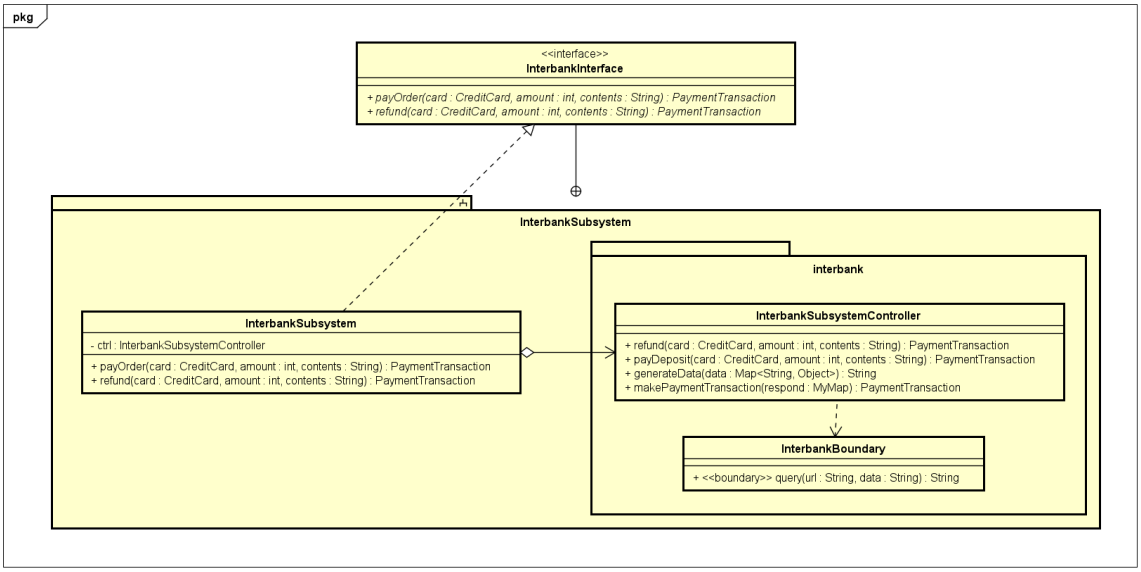
4.4.2.3 Class Diagram for Package “view”



4.4.2.4 Class Diagram for Package "utils"

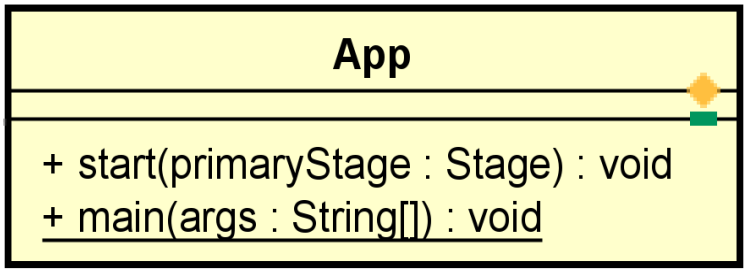


4.4.2.5 Class Diagram for Subsystem "interbank"



4.4.3 Class Design

- **Class "App"**



Attribute

None

Operation

#	Name	Return type	Description
1	start	void	javaFx framwork will call this
2	main	void	execution entry

Parameter:

- start()
  - stage: javaFx main stage

Exception:

None

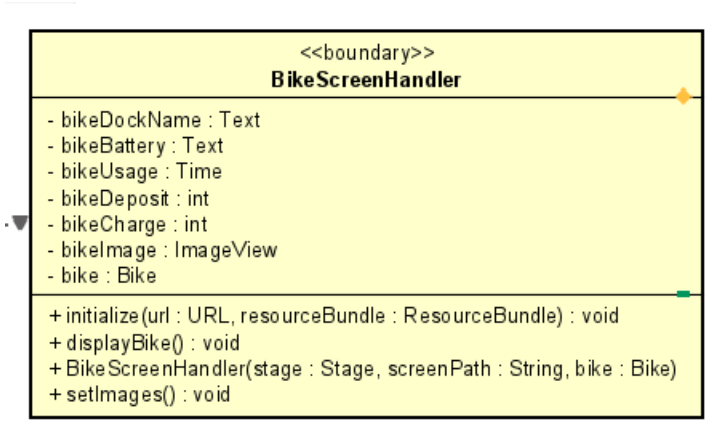
Method

None

State

None

- **Class "BikeScreenHandler"**



Attribute

#	Name	Data type	Default value	Description
1	bikeDockName	Text	NULL	Dock name fxml
2	bikeBattery	int	NULL	Bike battery fxml
3	bikeUsage	Time	NULL	Bike usage fxml
4	bikeDeposit	int	NULL	Bike deposit fxml
5	bikeCharge	int	NULL	Bike charge fxml
6	bikeImage	ImageView	NULL	Bike image
7	bike	Bike	NULL	bike

Operation

#	Name	Return type	Description
1	initialize	void	initialize
2	displayBike	void	Display bike info screen
3	setImages	void	Set bike image

Parameter:

- initialize
  - url - URL
  - resourceBundle - ResourceBundle
- BikeScreenHandle
  - stage - Stage

Exception:

None

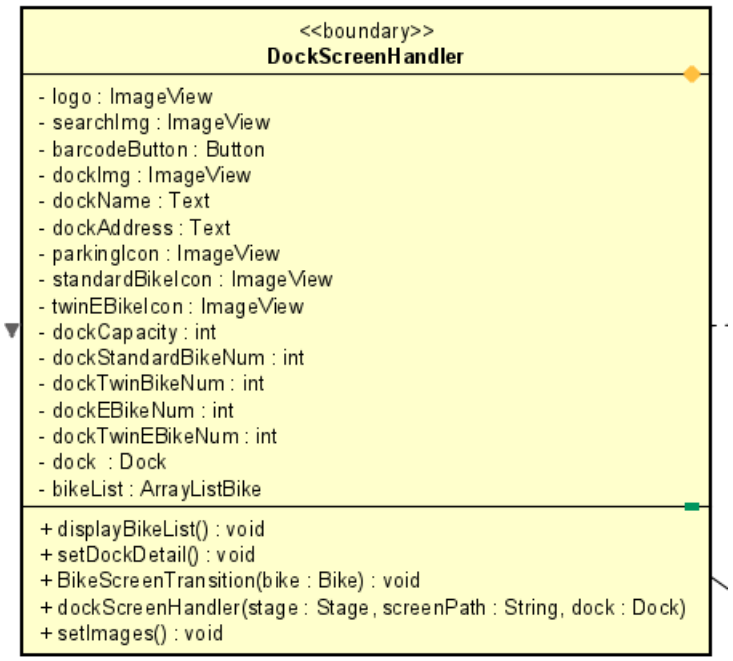
Method

None

State

None

- **Class "DockScreenHandler"**



Attribute

#	Name	Data type	Default value	Description
1	logo	ImageView	NULL	Logo image fxml
2	searchImg	imageView	NULL	Search image fxml
3	barcodeButton	Button	NULL	BarcodeButton fxml
4	dockName	Text	NULL	Dock name
5	dockAddress	Text	NULL	Dock address
6	dockImg	ImageView	NULL	Dock image fxml
7	parkingIcon	ImageView	NULL	Parking icon fxml
8	standardBikeIcon	ImageView	NULL	Standard bike icon fxml
9	twinEBikeIcon	ImageView	NULL	Twin bike icon fxml
10	dockStandardBikeNum	Int	NULL	Standard bike number

11	dockTwinBikeNum	Int	NULL	Twin bike number f
12	dockEBikeNum	Int	NULL	E-bike number
13	dockTwinEBikeNum	Int	NULL	Twin e-bikenumber
14	dock	Dock	NULL	Dock
15	bikeList	ArrayList<Bike>	NULL	Bike list in dock

Operation

#	**Name**	**Return type**	**Description**
1	displayBikeList	void	Display bike list
2	setDockDetail	void	Set dock detail
3	BikeScreenTransition	void	Transition to bike screen
4	setImages	void	Set dock image

Parameter:

- BikeScreenTransition
  - bike
- dockScreenHandle
  - stage – Stage
  - screenPath – String
  - dock - Dock

Exception:

None

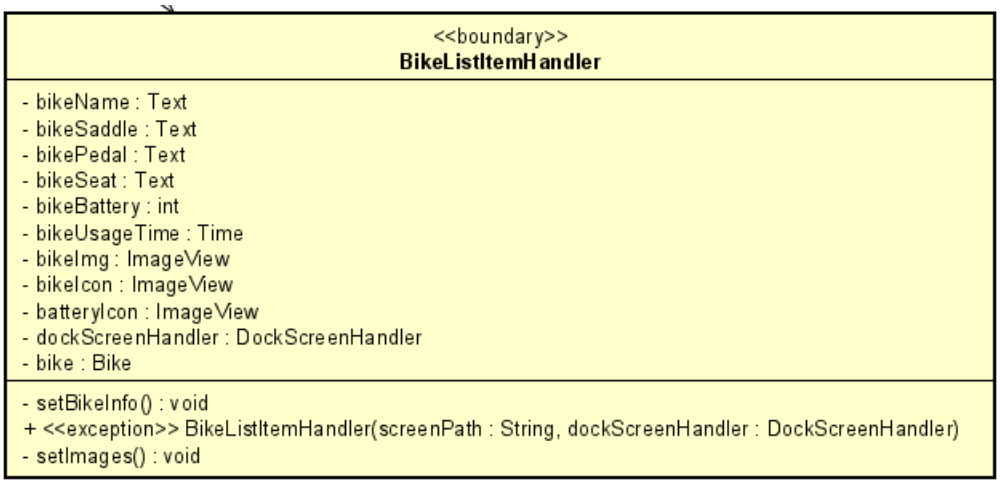
Method

None

State

None

- **Class "BikeListItemHandler"**



Attribute

#	Name	Data type	Default value	Description
1	bikeImg	ImageView	NULL	Bike image fxml
2	bikeIcon	imageView	NULL	Bike image fxml
4	bikeName	Text	NULL	Bike name
5	bikeSaddle	Text	NULL	Bike saddle
6	bikePedal	Text	NULL	Bike pedal
7	bikeSeat	Text	NULL	Bike seat
8	dockScreenHandler	DockScreenHandler	NULL	object
9	bike	Bike	NULL	object
10	bikeBattery	Int	NULL	Bike Battery
11	BikeUsageTime	Time	NULL	Bike Usage Time

Operation

#	Name	Return type	Description
1	setBikeInfo	void	Set bike info
2	setImages	void	Set bike image in list

Parameter:

none

Exception:

IOException

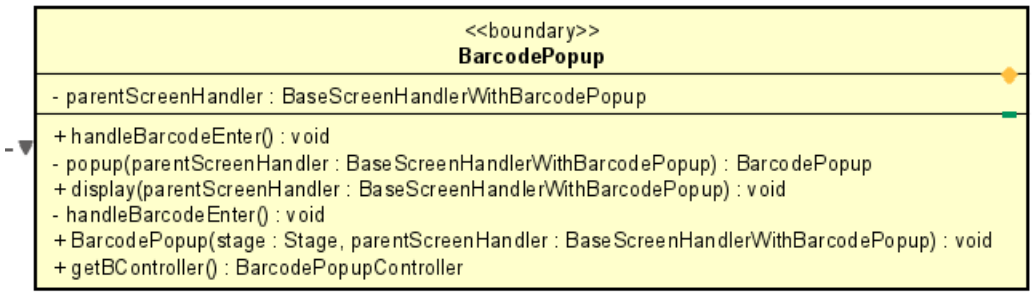
Method

None

State

None

- **Class "BarcodePopup"**



Attribute

#	Name	Data type	Default value	Description
1	parentScreenHandler	BaseScreenHandlerWithBarcodePopup	NULL	Parent screen

Operation

#	Name	Return type	Description
1	handleBarcodeEnter	void	Handle action "enter barcode"
2	popup	BaseScreenHandlerWithBarcodePopup	BarcodePopup
3	display	void	Display popup

4	handleBarcodeEnter	void	Button enter handler
5	barcodePopup	void	Popup barcode
6	getBController	BarcodePopupController	Get barcode

Parameter:

- popup
  - parentScreenHandler
- display
  - parentScreenHandler
- barcodePopup
  - parentScreenHandler
  - stage

Exception:

None

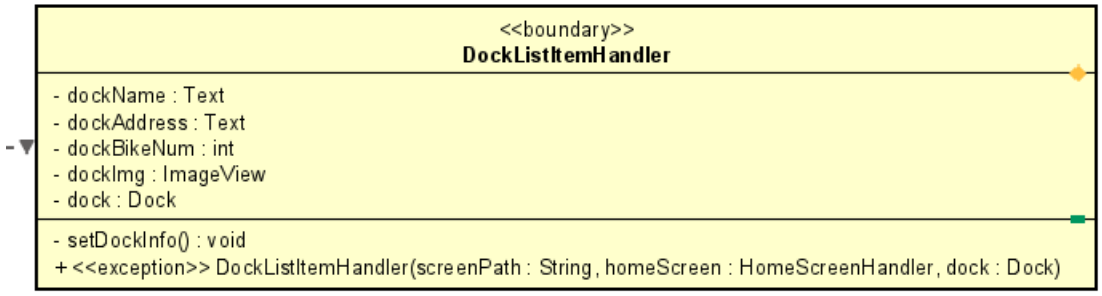
Method

None

State

None

- **Class "DockListItemHandler"**



Attribute

#	Name	Data type	Default value	Description
1	dockName	Text	NULL	Dock name
2	dockAddress	Text	NULL	Dock address

3	dockBikeNum	int	NULL	Number of bikes in dock
4	dockImg	ImageView	NULL	Dock image
5	dock	Dock	NULL	dock

Operation

#	**Name**	**Return type**	**Description**
1	setDockInfo	void	Set dock info

Parameter:

None

Exception:

IOException

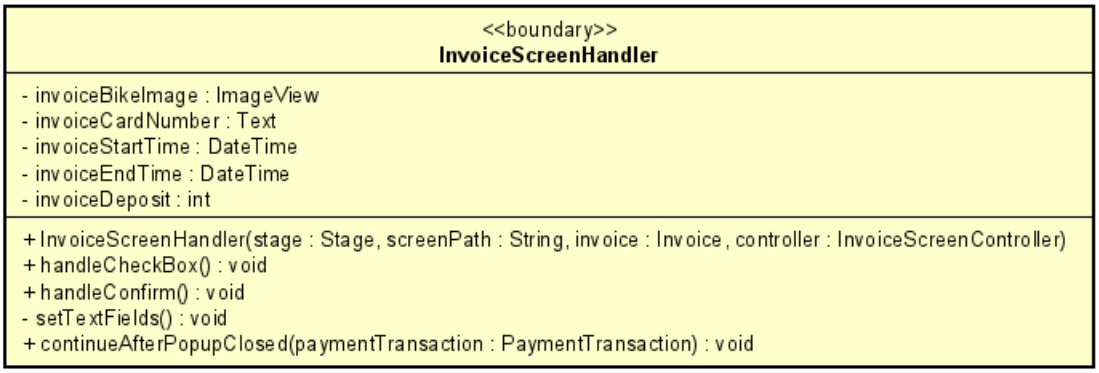
Method

None

State

None

- **Class "InvoiceScreenHandler"**



Attribute

#	Name	Data type	Default value	Description
2	invoiceBikeImage	ImageView	NULL	Bike image
3	invoiceCardNumber	Text	NULL	Invoice card number
4	invoiceStartTime	DateTime	NULL	Invoice start time
5	invoiceEndTime	DateTime	NULL	Invoice end time
6	invoiceDeposit	int	NULL	Invoice deposit

Operation

#	Name	Return type	Description
1	handleCheckBox	void	Handle check box
2	handleConfirmButton	void	Handle confirm button
3	setTextFields	void	Set text fields
4	continueAfterPopupClosed	void	Continue program after popup close

Parameter:

- continueAfterPopupClosed
    - paymentTransaction

Exception:

None

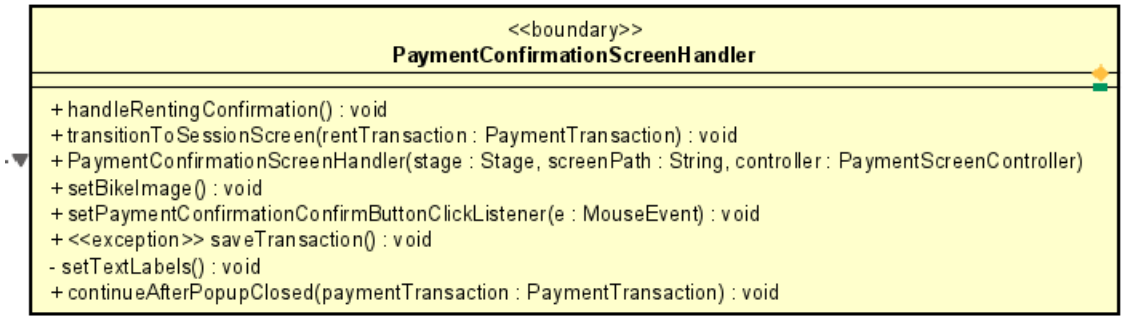
Method

None

State

None

- **Class "PaymentConfirmationScreenHandler"**



Attribute

#	Name	Data type	Default value	Description

Operation

#	Name	Return type	Description
1	handleRentingConfirmation	void	Handle confirmation action
2	transitionToSessionScreen	void	Transition to session screen
3	paymentConfirmationScreenHandler	void	Handle confirm payment
4	setBikeImage	void	Set bike image
5	setPaymentConfirmationConfirmButtonClickListener	void	Payment Confirmation Confirm Button Click Listener
6	saveTransaction	void	Save transaction
7	setTextLabels	void	Set text
8	continueAfterPopupClosed	void	Continue program after popup close

Parameter:

- transitionToSessionScreen
  - rentTransaction
- paymentConfirmationScreenHandler
  - stage
  - screenPath
  - controller
- setPaymentConfirmationConfirmButtonClickListener
  - e
- continueAfterPopupClosed
  - paymentTransaction

Exception:

IOException

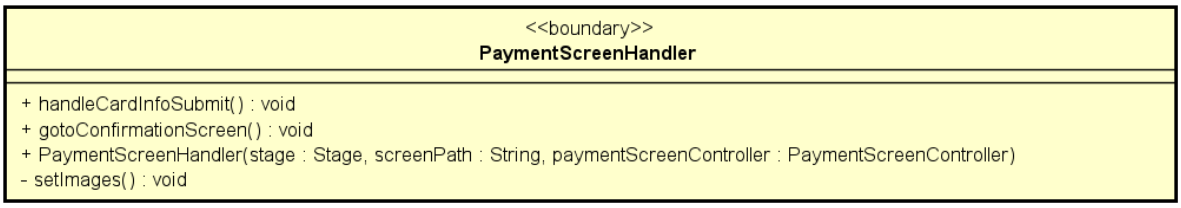
Method

None

State

None

- **Class "PaymentScreenHandler"**



Attribute

#	Name	Data type	Default value	Description

Operation

#	**Name**	**Return type**	**Description**

1	handleCardInfoSubmit	void	Handle action submit card
2	gotoConfirmationScreen	void	Transition to confirmation screen
3	setImages	vodi	Set payment image

Parameter:

None

Exception:

None

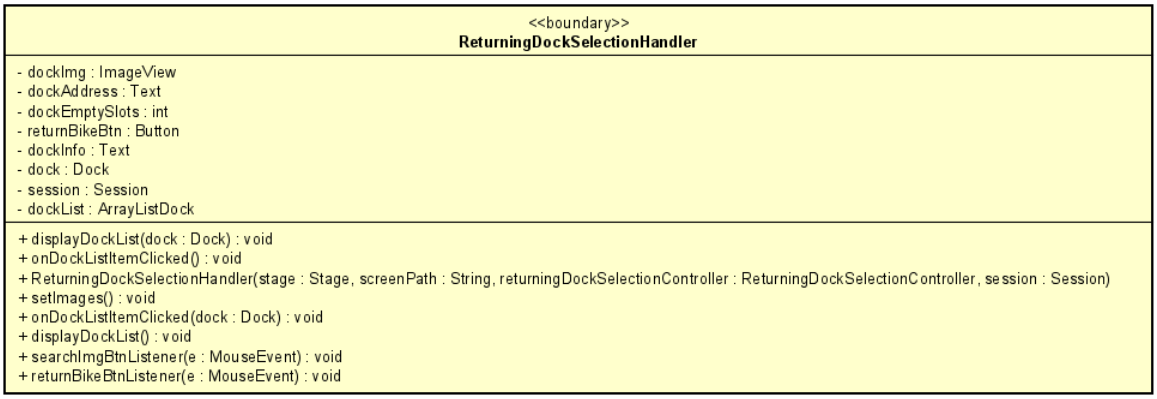
Method

None

State

None

- **Class "ReturningDockListSelectionHandler"**



Attribute

#	Name	Data type	Default value	Description
1	returnBikeBtn	Button	NULL	Button to return bike
2	dockEmptySlots	int	NULL	Dock parking slot
3	dockImg	ImageView	NULL	Dock image fxml
4	dockInfo	Text	NULL	Dock information
5	dockAdress	Text	NULL	Dock address

6	dock	Dock	NULL	dock
7	session	Session	NULL	Session
8	dockList	ArrayListDock	NULL	Dock list

Operation

#	Name	Return type	Description
1	setDockInfo	void	Set dock info
2	onDockListItemClicked	void	Return dock clicked listener
3	setImages	void	Set return dock image
4	displayDockList	void	Display return dock list
5	searchImgBtnListener	void	Search image button listener
6	returnBikeBtnListener	void	Return bike button listener

Parameter:

None

Exception:

None

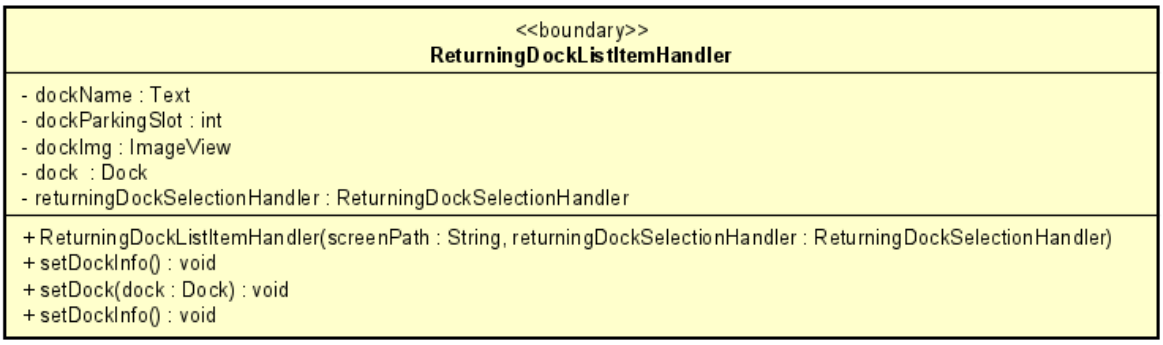
Method

None

State

None

- **Class "ReturningDockListItemHandler"**



Attribute

#	Name	Data type	Default value	Description
1	dockName	Text	NULL	Dock name fxml
2	dockParkingSlot	int	NULL	Dock address fxml
3	dockImg	ImageView	NULL	Dock image fxml
4	dock	Dock	NULL	dock
5	returningDockSelectionHandler	ReturningDockSelectionHandler	NULL	Returning Dock Selection Handler

Operation

#	Name	Return type	Description
1	setDockInfo	void	Display dock information
2	setDock	void	Display dock

Parameter:

- setDock

- dock

Exception:

None

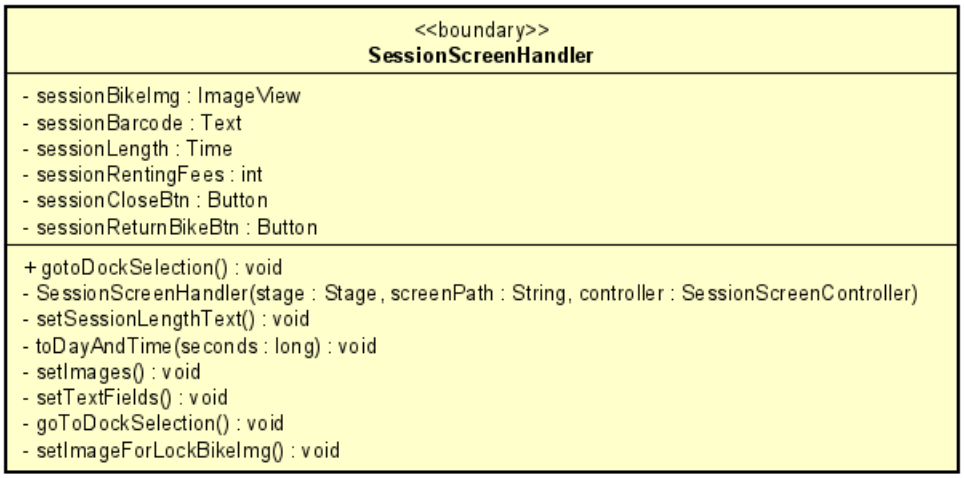
Method

None

State

None

- **Class "SessionScreenHandler"**



Attribute

#	Name	Data type	Default value	Description
1	sessionBikeImg	ImageView	NULL	Bike image fxml
2	sessionBarcode	Text	NULL	Session barcode
3	sessionLength	Time	NULL	Session length
4	sessionRentingFees	Int	NULL	Session renting fees
5	sessionCloseBtn	Button	NULL	Button
6	sessionReturnBikeBtn	Button	NULL	Button

Operation

#	Name	Return type	Description
1	gotoDockSeletion	void	Transition to dock selection
2	setSessionLengthText	void	Set session's length text
3	toDayAndTime	void	Day and time conversion
4	setImages	void	Set image for session
5	setTextFields	void	Set text fields for session
6	goToDockSelection	void	Back to the dock selection
7	setImageForLockBikeImg	void	Set image for bike

Parameter:

- ToDayAndTime
  - seconds

Exception:

None

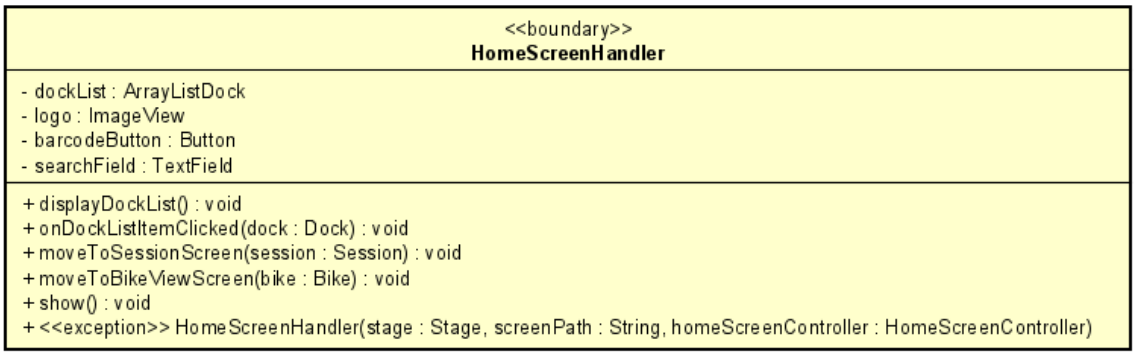
Method

None

State

None

- **Class "HomeScreenHandler"**



Attribute

#	Name	Data type	Default value	Description
1	vboxDockList	VBox	NULL	Vbox fxml
2	dockList	ArrayListDock	NULL	Dock list
3	logo	ImageView	NULL	Logo fxml
4	barcodeButton	Button	NULL	Barcode button fxml
5	searchField	TextField	NULL	Search field fxml

Operation

#	Name	Return type	Description
1	displayDockList	void	Display dock list
2	onDockListItemClicked	void	Dock list item clicked listener
3	moveToSessionScreen	void	Move to session screen
4	moveToBikeViewScreen	void	Move to bike view screen
5	show	void	Show screen

Parameter:

- onDockListItemClicked
    - dock
- moveToSessionScreen
    - session
- moveToBikeViewScreen
    - bike

Exception:

IOException

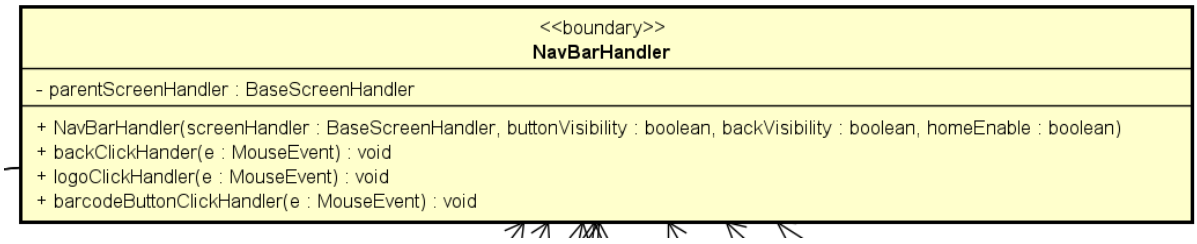
Method

None

State

None

- **Class "NavBarHandler"**



Attribute

#	Name	Data type	Default value	Description
1	parentScreenHandler	BaseScreenHandler	NULL	handler

Operation

#	Name	Return type	Description
1	backClickHandler	void	go to previous screen
2	logoClickHandler	void	go to home screen
3	barcodeButtonClickHandler	void	bring up barcode popup

Parameter:

- backClickHandler
  - e – MouseEvent
- logoClickHandler
  - e - MouseEvent

Exception:

None

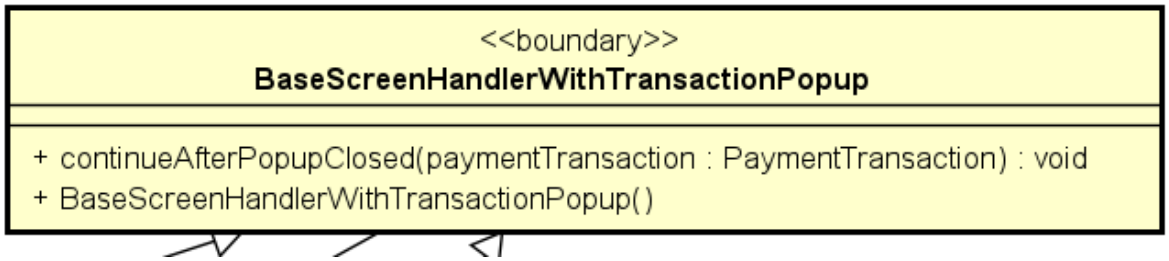
Method

None

State

None

- **Class "BaseScreenHandlerWithTransactionPopup"**



Attribute

None

Operation

#	Name	Return type	Description
1	continueAfterPopupClosed	void	show then closed popup of payment result

Parameter:

- continueAfterPopupClosed
  - paymentTransaction – transaction to be shown result

Exception:

None

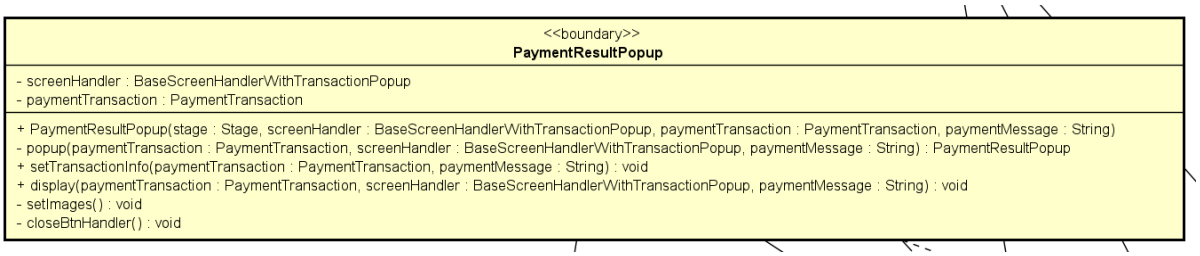
Method

None

State

None

- **Class “PaymentResultPopup”**



Attribute

#	Name	Data type	Default value	Description
1	screenHandler	BaseScreenHandlerWithTransactionPopup	new	handler
2	paymentTransaction	PaymentTransaction	NULL	transaction to be shown

Operation

#	Name	Return type	Description
1	setTransactionInfo	void	Display result
2	display	void	Render result
3	setImage	void	Set image for screen
4	closeBtnHandler	void	Close the popup and move to next screen
5	popup	PaymentResultPopup	Create then return popup

Parameter:

- popup
  - screenHandler
  - paymentTransaction
  - paymentMessage

Exception:

None

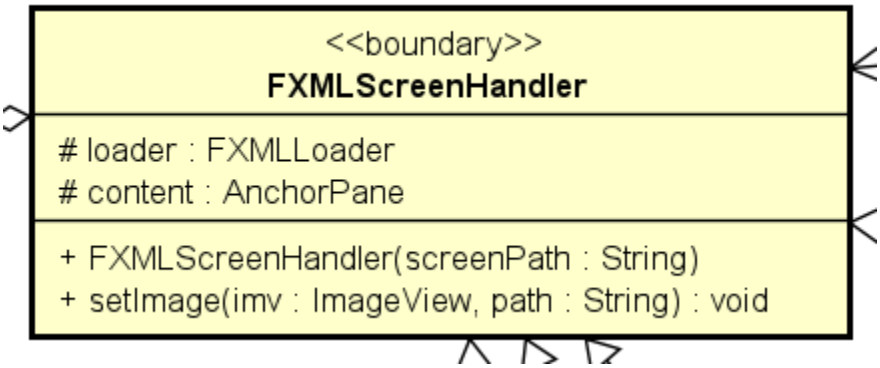
Method

None

State

None

- **Class "FXMLScreenHandler"**



Attribute

#	Name	Data type	Default value	Description
1	loader	FXMLLoader	new	
2	content	AnchorPane	NULL	main app anchor

Operation

#	Name	Return type	Description
1	setImage	void	set image for screen

Parameter:

- setImage
  - imv – image holder to be set
  - path – path to image

Exception:

None

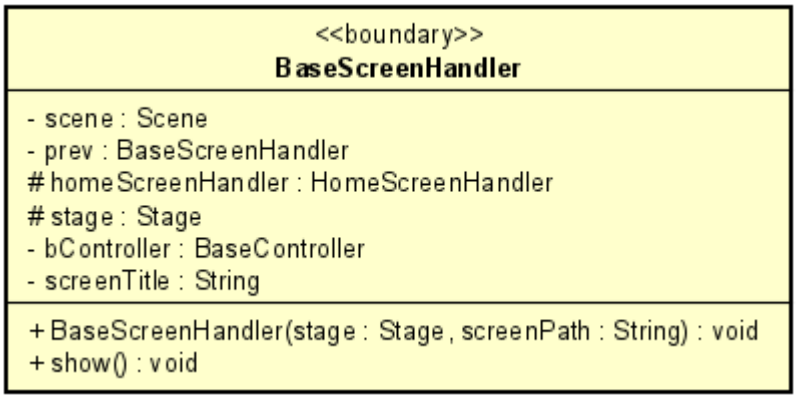
Method

None

State

None

- **Class "BaseScreenHandler"**



Attribute

#	Name	Data type	Default value	Description
1	scene	Scene	NULL	Scene to render
2	prev	BaseScreenHandler	NULL	Previous screen
3	homeScreenHandler	HomeScreenHandler	NULL	Home screen
4	stage	Stage	NULL	JavaFx stage
5	bController	BaseController	NULL	Controller for this screen
6	screenTitle	String	NULL	Title of this screen

Operation

#	Name	Return type	Description
1	show	void	Render the screen

Parameter:

Exception:

None

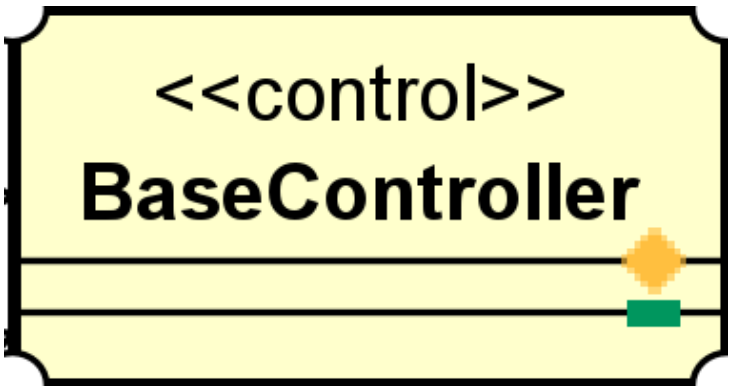
Method

None

State

None

- **Class "BaseController"**



Attribute

None

Operation

None

Parameter:

None

Exception:

None

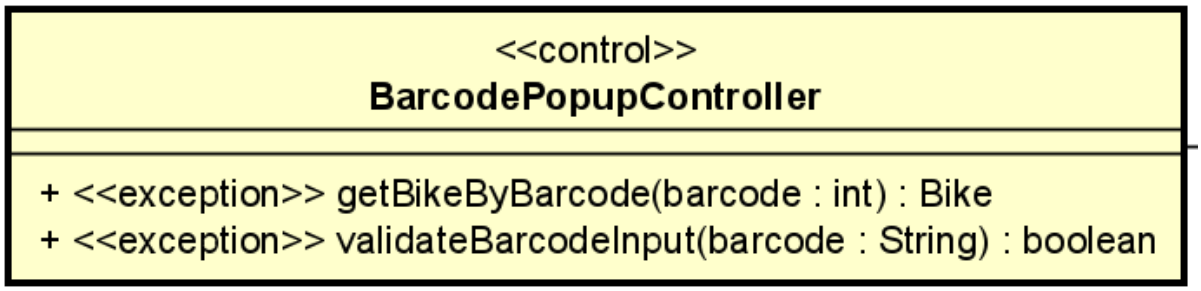
Method

None

State

None

- **Class "BarcodePopupController"**



Attribute

None

Operation

#	Name	Return type	Desciption
1	getBikeByBarcode	Bike	Get bike by using barcode
2	ValidateBarcodeInput	boolean	Checks if barcode is entered or not

Parameter:

- Parameter of getBikeByBarcode()
  - barcode– int
- Parameter of ValidateBarcodeInput()
  - barcode - string

Exception:

- BarcodeNotFoundException
- NullBarcodeException
- InvalidBarcodeException

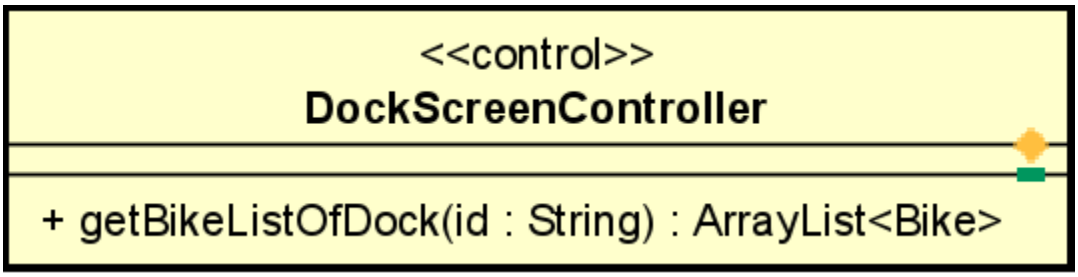
Method

None

State

None

- **Class "DockScreenController"**



Attribute

None

Operation

#	Name	Return type	Desciption
1	getBikeListOfDock	ArrayList<Bike>	Get bike list by using dock id

Parameter:

- Parameter of getBikeListOfDock()
  - Id – dock id

Exception:

None

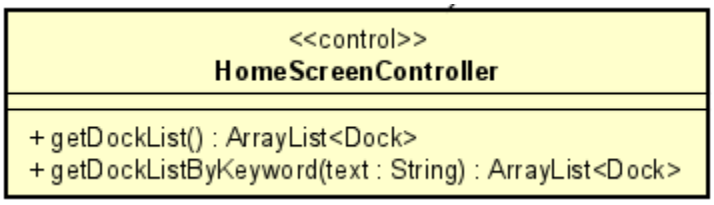
Method

None

State

None

- **Class "HomeScreenController"**



Attribute

None

Operation

#	Name	Return type	Desciption
1	getDockList	ArrayList<Dock>	Get dock list
2	getDockListByKeyword	ArrayList<Dock>	Get dock list by keyword

Parameter:

- Parameter of getDockListByKeyword()
  - Text - keyword

Exception:

None

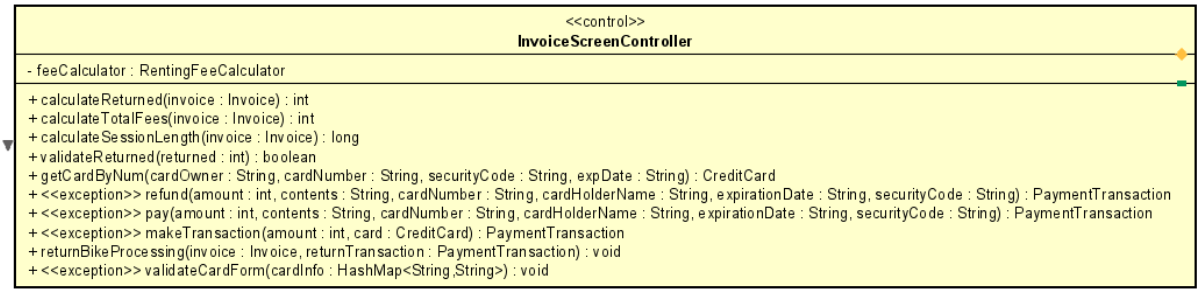
Method

None

State

None

- **Class "InvoiceScreenController"**



Attribute

#	Name	Data type	Default value	Description
1	feeCalculator	RentingFeeCalculator	new RentingFeeBySecondsCalculator	calculate fee

Operation

#	Name	Return type	Desciption
1	calculateReturned	Int	Calculate returned after deducting renting fees
2	calculateTotalFees	Int	Calculate total fees
3	calculateSessionLength	long	Session length
4	refund	PaymentTransaction	Create return transaction
5	validateReturned	boolean	Check returned validity
6	getCardByNum	CreditCard	Get credit card info
7	refund	PaymentTransaction	Setup Return transaction
8	pay	PaymentTransaction	Setup transaction
9	makeTransaction	PaymentTransaction	Proceed transaction
10	returnBikeProcessing	void	Process return bike
11	validateCardForm	void	Check card form validity

Parameter:

- Parameter of calculateReturned()
  - Invoice
- Parameter of calculateTotalFees()
  - Invoice
- Parameter of calculateSessionLength()
  - Invoice
- Amount Parameter of getCardByNum()
  - cardOwner
  - cardNumber
  - securityCode
  - expDate
- Parameter of refund()
  - amount
  - contents
  - cardNumber
  - cardHolderName
  - expirationDate
  - securityCode
- Parameter of pay()
  - amount
  - contents
  - cardNumber
  - cardHolderName
  - expirationDate
  - securityCode
- ExpirationDate Parameter of makeTransaction()
  - amount
  - card
- Parameter of validateReturned()
  - returned
- Parameter of returnBikeProcessing()
  - Invoice
  - returnTransaction
- Parameter of validateCardForm()
  - cardInfo

Exception:

PaymentException

NullCardNumberException

NullCardOwnerException

NullExpDateException

NullSecurityCodeException

CardExpiredException

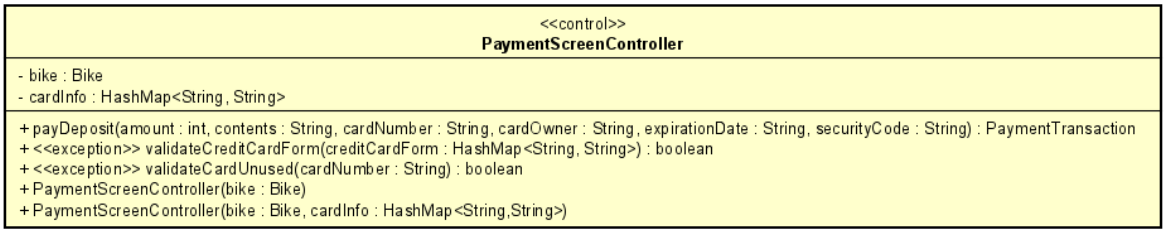
Method

None

State

None

- **Class "PaymentScreenController"**



Attribute

#	Name	Data type	Default value	Description
1	bike	Bike	NULL	bike
2	cardInfo	HashMap<String, String>	NULL	Card info

Operation

#	Name	Return type	Desciption
1	payDeposit	PaymentTransaction	Set rent transaction
2	validateCreditCardForm	boolean	Check creditcard form
3	validateCardUnused	boolean	Check creditcard used or not

Parameter:

- Parameter of payDeposit()
  - amount
  - contents
  - cardNumber
  - cardOwner
  - expirationDate
  - securityCode
- Parameter of validateCreditCardForm()
  - creditCardForm
- Parameter of validateCardUnused()
  - cardNumber

Exception:

None

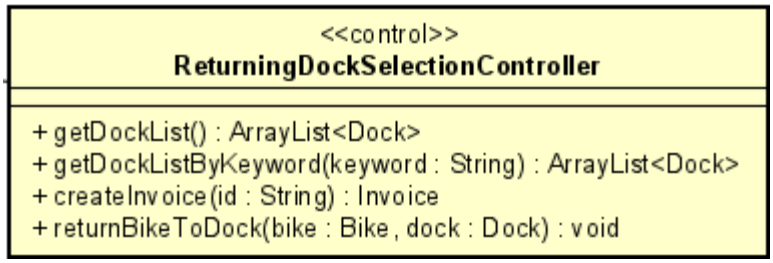
Method

None

State

None

- **Class "ReturningDockSelectionController"**



Attribute

None

Operation

#	Name	Return type	Desciption
1	getDockList	ArrayList<Dock>	Get dock list
2	getDockListByKeyWord	ArrayList<Dock>	Get dock list by keyword
3	createInvoice	Invoice	Create new invoice
4	returnBikeToDock	void	Return bike to the chosen dock

Parameter:

- Parameter of getDockListByKeyWord()
  - keyword
- Parameter of createInvoice()
  - id
- Parameter of returnBikeToDock()
  - bike
  - dock

Exception:

None

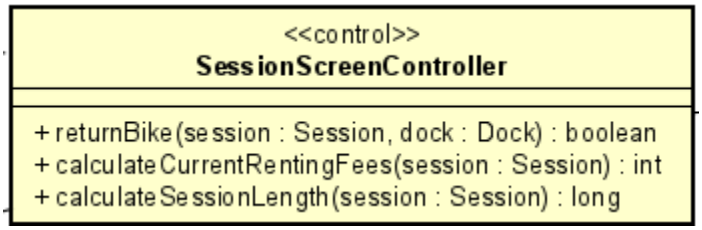
Method

None

State

None

- **Class "SessionScreenController"**



Attribute

None

Operation

#	Name	Return type	Desciption
1	returnBike	boolean	Check if bike is available
2	calculateCurrenRentingFees	int	Calculate current renting fees
3	calculateSessionLength	long	Calculate session length

Parameter:

- Parameter of returnBikeToDock()
  - session
  - dock
- Parameter of returnBikeToDock()
  - session
- Parameter of returnBikeToDock()

- session

Exception:

None

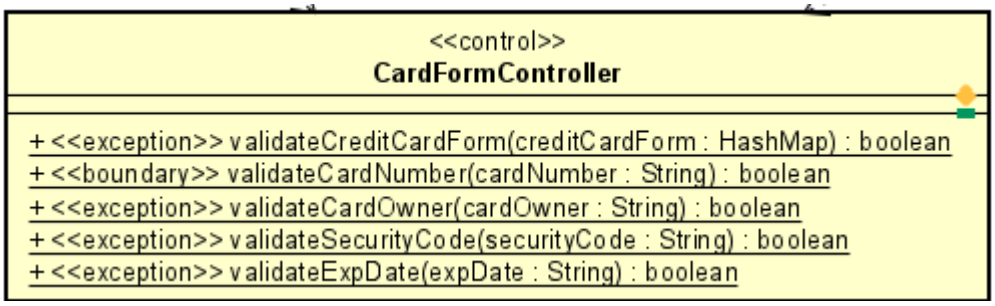
Method

None

State

None

- **Class "CardFormController"**



Attribute

None

Operation

#	Name	Return type	Desciption
1	validateCreditCardForm	boolean	Check if card is valid
2	validateCardNumber	boolean	Validate card number
3	validateCardOwner	boolean	Validate card owner
4	validateSecurityCode	boolean	Validate security code
5	validateExpDate	boolean	Validate exp date

Parameter:

- Parameter of validateCreditCardForm()
  - creditCardForm
- Parameter of validateCardNumber()

- CardNumber

➢ Parameter of validateCardOwner()
- cardOwner

➢ Parameter of validateSecurityCode()
- securityCode

➢ Parameter of validateExpDate()
- expDate

Exception:

None

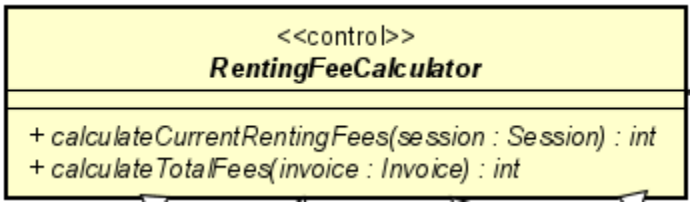
Method

None

State

None

- **Class "RentingFeeCaculator"**



Attribute

None

Operation

#	Name	Return type	Desciption
1	calculateCurrentRentingFees	int	Calculate current renting fees
2	calculateTotalFees	int	Calculate total fees

Parameter:

➢ Parameter of calculateCurrentRentingFees()
- session

- Parameter of calculateTotalFees()
  - invoice

Exception:

None

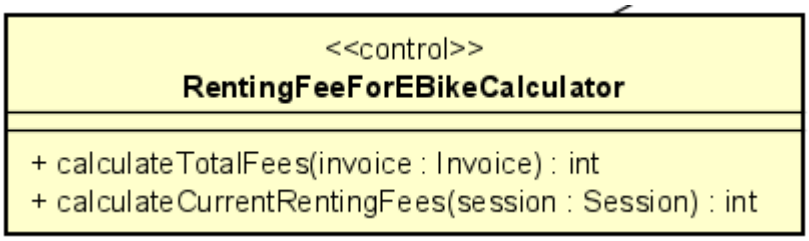
Method

None

State

None

- **Class "RentingFeeForEBikeCalculator"**



Attribute

None

Operation

#	Name	Return type	Desciption
1	calculateCurrentRentingFees	int	Calculate current renting fees
2	calculateTotalFees	int	Calculate total fees

Parameter:

- Parameter of calculateCurrentRentingFees()
  - session
- Parameter of calculateTotalFees()
  - invoice

Exception:

None

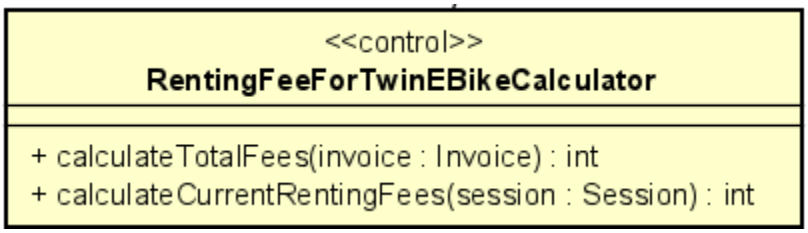
Method

None

State

None

- **Class "RentingFeeForTwinEBikeCalculator"**



Attribute

None

Operation

#	Name	Return type	Desciption
1	calculateCurrentRentingFees	int	Calculate current renting fees
2	calculateTotalFees	int	Calculate total fees

Parameter:

- Parameter of calculateCurrentRentingFees()
  - session
- Parameter of calculateTotalFees()
  - invoice

Exception:

None

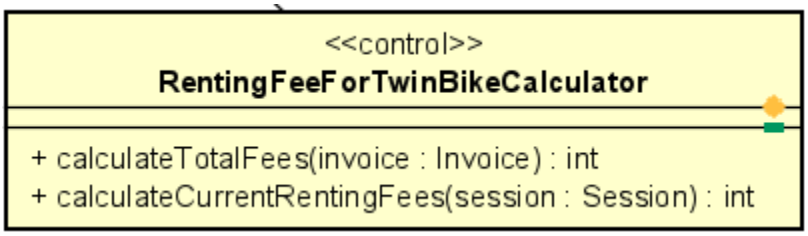
Method

None

State

None

- **Class "RentingFeeForTwinBikeCalculator"**



Attribute

None

Operation

#	Name	Return type	Desciption
1	calculateCurrentRentingFees	int	Calculate current renting fees
2	calculateTotalFees	int	Calculate total fees

Parameter:

- Parameter of calculateCurrentRentingFees()
  - session
- Parameter of calculateTotalFees()
  - invoice

Exception:

None

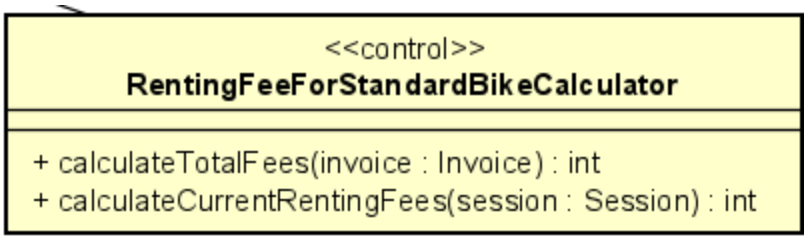
Method

None

State

None

- **Class "RentingFeeForStandardBikeCalculator"**



Attribute

None

Operation

#	Name	Return type	Desciption
1	calculateCurrentRentingFees	int	Calculate current renting fees
2	calculateTotalFees	int	Calculate total fees

Parameter:

- Parameter of calculateCurrentRentingFees()
  - session
- Parameter of calculateTotalFees()
  - invoice

Exception:

None

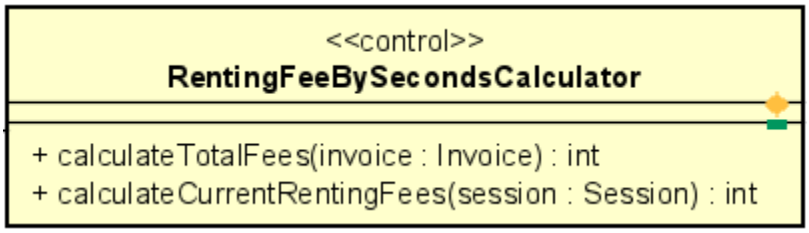
Method

None

State

None

- **Class "RentingFeeBySecondsCalculator"**



Attribute

None

Operation

#	Name	Return type	Desciption
1	calculateCurrentRentingFees	int	Calculate current renting fees
2	calculateTotalFees	int	Calculate total fees

Parameter:

- Parameter of calculateCurrentRentingFees()
  - session
- Parameter of calculateTotalFees()
  - invoice

Exception:

None

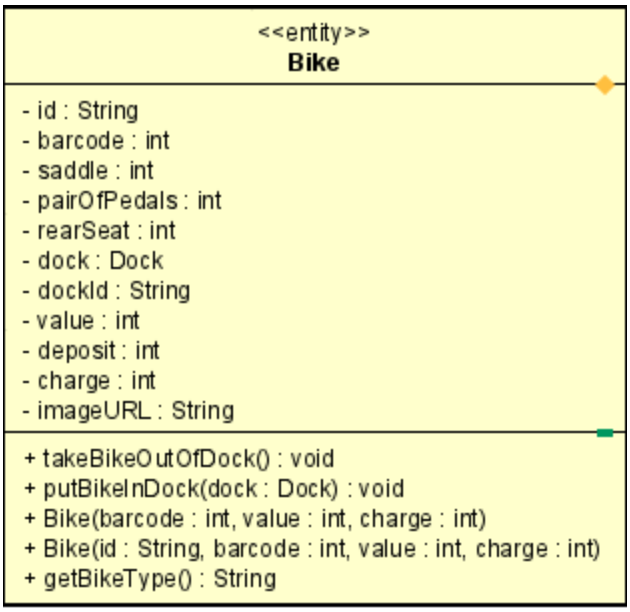
Method

None

State

None

- **Class "Bike"**



Attribute

#	Name	Data type	Default value	Description
1	id	String	NULL	Bike id
2	barcode	int	NULL	barcode
3	saddle	int	NULL	saddle
4	pairOfPedals	int	NULL	Pair of pedals
5	rearSeat	int	NULL	Number of rear seats
6	dock	Dock	NULL	dock
7	dockId	String	NULL	Dock id
8	value	int	NULL	value
9	doposit	int	NULL	deposit
10	charge	int	NULL	charge
11	imageURL	String	NULL	Image URL

Operation

#	Name	Return type	Desciption

1	takeBikeOutOfDock	void	Take bike out of dock
2	putBikeInDock	void	Put bike to a dock
3	getBikeType	String	Get bike type

Parameter:

- putBikeInDock
  - dock - Dock

Exception:

None

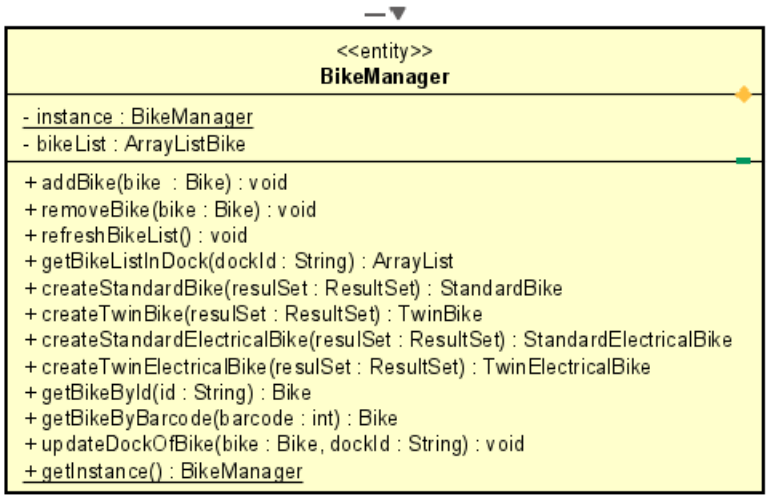
Method

None

State

None

- **Class "BikeManager"**



Attribute

#	Name	Data type	Default value	Description
1	instance	BikeManager	NULL	Bike instance
2	bikeList	ArrayList	NULL	Bike list

Operation

#	Name	Return type	Desciption
1	addBike	void	Add bike
2	removeBike	void	Remove bike
3	refreshBikeList	void	Refresh bike list
4	getBikeListInDock	ArrayList	Get bike list in dock
5	createStandardBike	StandardBike	Create standard bike
6	createTwinBike	TwinBike	Create twin bike
7	createStandardEBike	StandardEBike	Create standard e bike
8	createTwinEBike	TwinEBike	Create twin e bike
9	getBikeById	Bike	Get bike by id
10	getBikeByBarcode	Bike	Get bike by barcode
11	updateDockOfBike	void	Update dock
12	getInstance	BikeManager	Get bike instance

Parameter:

- addBike
  - bike - Bike
- getBikeListInDock
  - dockId - String
- createStandardBike
  - resultSet - ResultSet
- createTwinBike
  - resultSet - ResultSet
- createTwinEBike
  - resultSet - ResultSet

- getBikeById
    - id – String
- getBikeByBarcode
    - barcode - int
- updateDockOfBike
    - dockId – String
    - bike - Bike

Exception:

None

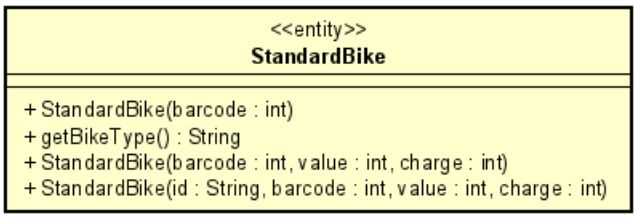
Method

None

State

None

- **Class "StandardBike"**



Attribute

None

Operation

#	Name	Return type	Desciption
1	getBikeType	String	Get bike type

Parameter:

None

Exception:

None

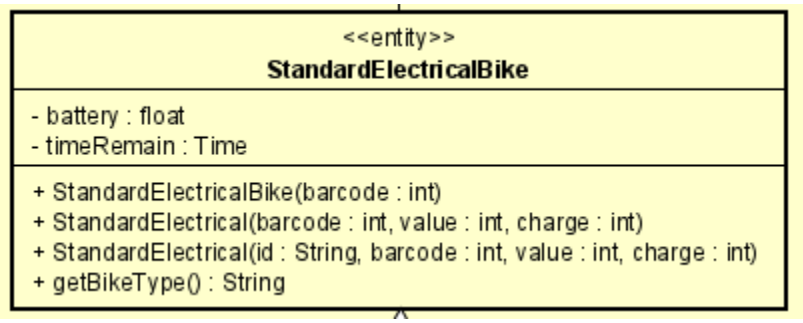
Method

None

State

None

- **Class "StandardElectricalBike"**



Attribute

#	Name	Data type	Default value	Description
1	battery	float	100.0	battery remained
2	timeRemain	Time	NULL	usage time base on battery and bike

Operation

#	Name	Return type	Desciption
1	getBikeType	String	Get bike type

Parameter:

None

Exception:

None

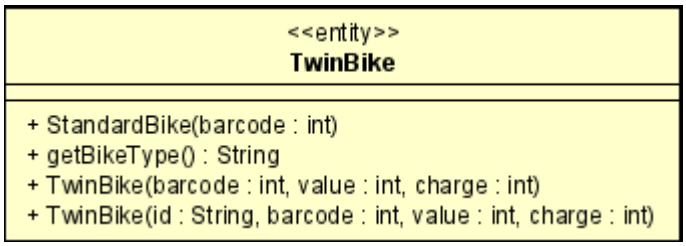
Method

None

State

None

- **Class "TwinBike"**



Attribute

None

Operation

#	Name	Return type	Desciption
1	getBikeType	String	Get bike type

Parameter:

None

Exception:

None

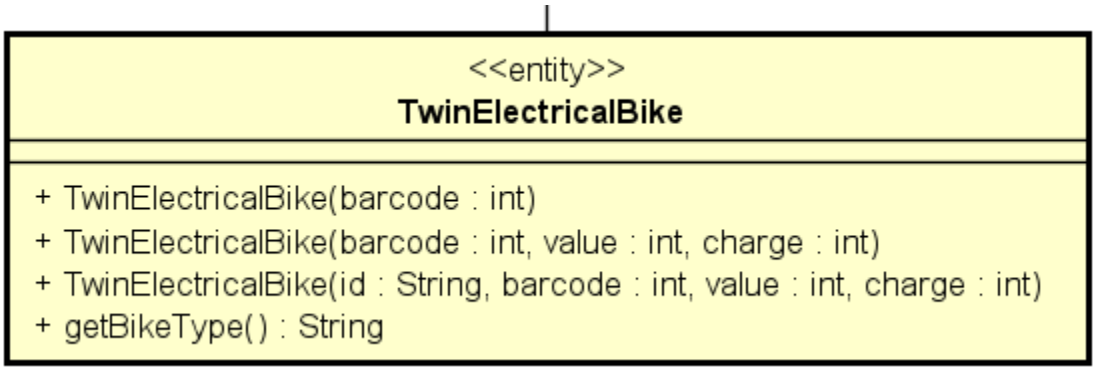
Method

None

State

None

- **Class "TwinElectricalBike"**



Attribute

None

Operation

#	Name	Return type	Desciption
1	getBikeType	String	Get bike type

Parameter:

None

Exception:

None

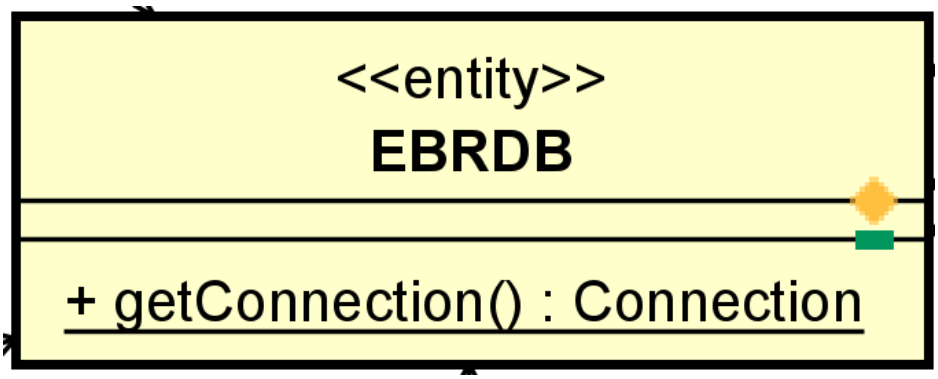
Method

None

State

None

- **Class "EBRDB"**



Attribute

None

Operation

#	Name	Return type	Desciption
1	getConnection	Connect	get a connection to PostgreSQL database

Parameter:

None

Exception:

None

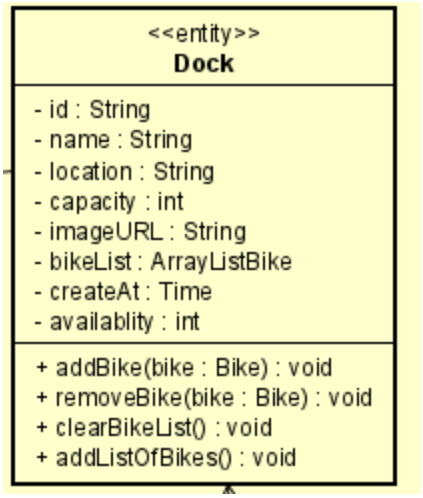
Method

None

State

None

- **Class "Dock"**



Attribute

#	Name	Data type	Default value	Description
1	id	String	NULL	Dock id
2	name	String	NULL	Name of dock
3	location	String	NULL	Dock location
4	capacity	int	NULL	Dock size
5	imageURL	String	NULL	Dock image
6	bikeList	ArrayList<Bike>	NULL	Bike list in dock
7	createAt	Time	NULL	Time Create Dock
8	Availalbity	Int	NULL	Number of availability bike

Operation

#	Name	Return type	Desciption
1	addBike	void	Add bike to dock
2	removeBike	void	Remove bike out of dock
3	clearBikeList	void	Clear bike list in dock
4	addListOfBike	void	Add bike list to dock

Parameter:

- addBike
  - bike – Bike
- removeBike
  - bike – Bike

Exception:

None

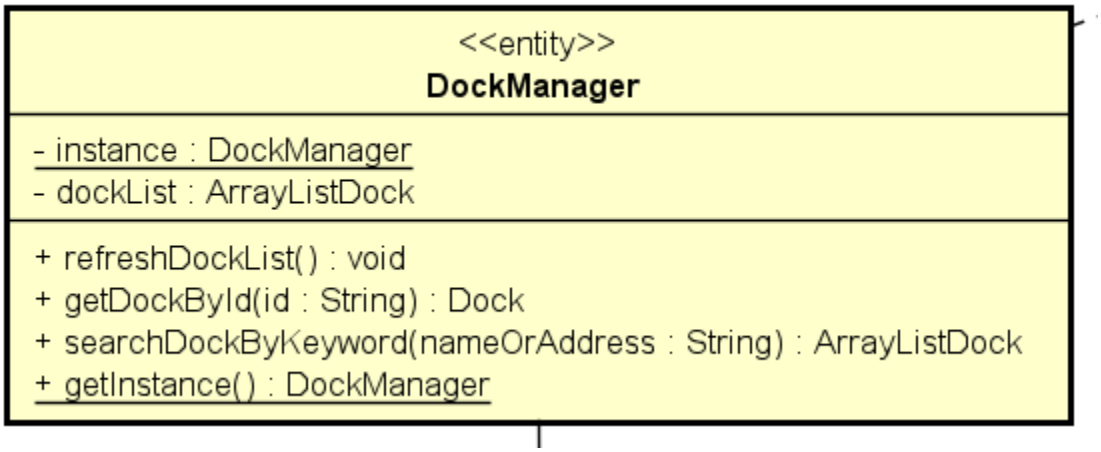
Method

None

State

None

- **Class "DockManager"**



Attribute

#	Name	Data type	Default value	Description
1	instance	DockManager	NULL	Instance of a dock
2	dockList	ArrayList<Dock>	NULL	Dock list

Operation

#	Name	Return type	Desciption
1	refreshDockList	void	Refresh dock list
2	getDockById	Dock	Get dock by id
3	searchDockByKeyword	ArrayList<Dock>	Get docks by keyword
4	getInstance	DockManager	Get dock instance

Parameter:

- getDockById
  - id - String
- searchDockByKeyWord
  - nameOrAddress - String

Exception:

None

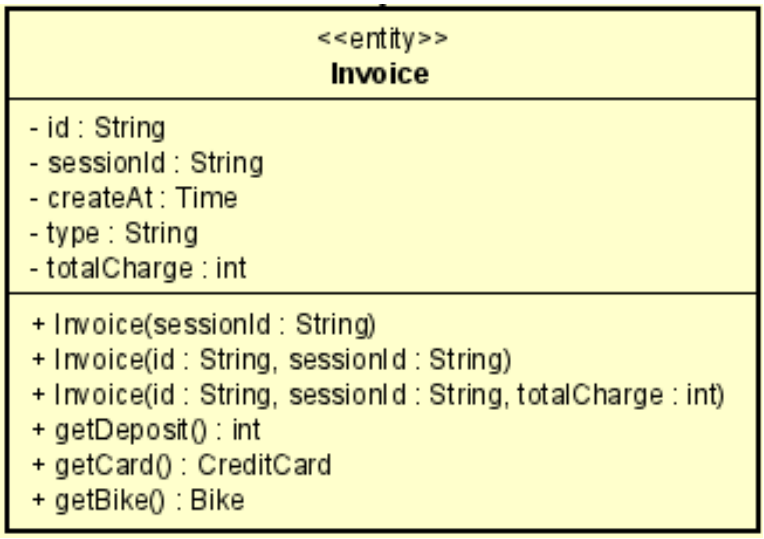
Method

None

State

None

- **Class "Invoice"**



Attribute

#	Name	Data type	Default value	Description
1	id	String	NULL	Invoice id
2	sessionId	String	NULL	Session id
3	createAt	Time	NULL	Create time
4	type	String	NULL	Session corresponds to invoice
5	totalCharge	int	NULL	Total Charge

Operation

#	Name	Return type	Desciption
1	getDeposit	int	Get deposit
2	getCard	CreditCard	Get credit card
3	getBike	Bike	Get bike

Parameter:

None

Exception:

None

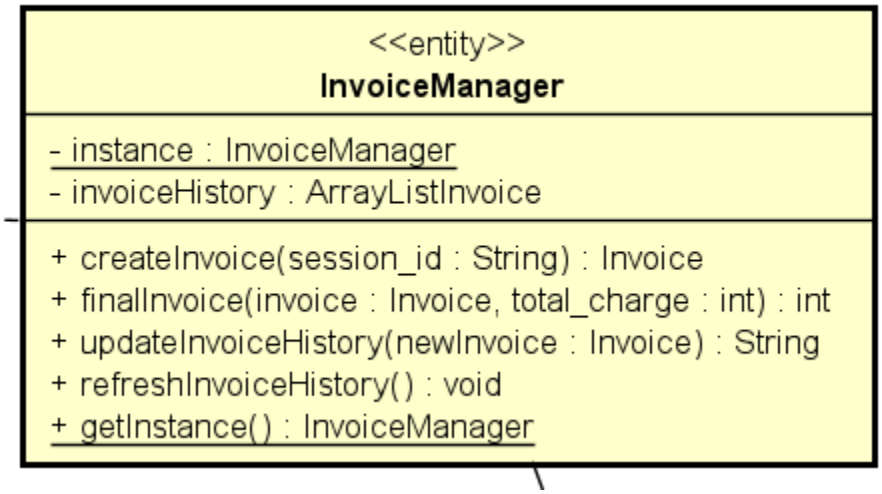
Method

None

State

None

- **Class "InvoiceManager"**



Attribute

#	Name	Data type	Default value	Description
1	instance	DockManager	NULL	Instance of a dock
2	invoiceHistory	ArrayList<Invoice>	NULL	Invoice History

Operation

#	Name	Return type	Desciption
1	createInvoice	Invoice	Create new invoice
2	finalInvoice	int	Complete invoice
3	updateInvoiceHistory	String	Update invoice history
4	refreshInvoiceHistory	void	Refresh invoice history
5	getInstance	InvoiceManager	Get invoice instance

Parameter:

- createInvoice
  - session_id - String
- finallInvoice
  - invoice – Invoice
  - total_charge: int
- updateInvoiceHistory
  - newInvoice - Invoice

Exception:

None

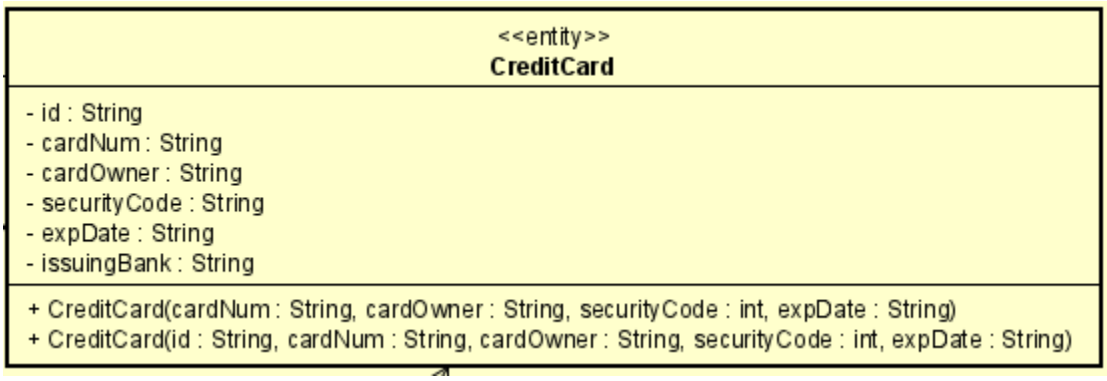
Method

None

State

None

- **Class "CreditCard"**



Attribute

#	Name	Data type	Default value	Description
1	id	String	NULL	Card id
2	cardNum	String	NULL	Card number
3	cardOwner	String	NULL	Card owner's name

4	securityCode	String	NULL	Card security code
5	expDate	String	NULL	Card exp date
6	issuingBank	String	NULL	IssuingBank

Operation

None

Parameter:

None

Exception:

None

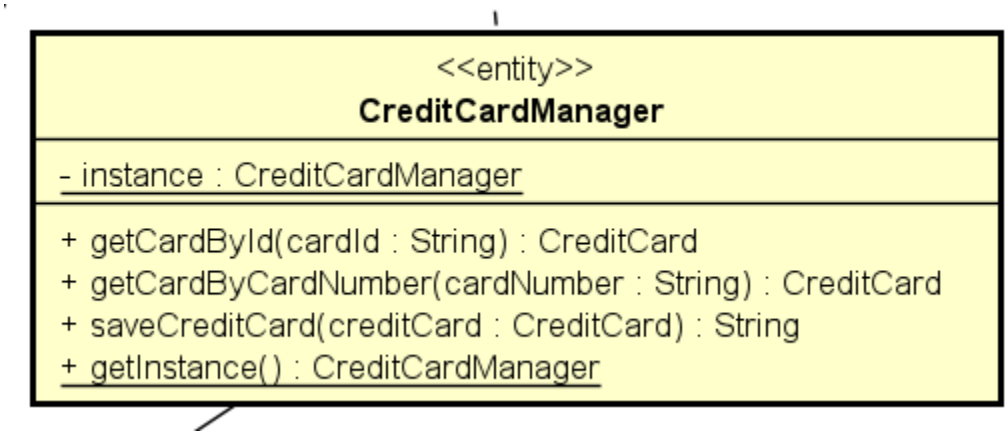
Method

None

State

None

- **Class "CreditCardManager"**



Attribute

#	Name	Data type	Default value	Description
1	instance	CreditCardManager	NULL	Instance of a card

Operation

#	Name	Return type	Desciption
1	getCardById	CreditCard	Get card by id
2	getCardByCardNumber	CreditCard	Get card by card number
3	saveCreditCard	String	Save credit card
4	getInstance	CreditCardManager	Get card instance

Parameter:

- getCardById
  - id - String
- getCardByCardNumber
  - carNumber – String
- saveCreditCard
  - credidCard - CreditCard

Exception:

None

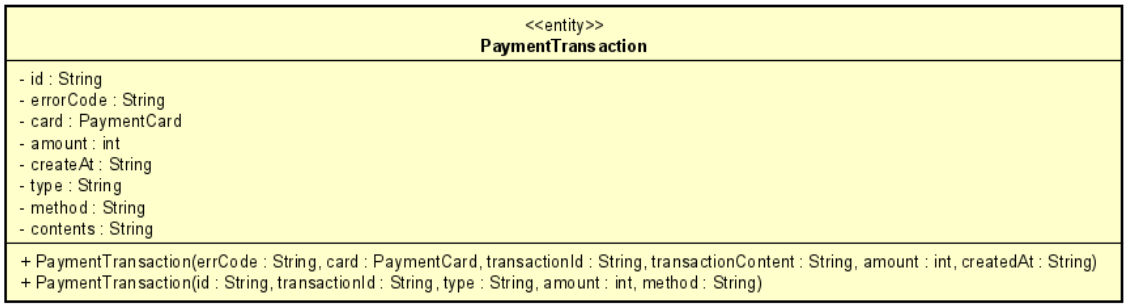
Method

None

State

None

- **Class "PaymentTransaction"**



Attribute

#	Name	Data type	Default value	Description
1	errorCode	String	NULL	Error code
2	card	PaymentCard	NULL	Credit card
3	transactionId	String	NULL	Transaction id
4	transactionContent	String	NULL	Contents of transaction
5	amount	int	NULL	Transaction amount
6	createAt	String	NULL	Transaction create time
7	id	String	NULL	PaymentTransaction id
8	type	String	NULL	Transaction type
9	method	String	NULL	Transaction method

Operation

None

Parameter:

None

Exception:

None

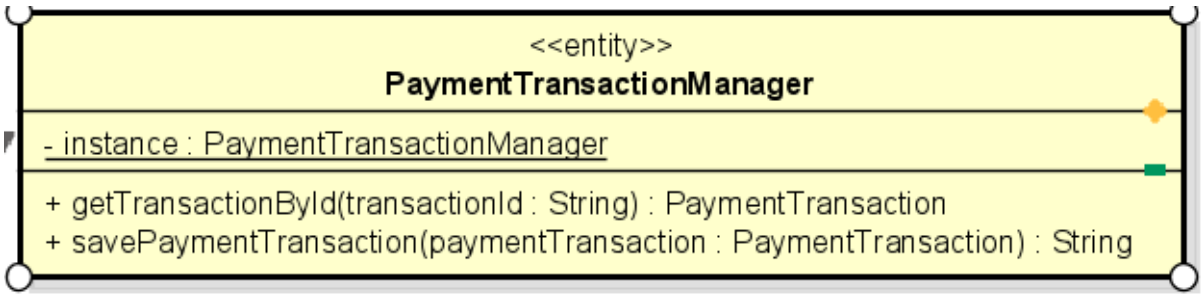
Method

None

State

None

- **Class "PaymentTransactionManager"**



Attribute

#	Name	Data type	Default value	Description
1	instance	PaymentTransaction	NULL	Instance of transaction

Operation

#	Name	Return type	Desciption
1	getTransactionById	PaymentTransaction	Get transaction by Id
2	savePaymentTransaction	String	Save payment transaction

Parameter:

- getTransactionById
  - transactionId - String
- savePaymentTransaction
  - paymentTransaction - Paymentransaction

Exception:

None

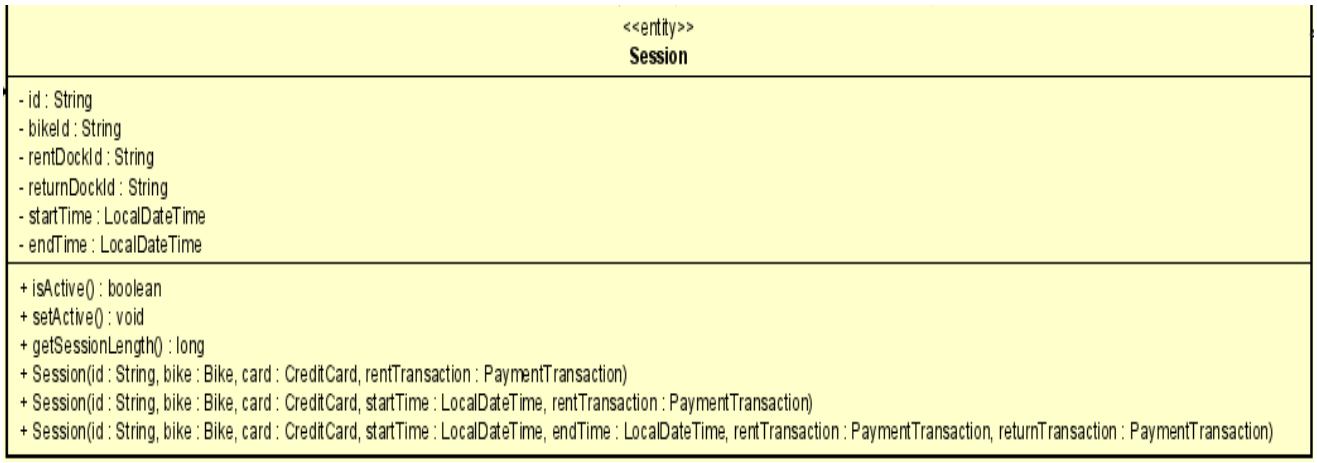
Method

None

State

None

- **Class "Session"**



Attribute

#	Name	Data type	Default value	Description
1	id	String	NULL	Session id
2	bikeId	String	NULL	Bike Id
3	rentDockId	String	NULL	Dock rented Id
4	returnDockId	String	NULL	Dock Return Id
5	endTime	LocalDateTime	NULL	End time
6	startTime	LocalDateTime	NULL	Start time

Operation

#	Name	Return type	Desciption
1	isActive	boolean	Session is active or not
2	setActive	void	Set session active
3	getSessionLength	long	Get session length

Parameter:

None

Exception:

None

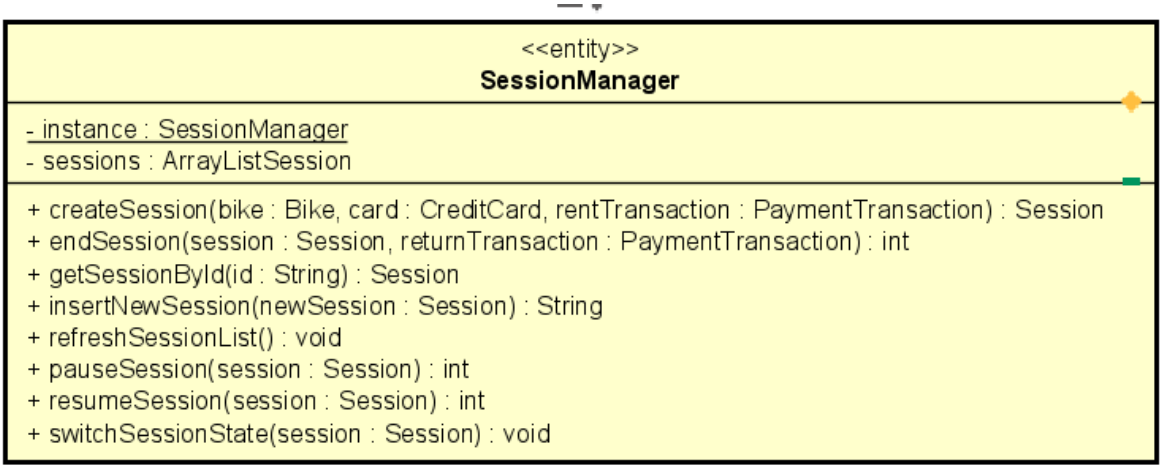
Method

None

State

None

- **Class "SessionManager"**



Attribute

#	Name	Data type	Default value	Description
1	instance	SessionManager	NULL	Session instance
2	sessions	ArrayList<Session>	NULL	Session list

Operation

#	Name	Return type	Desciption
1	createSession	Session	Create a session
2	endSession	int	Complete session
3	getSessionById	Session	Get session by id
4	insertNewSession	String	Insert new session to db
5	refreshSessionList	void	Refresh session list
6	pauseSession	int	Pause session
7	resumeSession	int	Resume session
8	switchSessionState	void	Switch session state

Parameter:

- createSession
  - bike - Bike
  - card – CreditCard
  - rentTransaction - PaymentTransaction
- endSession
  - session – Session
  - returnTransaction - PaymentTransaction
- getSessionById
  - id - String
- insertNewSession
  - newSession – Session
- pauseSession
  - session – Session
- resumeSession

    - session – Session
- switchSessionState
    - session – Session

Exception:

None

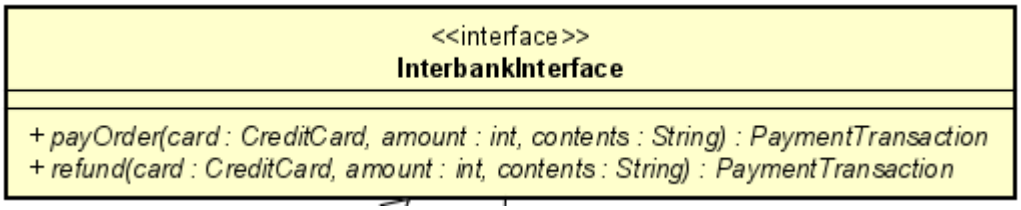
Method

None

State

None

- **Class "InterbankInterface"**



Attribute

None

Operation

#	Name	Return type	Description
1	payOrder	PaymentTransaction	Pay order
2	refund	PaymentTransaction	Refund money

Parameter:

- payOrder
    - creditCard
    - amount
    - contents
- refund
    - card

- amount
- contents

Exception:

None

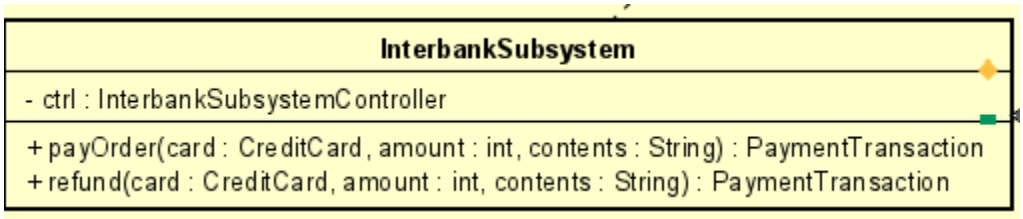
Method

None

State

None

- **Class "InterbankSubsystem"**



Attribute

#	Name	Data type	Default value	Description
1	ctrl	InterbankSubsystemController	NULL	Controller payment transaction

Operation

#	Name	Return type	Description
1	payOrder	PaymentTransaction	Pay order
2	refund	PaymentTransaction	Refund money

Parameter:

- payOrder
  - card
  - amount
  - contents
- refund
  - card
  - amount
  - contents

Exception:

None

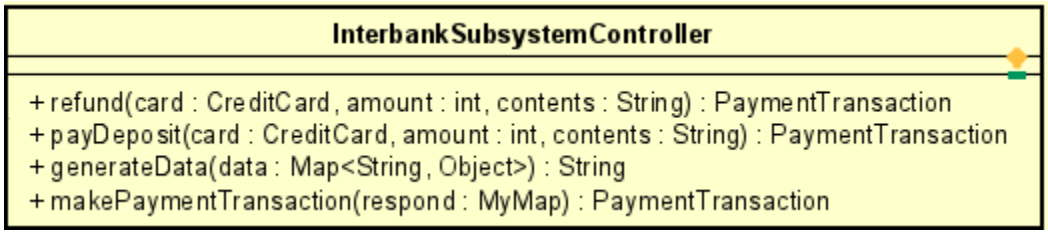
Method

None

State

None

- **Class "InterbankSubsystemController"**



Attribute

None

Operation

#	Name	Return type	Description
1	refund	PaymentTransaction	Refund money
2	payDeposit	PaymentTransaction	Pay deposit money
3	generateData	String	Generate data
4	makePaymentTransaction	PaymentTransaction	Make payment transaction

Parameter:

- payDeposit
  - creditCard
  - amount
  - contents
- refund
  - card
  - amount
  - contents
- generateData
  - data
- makePaymentTransaction
  - respond

Exception:

None

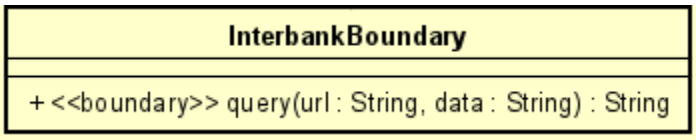
Method

None

State

None

- **Class "InterbankBoundary"**



Attribute

None

Operation

#	Name	Return type	Description
1	query	String	Query url and data with boundary bank

Parameter:

- url
- data

Exception:

None

Method

None

State

None

5 Design Considerations

5.1 Goals and Guidelines

Goals:

- Thiết kế cơ sở dữ liệu tập trung vào tốc độ lưu trữ tối ưu. Do các yêu cầu phi chức năng của phần mềm đòi hỏi khả năng đáp ứng nhanh, vì vậy, lượng dữ liệu không quá lớn.

Guidelines:

- The naming convention will follow the Java naming convention.

5.2 Architectural Strategies

- Design architecture: MVC
- Design principles: SOLID principles
- Database design: PostgreSQL
- Framework: JavaFX.

5.3 Coupling and Cohesion

- Coupling:
  - Không có coupling mức độ Content, Common, Control
  - Hầu hết các package đạt Data Coupling.
- Cohesion:
  - Không có Cohesion mức độ Temporal, Procedural và Communicational.
  - Class Utils bị vi phạm Concidental Cohesion nhưng đây là class hỗ trợ cho nên không phải là vấn đề.

5.4 Design Principles

- Single Responsibility

  Mỗi lớp được thiết kế tuân theo nguyên tắc này. Chỉ có ngoại lệ là lớp Utils vì nó chứa phương thức cho cả bảo mật và định dạng. Tuy nhiên, như đã giải thích trong

phần kết hợp, tất cả các phương thức này là toàn cục, vì vậy việc chia nó thành 2 lớp tiện ích không ảnh hưởng nhiều.

- Open / Closed

  Sử dụng các class cha để tổng quát hóa, ví dụ khi thêm cách tính phí thuê xe, thêm loại xe, thêm phương thức thanh toán

- Liskov Substitution Principle

  Tất cả các lớp cha có thể được thay thế bằng các lớp con của nó. Thiết kế có một hệ thống phân cấp kế thừa rõ ràng và hợp lý.

- Interface Segregation Principle

  InterbankSubsytem Interface đã được thiết kế rất đơn giản và chỉ chứa 2 phương thức cần thiết để thanh toán và hoàn tiền.

- Dependency Inversion Principle

  Các mô-đun cấp cao hơn không phụ thuộc vào các mô-đun cấp thấp hơn. Tất cả các hệ thống con đều có giao diện riêng để người khác sử dụng.