

TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI

Viện Công nghệ thông tin và Truyền thông

Tài liệu mô tả thiết kế phần mềm

(Software Design Description)

Phiên bản 1

THIẾT KẾ VÀ XÂY DỰNG PHẦN MỀM CHO HỆ THỐNG EcobikeRental

Môn: Thiết kế và xây dựng phần mềm

Nhóm 18

Họ tên sinh viên	MSSV
Trần Văn Trí	20173410
Đỗ Viết Trí	20173412
Nguyễn Ngọc Trinh	20173413
Nguyễn Mạnh Trường	20177022

Giảng viên hướng dẫn: TS.Nguyễn Thị Thu Trang

Hà Nội, ngày ... tháng ... năm ...

Mục lục

1	Giới thiệu.....	3
1.1	Mục đích.....	3
1.2	Phạm vi.....	3
1.3	Từ điển thuật ngữ.....	3
1.4	Tham khảo.....	3
2	Thiết kế kiến trúc.....	3
2.1	Lựa chọn kiến trúc phần mềm.....	3
2.2	Thiết kế tổng quan.....	5
2.3	Thiết kế chi tiết gói.....	6
2.3.1	Gói application.....	6
2.3.2	Gói Model.....	7
2.3.3	Gói controller.....	8
2.3.4	Gói views.....	9
2.4	Biểu đồ tương tác.....	10
2.4.1	Biểu đồ tương tác cho UC001 – Xem thông tin bãi xe.....	10
2.4.2	Biểu đồ tương tác cho UC002 – Xem thông tin chi tiết xe.....	10
2.4.3	Biểu đồ tương tác cho UC003 - Thuê xe.....	12
2.4.4	Biểu đồ tương tác cho UC004 – Trả xe.....	12
3	Thiết kế giao diện.....	13
3.1	Giao diện với thiết bị phần cứng.....	13
3.2	Giao diện với phần mềm khác.....	13
3.3	Giao diện người dùng.....	15
3.3.1	Biểu đồ dịch chuyển màn hình.....	15
3.3.2	Thiết kế giao diện.....	15
4	Thiết kế lớp.....	23
4.1	Biểu đồ lớp thiết kế.....	23
4.2	Thiết kế lớp chi tiết.....	23
4.2.1	Thiết kế chi tiết lớp Contants.....	23
4.2.2	Thiết kế lớp InterbankService.....	24
4.2.3	Thiết kế lớp Bike.....	24
4.2.4	Thiết kế chi tiết lớp Station.....	25
4.2.5	Thiết kế chi tiết lớp Card.....	25
4.2.6	Thiết kế chi tiết lớp Customer.....	26
4.2.7	Thiết kế chi tiết lớp Rent.....	26

4.2.8	Thiết kế chi tiết lớp HomeController	26
4.2.9	Thiết kế chi tiết lớp ListStationController	27
4.2.10	Thiết kế chi tiết lớp PaymentFormController	27
4.2.11	Thiết kế chi tiết lớp RentBikeController	27
4.2.12	Thiết kế chi tiết lớp ReturnBikeController	27
4.2.13	Thiết kế chi tiết lớp ViewBikeController	27
4.2.14	Thiết kế chi tiết lớp ViewStationController	28
5	Thiết kế mô hình dữ liệu	28
5.1	Mô hình dữ liệu mức khái niệm	28
5.2	Mô hình dữ liệu mức logic	28
5.3	Thiết kế chi tiết	29
5.3.1	Bảng Station(Bãi xe)	29
5.3.2	Bảng Bike (Xe)	30
5.3.3	Bảng Customer (Khách hàng)	30
5.3.4	Bảng Card (Tài khoản thẻ)	30
5.3.5	Bảng Transaction (Thông tin giao dịch)	31
5.3.6	Bảng Rent (Thuê xe)	31

1 Giới thiệu

1.1 Mục đích

Tài liệu này đưa ra mô tả chi tiết cho người dùng về thiết kế kiến trúc, thiết kế giao diện và thiết kế lớp cho từng chức năng của hệ thống, cũng như việc thiết kế cơ sở dữ liệu của cả hệ thống thuê xe EcobikeRental. Từ đó các bên liên quan sẽ có cái nhìn rõ ràng hơn về phần mềm cần xây dựng.

Tài liệu dành cho các bên liên quan (stakeholder) và các nhà phát triển phần mềm.

1.2 Phạm vi

Hệ thống sử dụng mô hình MVC để cài đặt chương trình. Hệ thống có tương tác với ngân hàng để xử lý giao dịch thanh toán hoặc hoàn tiền.

1.3 Từ điển thuật ngữ

STT	Thuật ngữ	Giải thích
1	MVC	Mô hình MVC viết tắt của Model-View-Controller

1.4 Tham khảo

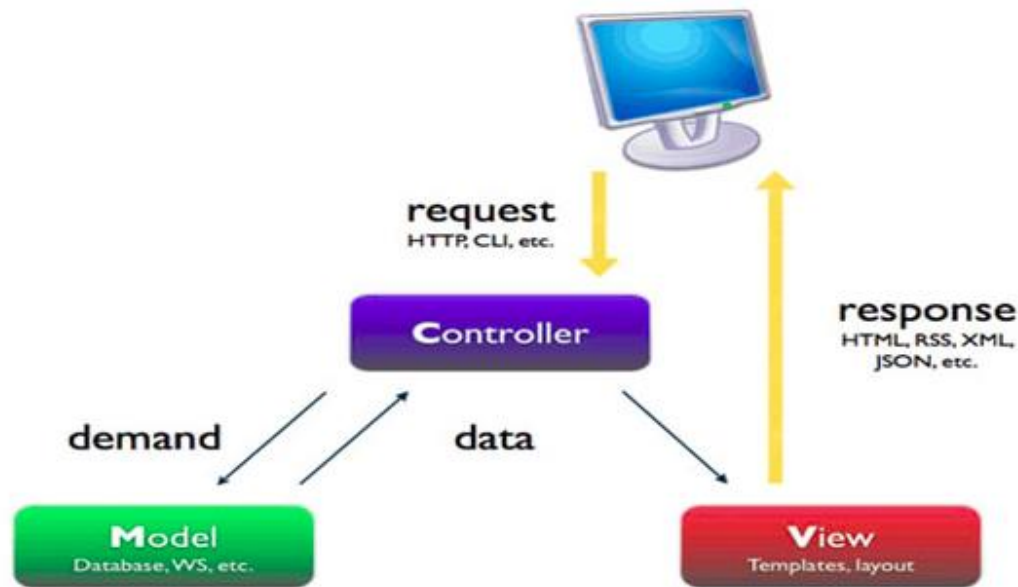
Tài liệu tham khảo được sử dụng trong tài liệu này

+ Tài liệu đặt tả yêu cầu phần mềm SRS

2 Thiết kế kiến trúc

2.1 Lựa chọn kiến trúc phần mềm

Hệ thống xây dựng theo kiến trúc mô hình MVC



- Các thành phần trong kiến trúc:

+ Model: Đây là thành phần chức tất cả các nghiệp vụ logic, phương thức xử lý, truy xuất database, đối tượng mô tả dữ liệu như các class, hàm xử lý,

+ View: Đảm nhận việc hiển thị thông tin, tương tác với người dùng, nơi chứa tất cả các đối tượng GUI như textbox, images,... Hiểu một cách đơn giản, nó là tập hợp các form hoặc các file HTML

+ Controller: Giữ nhiệm vụ nhận điều hướng các yêu cầu từ người dùng và gọi đúng những phương thức xử lý chúng,... Chẳng hạn thành phần này sẽ nhận request từ url và form nào để thao tác trực tiếp với model.

- Luồng hoạt động trong MVC

Khi có một yêu cầu từ phía client gửi đến server, Bộ phận controller có nhiệm vụ nhận yêu cầu, xử lý yêu cầu đó. Và nếu cần, nó sẽ gọi đến phần model, vốn là bộ phận làm việc với Database..

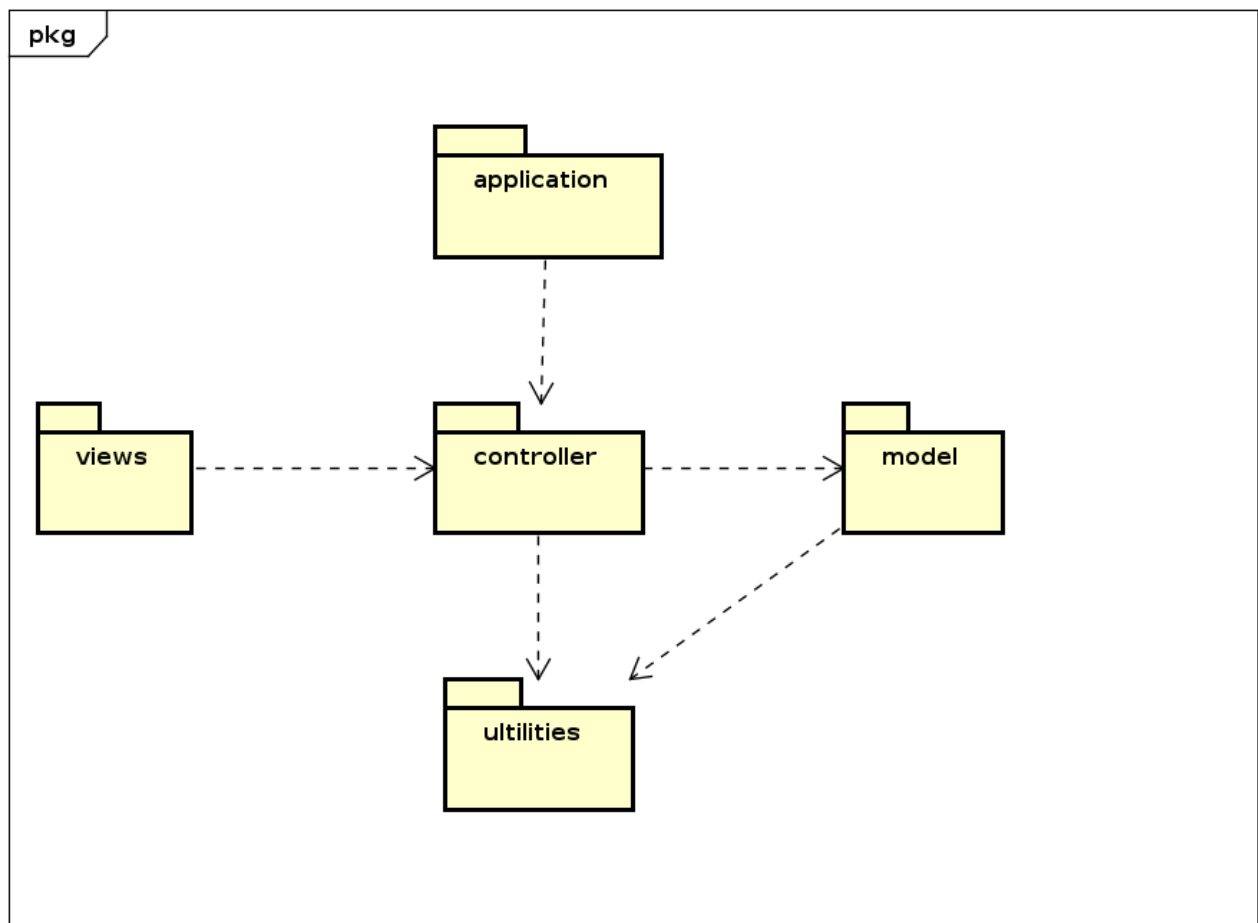
Sau khi xử lý xong, toàn bộ kết quả được đẩy về phần View. Tại View, sẽ gen ra mã Html tạo nên giao diện, và trả toàn bộ html về trình duyệt để hiển thị.

- Ưu điểm và nhược điểm của MVC

+ Ưu điểm: Thể hiện tính chuyên nghiệp trong lập trình, phân tích thiết kế. Do được chia thành các thành phần độc lập nên giúp phát triển ứng dụng nhanh, đơn giản, dễ nâng cấp, bảo trì,...

+ Nhược điểm: Đối với dự án nhỏ việc áp dụng mô hình MVC gây công kênh, tốn thời gian trong quá trình phát triển. Tốn thời gian trung chuyển dữ liệu của các thành phần.

2.2 Thiết kế tổng quan

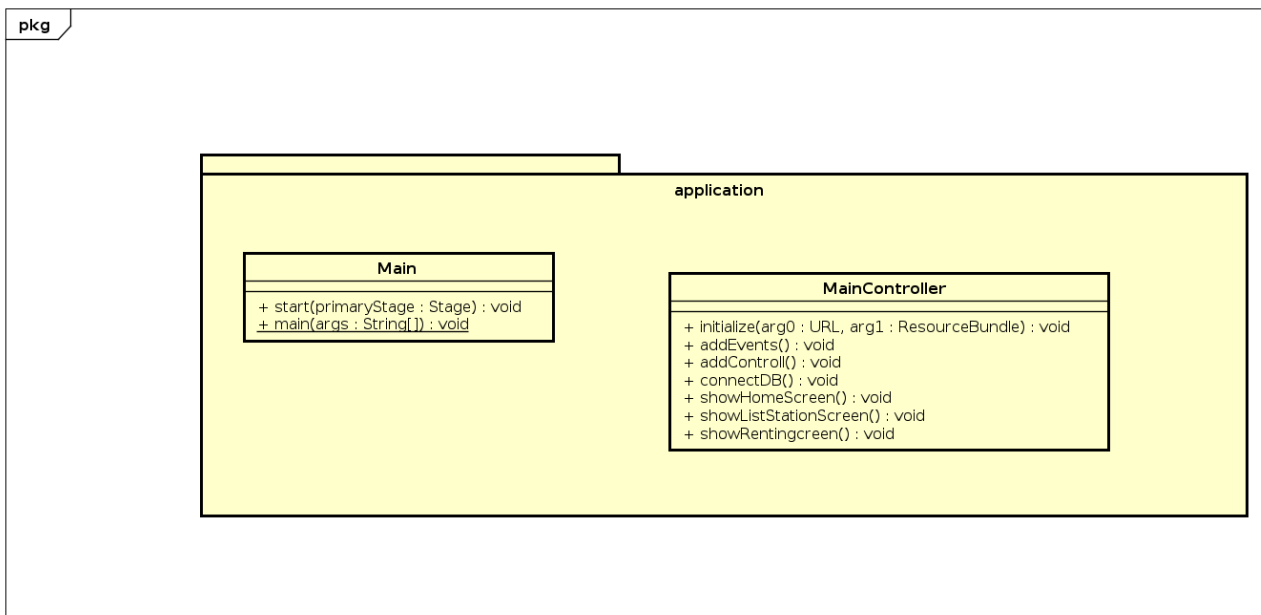


Hình 1 biểu đồ phụ thuộc gói.

- Gói application:
 - Chứa tất cả các lớp main
- Gói views
 - Chứa toàn bộ lớp giao diện của chương trình
- Gói controllers
 - Chức toàn bộ lớp điều khiển của chương trình
- Gói model
 - Chứa toàn bộ lớp thực thể của chương trình, tương tác với cơ sở dữ liệu
- Gói utilites: Chứa 2 lớp InterbankService và gói Contants
 - Lớp InterbankService có nhiệm vụ gọi API và xử lý giao dịch với ngân hàng
 - Lớp Contants định nghĩa dữ liệu static

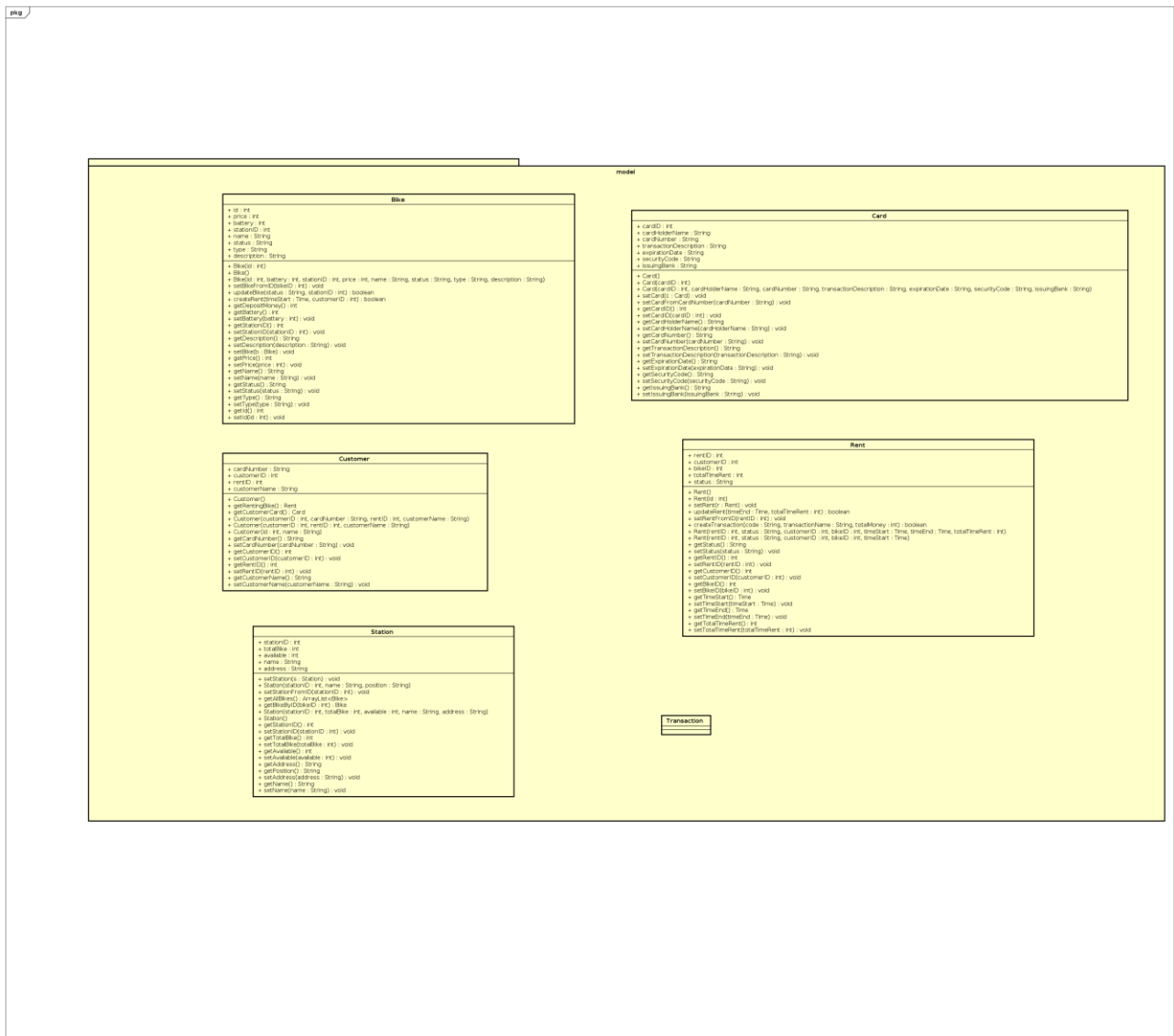
2.3 Thiết kế chi tiết gói

2.3.1 Gói application



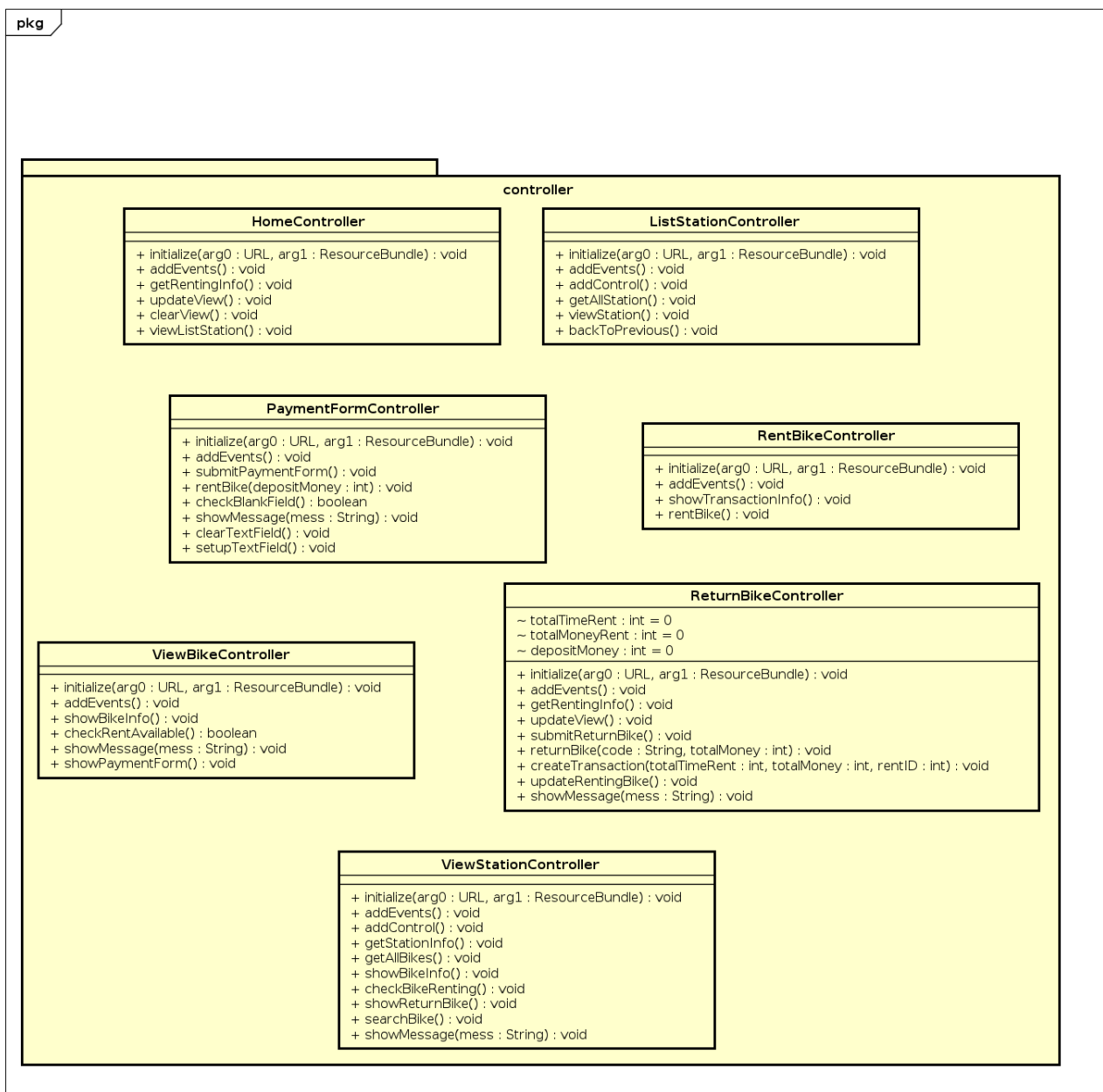
Hình 1 Ví dụ thiết kế gói.

2.3.2 Gói Model



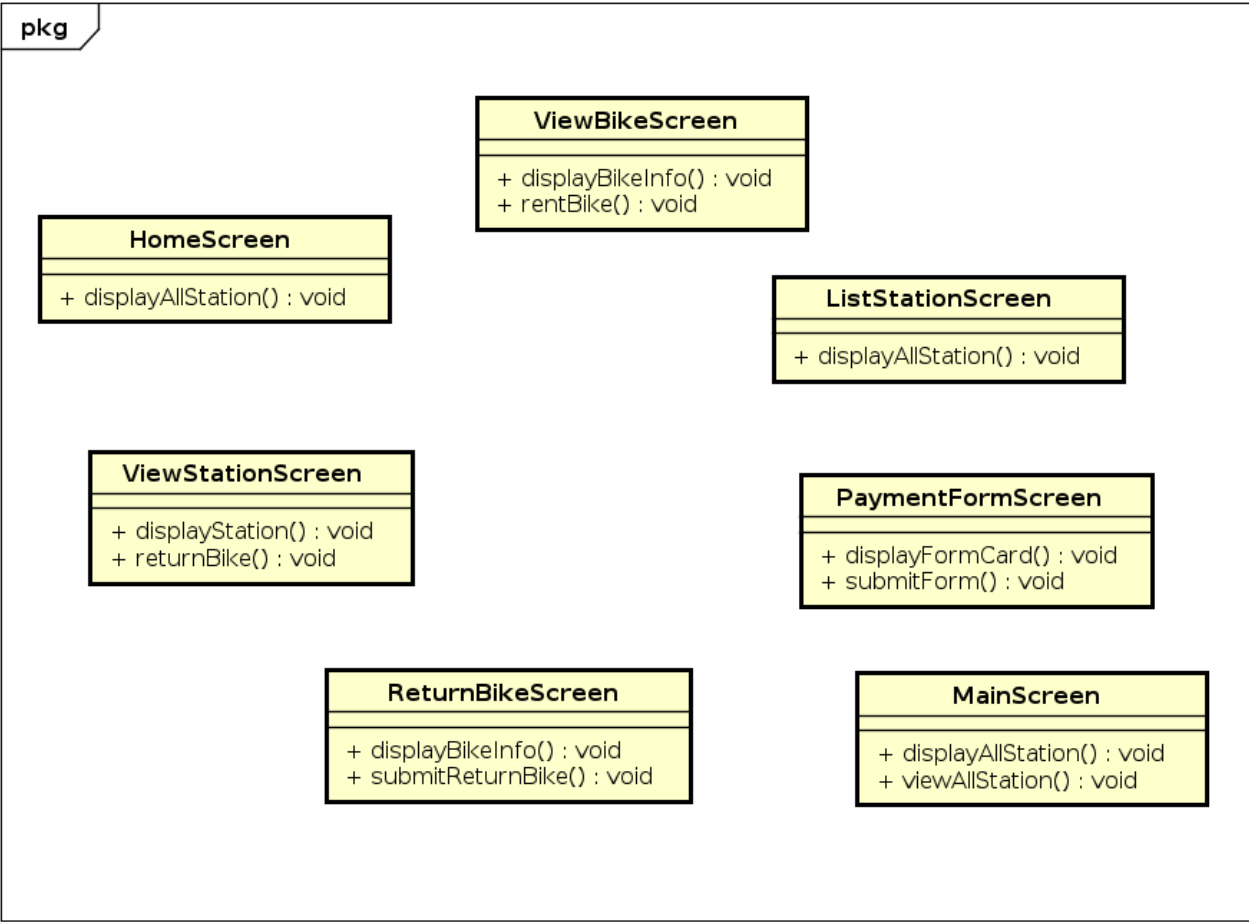
Hình 2: Gói model

2.3.3 Gói controller



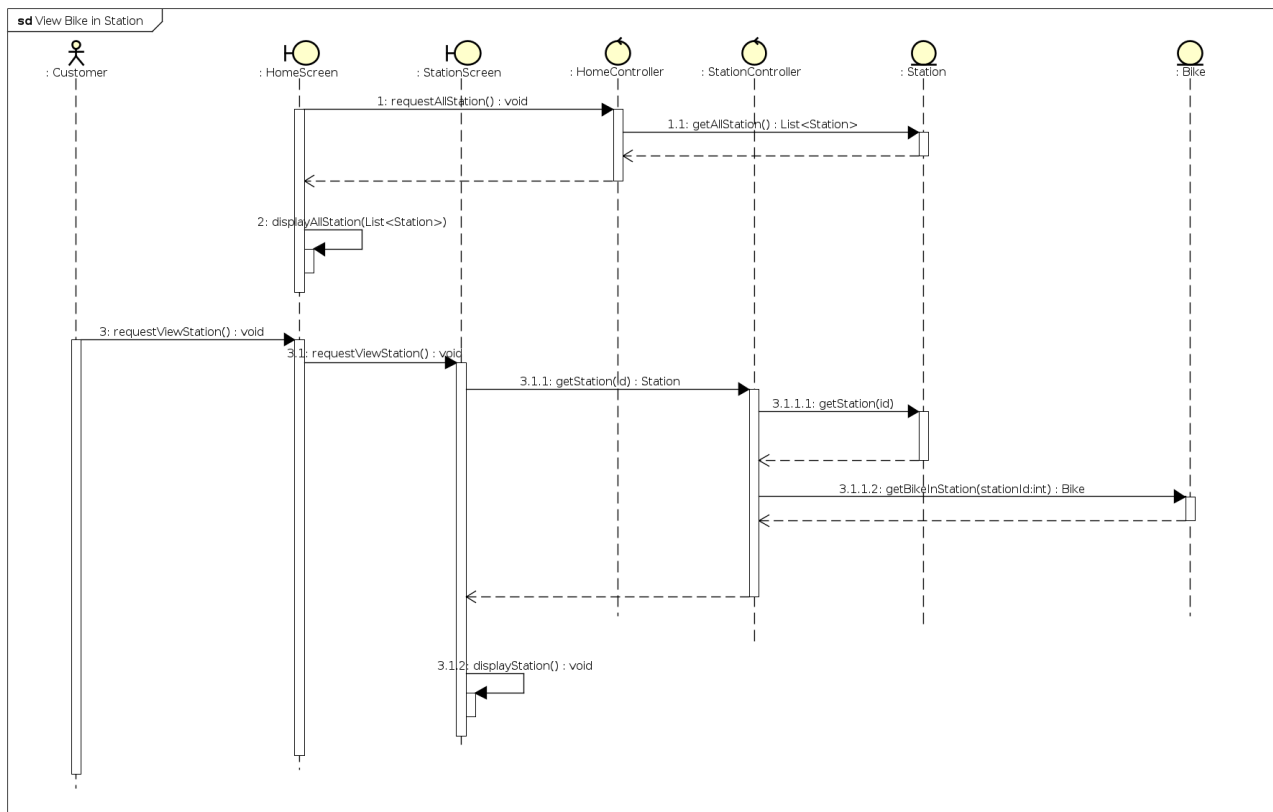
Hình 3: Gói controller

2.3.4 Gói views

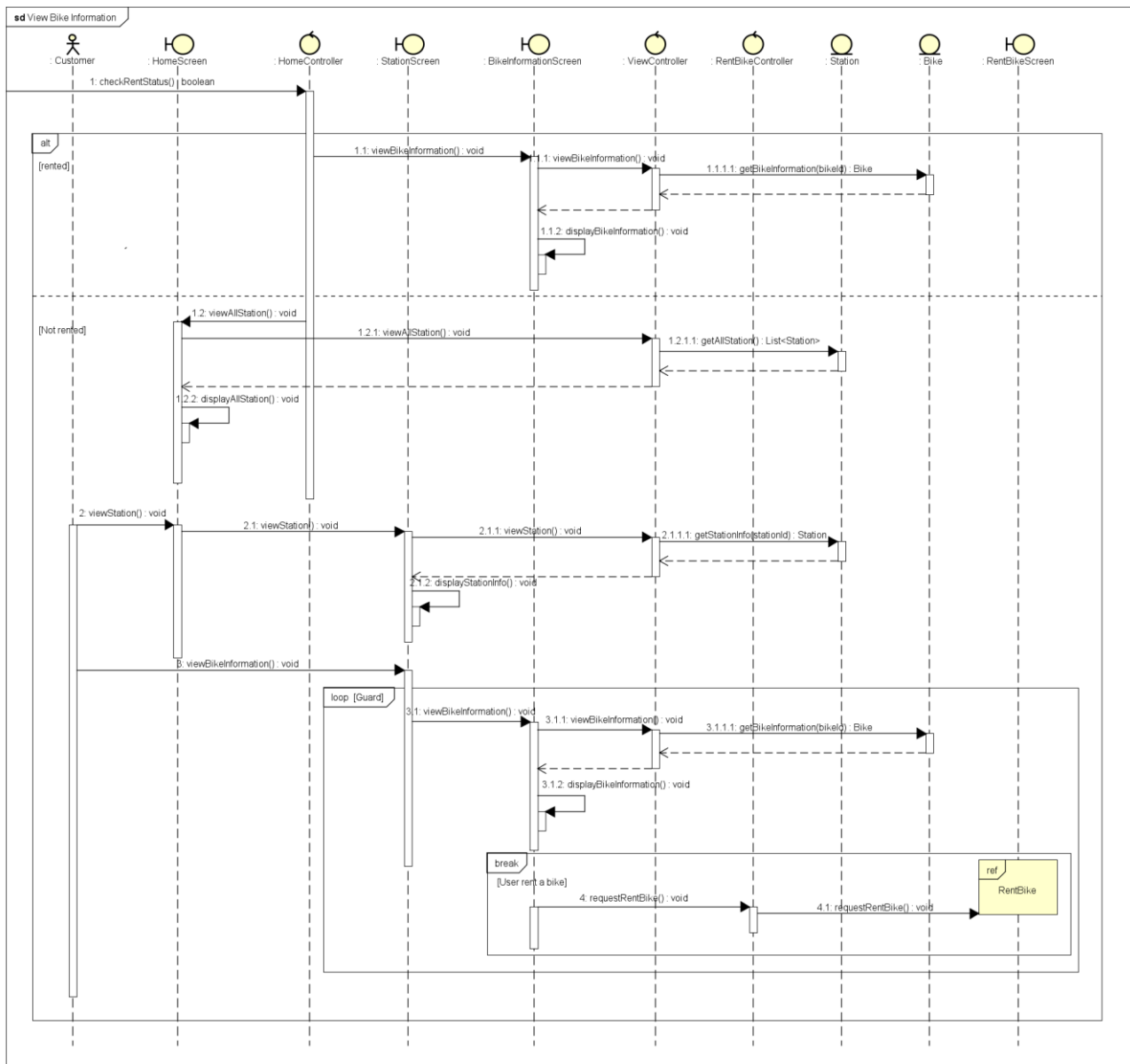


2.4 Biểu đồ tương tác

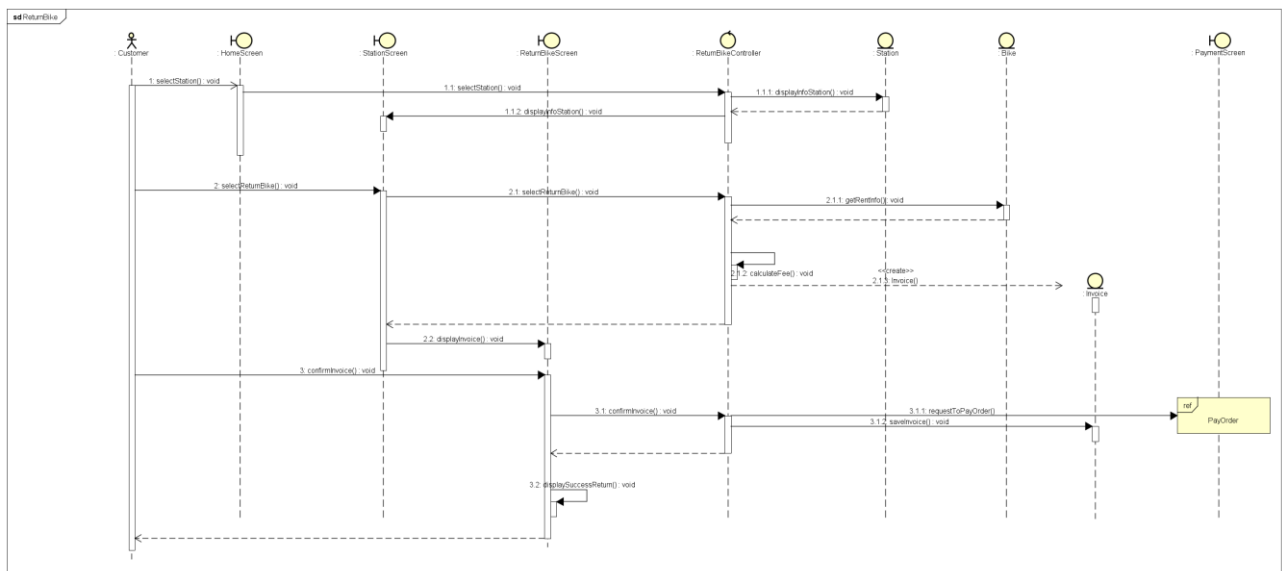
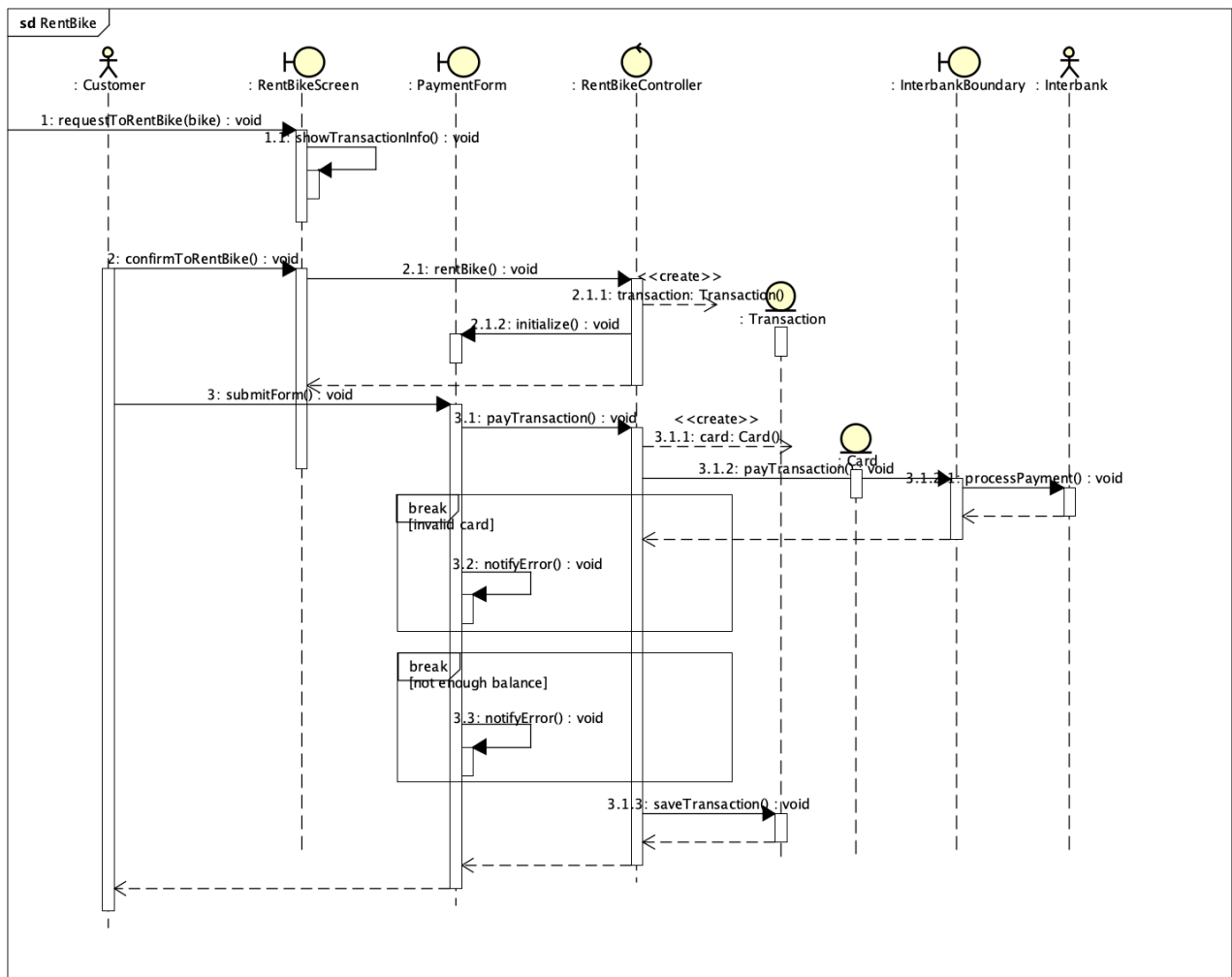
2.4.1 Biểu đồ tương tác cho UC001 – Xem thông tin bãi xe



2.4.2 Biểu đồ tương tác cho UC002 – Xem thông tin chi tiết xe



2.4.3 Biểu đồ tương tác cho UC003 - Thuê xe



2.4.4 Biểu đồ tương tác cho UC004 – Trả xe

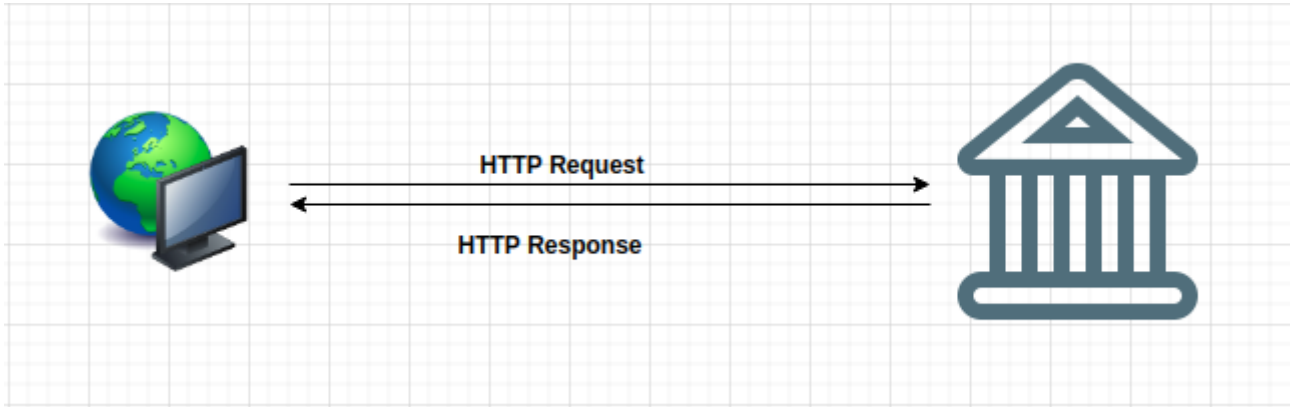
3 Thiết kế giao diện

3.1 Giao diện với thiết bị phần cứng

Không tương tác với phần cứng

3.2 Giao diện với phần mềm khác

Phần mềm tương tác với ngân hàng :



Phần mềm tương tác với ngân hàng thông qua giao thức Http qua method PATCH để thanh toán hoặc hoàn tiền:

- Ngân hàng cung cấp API

- Method: PATCH
- Path: <https://ecopark-system-api.herokuapp.com/api/card/processTransaction>

- Mỗi người dùng sẽ được cung cấp 1 cặp key là appCode và secretKey và có một tài khoản ngân hàng riêng

- Dữ liệu cần truyền đi là một chuỗi json

- Định dạng dữ liệu truyền đi

Field	Type	Required	Description
version	String	Yes	Phiên bản API: 1.0.1
transaction	Object	Yes	Giao dịch
cardCode	String	Yes	Mã thẻ
owner	String	Yes	Chủ tài khoản
cvvCode	String	Yes	Mã CVV
dateExpired	String	Yes	Ngày hết hạn
command	String	Yes	Mã API sử dụng,

- Mã cho giao dịch thanh toán là pay

- Hoàn tiền là refund

transactionContent	String	Yes	Nội dung giao dịch
amount	Number	Yes	Số tiền cần thanh toán
createdAt	String	Yes	Thời điểm tạo giao dịch (cần tuân thủ đúng format “năm-tháng-ngày giờ:phút:giây”)
appCode	String	Yes	Mã app sử dụng hệ thống thanh toán
hashCode	String	Yes	Mã kiểm tra, để đảm bảo không bị thanh đổi khi chuyển từ app lên server thanh toán

- Trước khi gửi yêu cầu lên ngân hàng thì cần phải mã hóa chuỗi thông tin giao dịch bằng mã băm MD5 kết hợp với secretKey đã được cho trước. Dưới đây là ví dụ một chuỗi JSON mã hóa:

```
{
  "secretKey": "BJrapO8Wdtw=",
  "transaction": {
    "command": "pay",
    "cardCode": "118609_group18_2020",
    "owner": "Group 18",
    "cvvCode": "390",
    "dateExpired": "Thanh toán",
    "amount": 100
  }
}
```

- Sau khi mã hóa xong sẽ lưu thông tin mã hóa vào trường hashCode

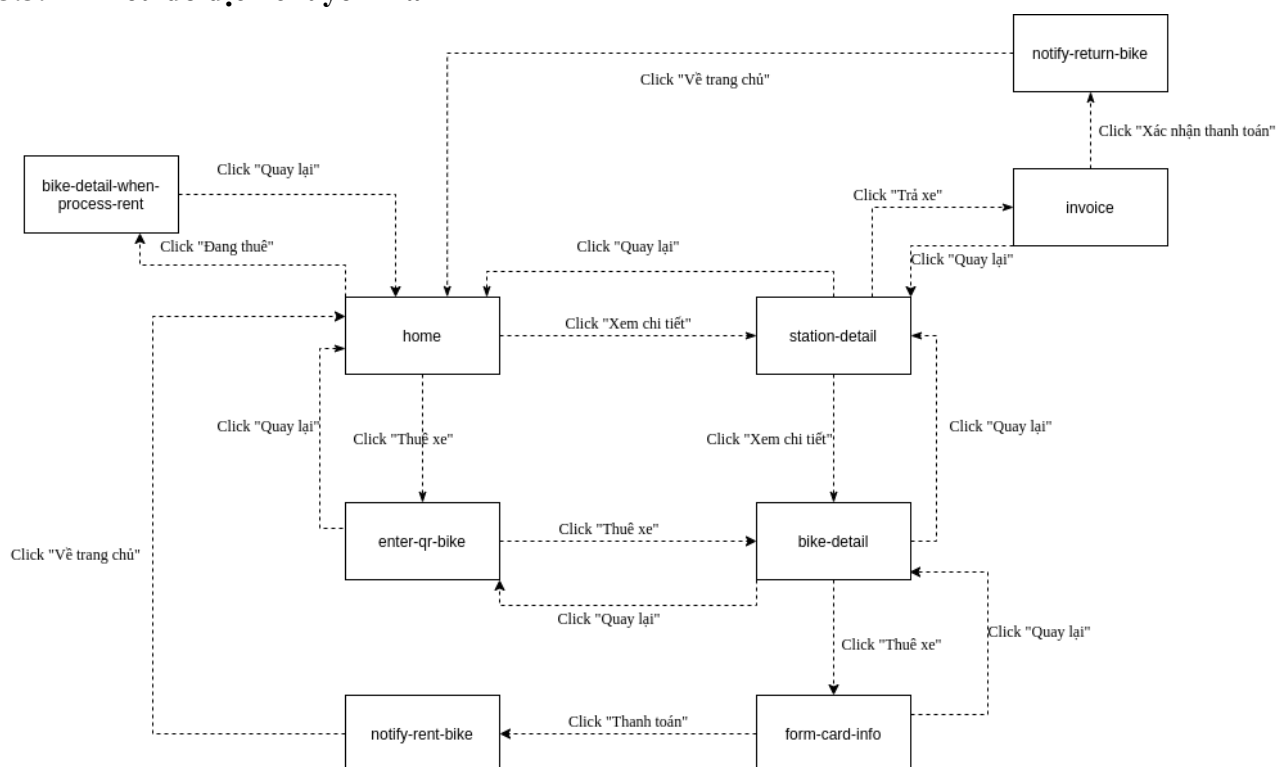
- Dữ liệu trả về:

Field	Type	Description
errorCode	Number	Mã lỗi

transaction	Object	Giao dịch
cardCode	String	Mã thẻ
owner	String	Chủ tài khoản
cvvCode	String	Mã CVV
dateExpired	String	Ngày hết hạn
command	String	Mã API sử dụng, - Mã cho giao dịch thanh toán là pay - Hoàn tiền là refund
transactionContent	String	Nội dung giao dịch
amount	Number	Số tiền cần thanh toán
createdAt	Date	Thời điểm tạo giao dịch (cần tuân thủ đúng format “năm-tháng-ngày giờ:phút:giây”)

3.3 Giao diện người dùng

3.3.1 Biểu đồ dịch chuyển màn hình



3.3.2 Thiết kế giao diện

Đặc tả màn hình Trang chủ



Control	Operation	Function
Khu vực hiển thị	Tự sinh	Khi hệ thống khởi chạy, một danh sách các bãi xe hiển thị lên màn hình
Nút "Trang chủ"	Nhấn (Click)	Người dùng ở bất cứ vị trí nào khi nhấn vào nút này thì sẽ đưa người dùng về trang chủ
Nút "Thuê xe"	Nhấn (Click)	Hệ thống sẽ đưa người dùng đến trang nhập mã QR của xe
Nút "Đang thuê"	Nhấn (Click)	Khi người dùng đang thuê xe thì hệ thống sẽ đưa người dùng đến trang thông tin của xe đang thuê còn nếu chưa thuê thì hệ thống sẽ hiển thị thông báo là khách hàng chưa thuê xe
Nút "Xem chi tiết"	Nhấn (Click)	Hệ thống sẽ đưa người dùng đến trang xem chi tiết bãi xe

Đặc tả màn hình Chi tiết bãi xe

EcobikeRental

Chào mừng bạn đến với EcobikeRental

Trang chủ
Thuê xe
Đang thuê

< Quay lại

-----Thông tin chi tiết bãi gửi xe-----

Thông tin bãi xe	
Tên bãi	AH111
Địa chỉ	HUST
Tổng xe	15
Số xe hiện tại	10

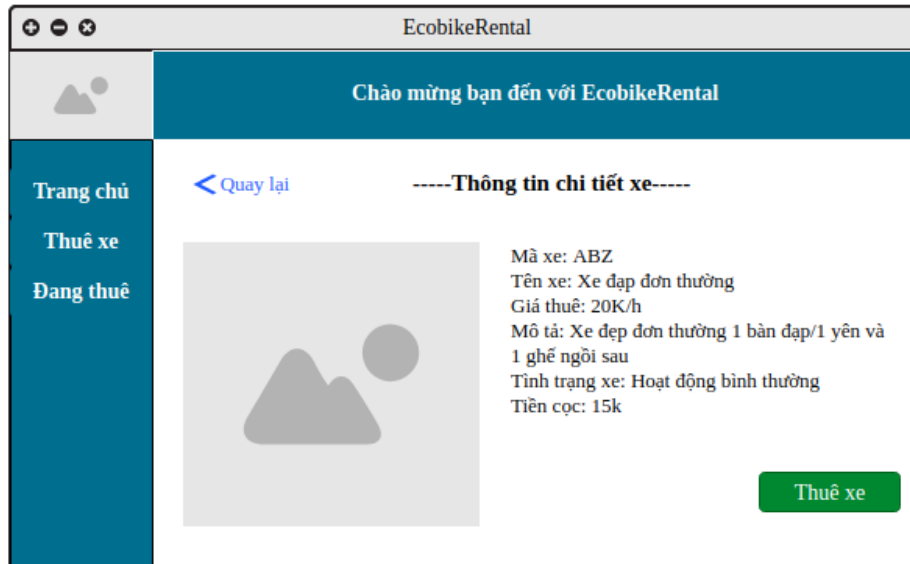
Trả xe

Danh sách xe trong bãi

Tên xe: Xe đạp đơn Giá thuê: 20K/h	Xem chi tiết
---------------------------------------	--------------

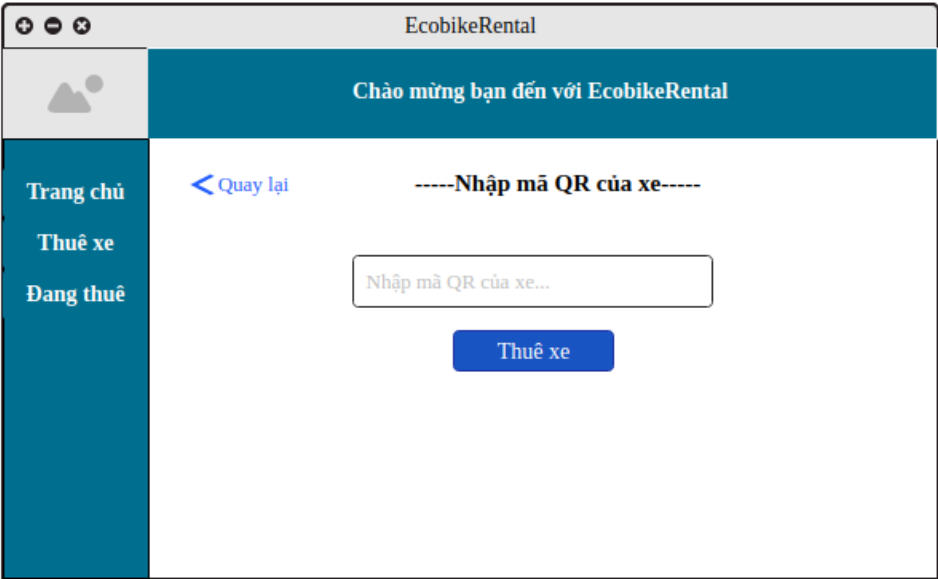
Control	Operation	Function
Khu vực hiển thị	Khi nhấn vào nút "Xem chi tiết" ở màn hình Trang chủ	Hệ thống sẽ hiển thị thông tin chi tiết của bãi xe gồm có thông tin và danh sách các xe có trong bãi
Nút "Quay lại"	Nhấn (Click)	Hệ thống sẽ đưa người dùng về Trang chủ
Nút "Trả xe"	Nhấn (Click)	Hệ thống sẽ đưa người dùng đến trang hóa đơn
Nút "Xem chi tiết"	Nhấn (Click)	Hệ thống sẽ đưa người dùng đến trang xem thông tin chi tiết của xe

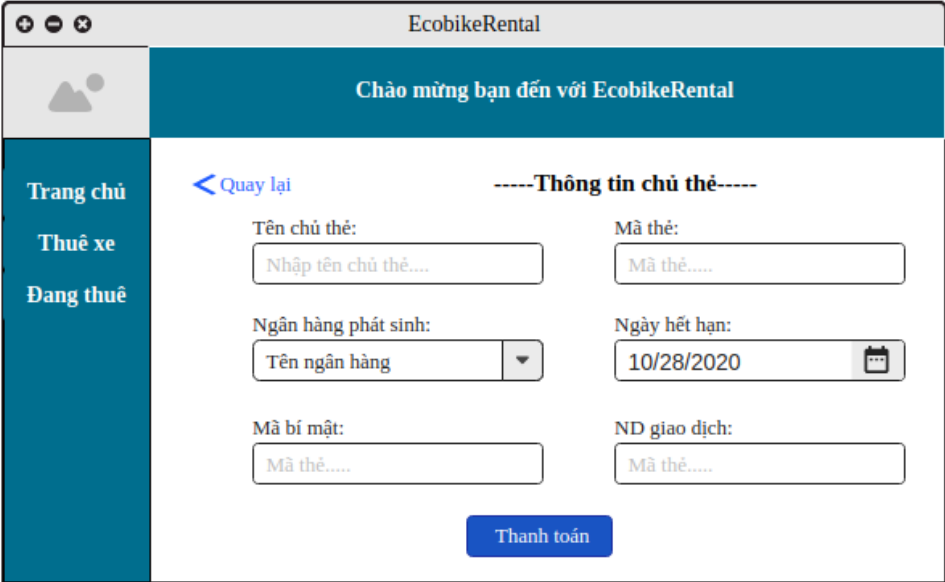
Đặc tả màn hình Chi tiết xe



Control	Operation	Function
Khu vực hiển thị	Khi nhấn vào nút "Xem chi tiết" ở trang chi tiết bãi xe	Hệ thống sẽ hiển thị thông tin chi tiết của xe trong bãi
Nút "Quay lại"	Nhấn (Click)	Hệ thống sẽ đưa người dùng về Trang chi tiết bãi xe
Nút "Thuê xe"	Nhấn (Click)	Hệ thống sẽ đưa người dùng đến trang nhập thông tin của tài khoản thẻ

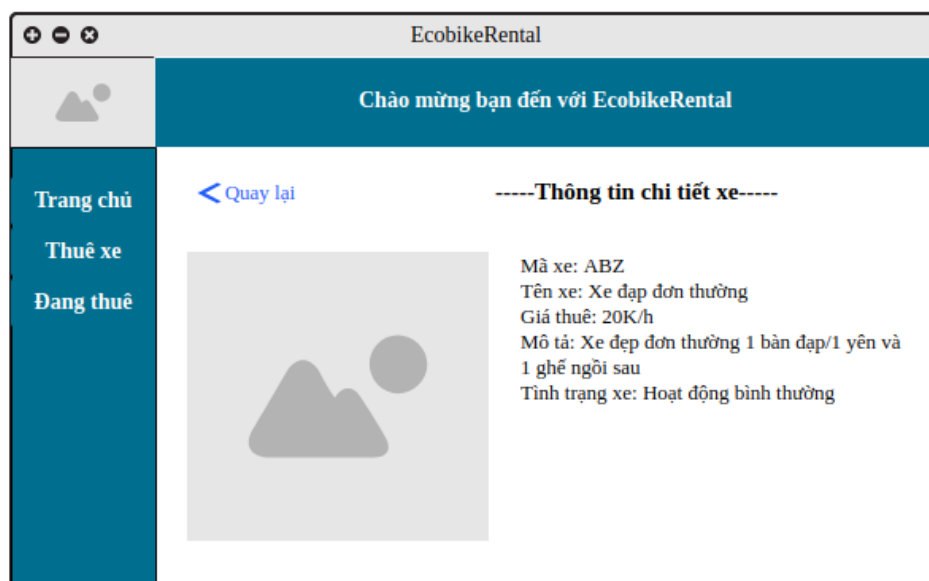
Đặc tả màn hình Nhập mã QR của xe

		
Control	Operation	Function
Khu vực hiển thị	Khi người dùng nhấn vào nút "Thuê xe" bên sidebar	Hệ thống sẽ hiển thị form để người dùng nhập mã QR của xe
Nút "Quay lại"	Nhấn (Click)	Hệ thống sẽ đưa người dùng về Trang chủ
Nút "Thuê xe"	Nhấn (Click)	Hệ thống sẽ đưa người dùng đến trang chi tiết xe

Đặc tả màn hình Nhập thông tin tài khoản thẻ		
		
Control	Operation	Function
Khu vực hiển thị	Khi người dùng nhấn vào nút "Thuê xe" ở màn hình Nhập mã QR của xe hoặc nhấn nút	Hệ thống hiển thị form để người dùng nhập thông tin tài khoản của người dùng vào

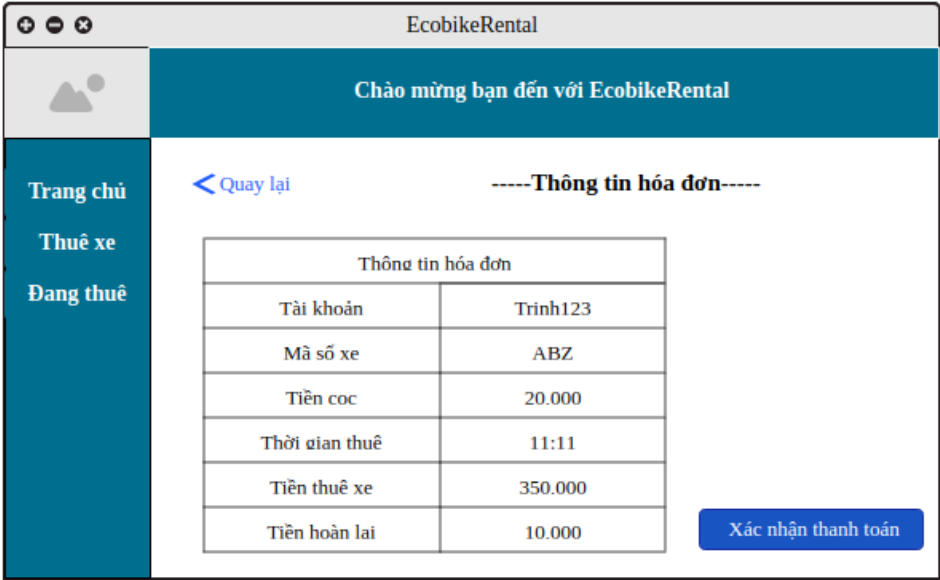
	"Thuê xe" ở màn hình Chi tiết xe	
Nút "Quay lại"	Nhấn (Click)	Hệ thống sẽ đưa người dùng về trang Chi tiết xe
Nút "Thanh toán"	Nhấn (Click)	Hệ thống sẽ giao tiếp với ngân hàng và trừ tiền trong tài khoản của khách hàng

Đặc tả màn hình Chi tiết xe khi đang thuê



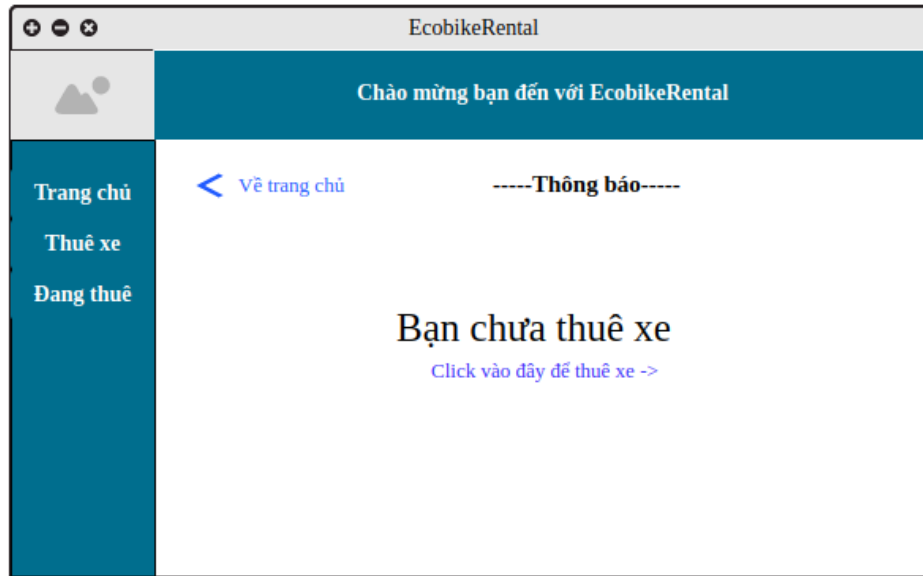
Control	Operation	Function
Khu vực hiển thị	Khi người dùng nhấn vào nút "Thuê xe" ở màn hình Nhập mã QR của xe hoặc nhấn nút "Thuê xe" ở màn hình Chi tiết xe	Hệ thống hiển thị form để người dùng nhập thông tin tài khoản của người dùng vào
Nút "Quay lại"	Nhấn (Click)	Hệ thống sẽ đưa người dùng về trang Chi tiết xe

Đặc tả màn hình Hóa đơn

		
Control	Operation	Function
Khu vực hiển thị	Khi người dùng nhấn nút "Trả xe" trong trang Chi tiết bãi xe	Hệ thống hiển thị hóa đơn của khách hàng
Nút "Quay lại"	Nhấn (Click)	Hệ thống sẽ đưa người dùng về trang Chi tiết bãi xe
Nút "Xác nhận thanh toán"	Nhấn (Click)	Hệ thống sẽ trừ tiền trong tài khoản của khách hàng

Đặc tả màn hình Thuê xe thành công		
		
Control	Operation	Function
Khu vực hiển thị	Tự sinh	Hệ thống hiển thị thông báo thuê xe thành công
Nút "Về trang chủ"	Nhấn (Click)	Hệ thống sẽ đưa người dùng về Trang chủ

Đặc tả màn hình Chưa thuê xe



Control	Operation	Function
Khu vực hiển thị	Khi click vào nút "Đang thuê" bên sidebar	Khi người dùng nhấn vào nút "Đang thuê" nếu người dùng chưa thuê xe thì sẽ hiển thị trang này
Nút "Về trang chủ"	Nhấn (Click)	Hệ thống sẽ đưa người dùng về Trang chủ
Nút "Click vào đây để thuê xe"	Nhấn (Click)	Hệ thống sẽ chuyển đến trang nhập mã QR của xe

4 Thiết kế lớp

4.1 Biểu đồ lớp thiết kế

InterbankService
- SECRETKEY : String = "B1rap08Wdtw=" - APPCODE : String = "CUgp9eRNgwU=" - VERSION : String = "1.0.1" - CARDCODE : String = null - OWNER : String = null - CVCODE : String = null - DATEEXPIRED : String = null
+ getMD5(input : String) : String + jsonInforToHash(card : Card, command : String, amount : int, currentTime : String) : String + jsonResetBalance() : String + processTransaction(card : Card, command : String, amount : int) : String + resetBalance(card : Card) : String + main(args : String[]) : void

Bike
+ id : int + price : int + battery : int + stationID : int + name : String + status : String + type : String + description : String
+ Bike(id : int) + Bike() + Bike(id : int, battery : int, stationID : int, price : int, name : String, status : String, type : String, description : String) + setBikeFromID(bikeID : int) : void + updateBike(status : String, stationID : int) : boolean + createRent(timeStart : Time, customerID : int) : boolean + getDepositMoney() : int + getBattery() : int + setBattery(battery : int) : void + getStationID() : int + setStationID(stationID : int) : void + getDescription() : String + setDescription(description : String) : void + setBike(b : Bike) : void + getPrice() : int + setPrice(price : int) : void + getName() : String + setName(name : String) : void + getStatus() : String + setStatus(status : String) : void + getType() : String + setType(type : String) : void + getId() : int + setId(id : int) : void

4.2.3 Thiết kế lớp Bike

4.2.4 T

Station
+ stationID : int + totalBike : int + available : int + name : String + address : String
+ setStation(s : Station) : void + Station(stationID : int, name : String, position : String) + setStationFromID(stationID : int) : void + getAllBikes() : ArrayList<Bike> + getBikeByID(bikeID : int) : Bike + Station(stationID : int, totalBike : int, available : int, name : String, address : String) + Station() + getStationID() : int + setStationID(stationID : int) : void + getTotalBike() : int + setTotalBike(totalBike : int) : void + getAvailable() : int + setAvailable(available : int) : void + getAddress() : String + getPosition() : String + setAddress(address : String) : void + getName() : String + setName(name : String) : void

lớp Station

4.2.5 Thiết kế chi tiết lớp Card

Card
+ cardID : int + cardHolderName : String + cardNumber : String + transactionDescription : String + expirationDate : String + securityCode : String + issuingBank : String
+ Card() + Card(cardID : int) + Card(cardID : int, cardHolderName : String, cardNumber : String, transactionDescription : String, expirationDate : String, securityCode : String, issuingBank : String) + setCard(c : Card) : void + setCardFromCardNumber(cardNumber : String) : void + getCardID() : int + setCardID(cardID : int) : void + getCardHolderName() : String + setCardHolderName(cardHolderName : String) : void + getCardNumber() : String + setCardNumber(cardNumber : String) : void + getTransactionDescription() : String + setTransactionDescription(transactionDescription : String) : void + getExpirationDate() : String + setExpirationDate(expirationDate : String) : void + getSecurityCode() : String + setSecurityCode(securityCode : String) : void + getIssuingBank() : String + setIssuingBank(issuingBank : String) : void

4.2.6 T

Customer
+ cardNumber : String + customerID : int + rentID : int + customerName : String
+ Customer() + getRentingBike() : Rent + getCustomerCard() : Card + Customer(customerID : int, cardNumber : String, rentID : int, customerName : String) + Customer(customerID : int, rentID : int, customerName : String) + Customer(id : int, name : String) + getCardNumber() : String + setCardNumber(cardNumber : String) : void + getCustomerID() : int + setCustomerID(customerID : int) : void + getRentID() : int + setRentID(rentID : int) : void + getCustomerName() : String + setCustomerName(customerName : String) : void

ết lớp Customer

h
i
ế
t

k
ế

c
h
i

t
i

Rent
+ rentID : int + customerID : int + bikeID : int + totalTimeRent : int + status : String
+ Rent() + Rent(id : int) + setRent(r : Rent) : void + updateRent(timeEnd : Time, totalTimeRent : int) : boolean + setRentFromID(rentID : int) : void + createTransaction(code : String, transactionName : String, totalMoney : int) : boolean + Rent(rentID : int, status : String, customerID : int, bikeID : int, timeStart : Time, timeEnd : Time, totalTimeRent : int) + Rent(rentID : int, status : String, customerID : int, bikeID : int, timeStart : Time) + getStatus() : String + setStatus(status : String) : void + getRentID() : int + setRentID(rentID : int) : void + getCustomerID() : int + setCustomerID(customerID : int) : void + getBikeID() : int + setBikeID(bikeID : int) : void + getTimeStart() : Time + setTimeStart(timeStart : Time) : void + getTimeEnd() : Time + setTimeEnd(timeEnd : Time) : void + getTotalTimeRent() : int + setTotalTimeRent(totalTimeRent : int) : void

4.2.7 Thiết kế chi tiết lớp Rent

4.2.8 Thiết kế chi tiết lớp HomeController

HomeController
+ initialize(arg0 : URL, arg1 : ResourceBundle) : void + addEvents() : void + getRentingInfo() : void + updateView() : void + clearView() : void + viewListStation() : void

4.2.9 Thiết kế chi tiết lớp ListStationController

ListStationController
+ initialize(arg0 : URL, arg1 : ResourceBundle) : void + addEvents() : void + addControl() : void + getAllStation() : void + viewStation() : void + backToPrevious() : void

4.2.10 Thiết kế chi

tiết lớp PaymentFormController

PaymentForm Controller
+ initialize(arg0 : URL, arg1 : ResourceBundle) : void + addEvents() : void + submitPaymentForm() : void + rentBike(depositMoney : int) : void + checkBlankField() : boolean + showMessage(mess : String) : void + clearTextField() : void + setupTextField() : void

4.2.11 Thiết kế chi tiết lớp RentBikeCon troller

RentBikeController
+ initialize(arg0 : URL, arg1 : ResourceBundle) : void + addEvents() : void + showTransactionInfo() : void + rentBike() : void

4.2.12 Thiết kế chi tiết lớp ReturnBikeController

ReturnBikeController
~ totalTimeRent : int = 0 ~ totalMoneyRent : int = 0 ~ depositMoney : int = 0 + initialize(arg0 : URL, arg1 : ResourceBundle) : void + addEvents() : void + getRentingInfo() : void + updateView() : void + submitReturnBike() : void + returnBike(code : String, totalMoney : int) : void + createTransaction(totalTimeRent : int, totalMoney : int, rentID : int) : void + updateRentingBike() : void + showMessage(mess : String) : void

4.2.13 Thiết kế chi tiết lớp ViewBikeController

ViewBikeController
+ initialize(arg0 : URL, arg1 : ResourceBundle) : void + addEvents() : void + showBikeInfo() : void + checkRentAvailable() : boolean + showMessage(mess : String) : void + showPaymentForm() : void

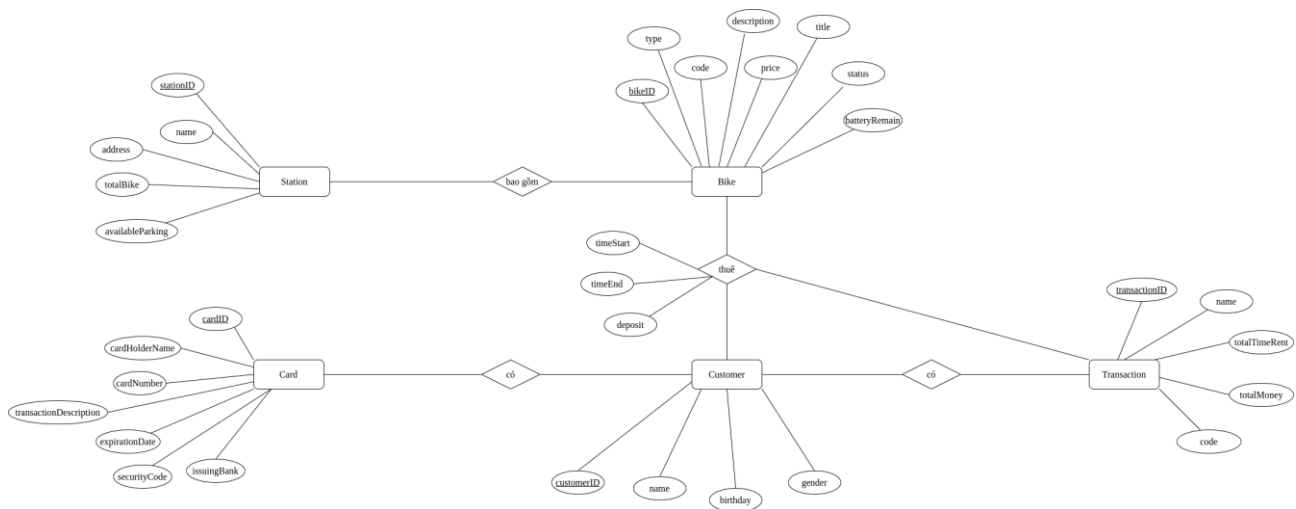
4.2.14 Thiết kế chi tiết lớp ViewStationController

ViewStationController
<pre>+ initialize(arg0 : URL, arg1 : ResourceBundle) : void + addEvents() : void + addControl() : void + getStationInfo() : void + getAllBikes() : void + showBikeInfo() : void + checkBikeRenting() : void + showReturnBike() : void + searchBike() : void + showMessage(mess : String) : void</pre>

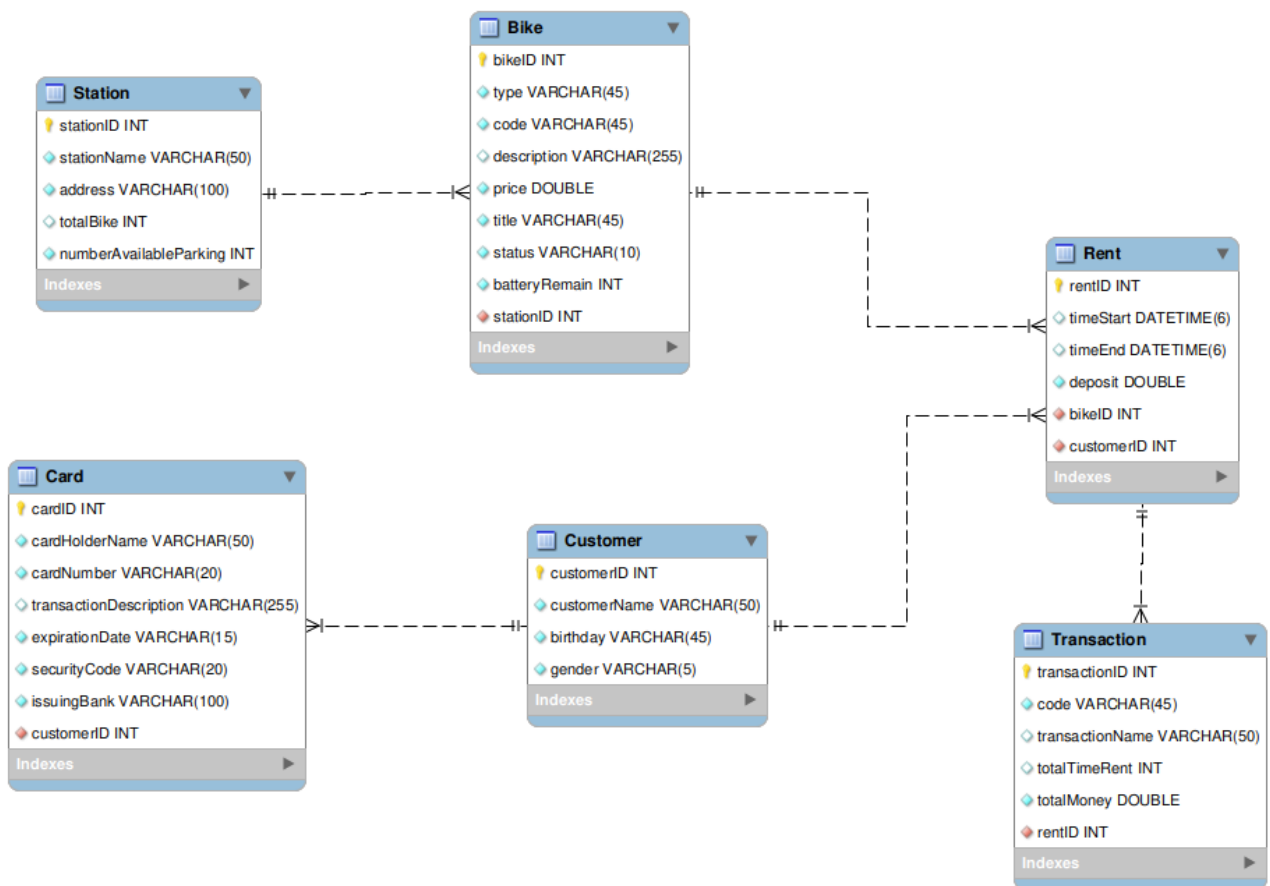
5 Thiết kế mô hình dữ liệu

5.1 Mô hình dữ liệu mức khái niệm

- Biểu đồ thực thể liên kết:



5.2 Mô hình dữ liệu mức logic



5.3 Thiết kế chi tiết

5.3.1 Bảng Station(Bãi xe)

Lưu thông của bãi gửi xe

Tên cột	Kiểu dữ liệu	Khóa chính	Khóa ngoại	Duy nhất	Ràng buộc	Mô tả
stationID	integer	Yes		Yes	NOT NULL	Khóa chính của bảng tự tăng
stationName	varchar(50)			Yes	NOT NULL	Tên của bến xe
address	varchar(100)				NOT NULL	Địa chỉ của bến xe
numberTotalBike	integer				NOT NULL	Tổng số xe trong bãi có thể chứa
availableParking	integer				NOT NULL	Số xe còn lại có thể nhận

5.3.2 Bảng Bike (Xe)

Lưu thông tin của xe

Tên cột	Kiểu dữ liệu	Khóa chính	Khóa ngoại	Duy nhất	Ràng buộc	Mô tả
bikeID	integer	Yes		Yes	NOT NULL	Khóa chính của bảng tự tăng
type	varchar(50)					Kiểu xe (xe đạp thường hay xe đạp điện,...)
code	varchar(45)			Yes	NOT NULL	Mã của xe
description	varchar(255)					Mô tả của xe
price	double					Giá tiền của xe
title	varchar(45)					Tên của xe
status	varchar(10)					Thẻ hiện xe đang được thuê hay chưa
batteryRemain	integer					Lượng pin còn lại của xe đối với xe điện
stationID	integer		Yes	Yes		Khóa ngoại liên kết với bảng Station

5.3.3 Bảng Customer (Khách hàng)

Lưu thông tin của khách hàng

Tên cột	Kiểu dữ liệu	Khóa chính	Khóa ngoại	Duy nhất	Ràng buộc	Mô tả
customerID	integer	Yes		Yes	NOT NULL	Khóa chính của bảng tự tăng
customerName	varchar(50)				NOT NULL	Tên khách hàng
birthday	varchar(100)					Ngày tháng năm sinh của khách hàng
gender	integer					Giới tính của khách hàng

5.3.4 Bảng Card (Tài khoản thẻ)

Lưu thông tin tài khoản thẻ của khách hàng

Tên cột	Kiểu dữ liệu	Khóa chính	Khóa ngoại	Duy nhất	Ràng buộc	Mô tả
cardID	integer	Yes		Yes	NOT NULL	Khóa chính của bảng tự tăng
cardHolderName	varchar(50)				NOT NULL	Tên chủ thẻ
cardNumber	varchar(100)			Yes	NOT NULL	Số thẻ
transactionDescription	varchar(255)					Nội dung giao dịch
expirationDate	varchar(15)				NOT NULL	Ngày hết hạn
securityCode	varchar(20)				NOT NULL	Mã bí mật
issuingBank	varchar(100)				NOT NULL	Ngân hàng phát hành
customerID	integer		Yes	Yes	NOT NULL	Khóa ngoại liên kết với bảng Customer

5.3.5 Bảng Transaction (Thông tin giao dịch)

Lưu thông tin giao dịch của khách hàng với hệ thống

Tên cột	Kiểu dữ liệu	Khóa chính	Khóa ngoại	Duy nhất	Ràng buộc	Mô tả
transactionID	integer	Yes		Yes	NOT NULL	Khóa chính của bảng tự tăng
code	varchar(50)				NOT NULL	Mã loại giao dịch
transactionName	varchar(100)				NOT NULL	Tên giao dịch
totalTimeRent	float					Tổng thời gian thuê xe tính theo giờ
totalMoney	double				NOT NULL	Tổng số tiền phải trả
rentID	varchar(20)		Yes	Yes	NOT NULL	Khóa ngoại liên kết với bảng Rent

5.3.6 Bảng Rent (Thuê xe)

Lưu thông tin thuê xe

Tên cột	Kiểu dữ liệu	Khóa chính	Khóa ngoại	Duy nhất	Ràng buộc	Mô tả
rentID	integer	Yes		Yes	NOT NULL	Khóa chính của bảng tự tăng
timeStart	datetime(6)				NOT NULL	Thời gian bắt đầu thuê
timeEnd	datetime(6)				NOT NULL	Thời gian kết thúc thuê
deposit	float				NOT NULL	Số tiền đặt cọc
bikeID	double		Yes	Yes	NOT NULL	Khóa ngoại liên kết với bảng Bike
customerID	varchar(20)		Yes	Yes	NOT NULL	Khóa ngoại liên kết với bảng Customer