

TRƯỜNG ĐẠI HỌC KHOA HỌC VÀ CÔNG NGHỆ HÀ NỘI

Viện Công nghệ Thông tin và Truyền thông

TÀI LIỆU THIẾT KẾ PHẦN MỀM

Software Design Document

## Mục lục

<b>1. Giới thiệu</b>	4
1.1 Mục tiêu	4
1.2 Phạm vi	4
1.3 Bảng chú giải thuật ngữ	4
1.4 Tài liệu tham khảo	4
<b>2. Mô tả chung</b>	5
2.1 Tổng quan chung	5
2.2 Giả định / Ràng buộc / Rủi ro	5
2.2.1 Giả định	5
2.2.2 Ràng buộc	5
2.2.3 Rủi ro	5
<b>3. Kiến trúc hệ thống và thiết kế kiến trúc</b>	6
3.1 Mô hình kiến trúc	6
3.2 Biểu đồ tương tác	7
3.2.1 Biểu đồ trình tự	7
3.2.2 Biểu đồ giao tiếp	9
3.3 Biểu đồ lớp phân tích	12
<b>4. Thiết kế chi tiết</b>	14
4.1 Thiết kế giao diện người dùng	14
4.1.1 Chuẩn hóa cấu hình màn hình	14
4.1.2 Sơ đồ chuyển đổi màn hình	14
4.1.3 Đặc tả màn hình	15
4.2 Mô hình dữ liệu	21
4.2.1 Mô hình hóa dữ liệu khái niệm	21
4.2.2 Thiết kế cơ sở dữ liệu	21
4.3 Non-Database Management System Files	22
4.4 Thiết kế lớp	22
4.4.1 Biểu đồ lớp tổng quát	22
4.4.2 Biểu đồ lớp	23
<b>5. Cân nhắc kiến trúc</b>	27
5.1 Mục tiêu và nguyên tắc	27
5.1.1 Mục tiêu phần mềm	27
5.1.2 Nguyên tắc	27
5.2 Chiến lược kiến trúc	27

5.3 Coupling and Cohesion .....	27
5.3.1 Coupling.....	27
5.3.2 Conhesion .....	28
5.4 Nguyên tắc thiết kế.....	28

## **1. Giới thiệu**

### **1.1 Mục tiêu**

Tài liệu này đưa ra mô tả chi tiết cho các chức năng người dùng có thể sử dụng được tại thời gian chạy. Tài liệu mô tả mục đích và các tính năng của hệ thống, các giao diện, ràng buộc của hệ thống cần thực hiện để phản ứng tới các kích thích bên ngoài.

### **1.2 Phạm vi**

Trong thực tế, bất kỳ phần mềm nào cũng cần có các tính năng quản lý người dùng, nhóm người dùng, và cần phân quyền sử dụng các chức năng trong hệ thống một cách linh động.

Mục đích của phần mềm nhằm tạo ra phân hệ quản lý người dùng (user), vai trò của người dùng (role) và các chức năng (function) mà người dùng / vai trò người dùng có thể sử dụng tại thời điểm chạy. Người dùng có thể đăng ký để tạo ra tài khoản cho mình, sau đó có thể đăng nhập để sử dụng các chức năng của hệ thống. Người dùng có thể đăng nhập sử dụng tài khoản của hệ thống, hoặc đăng nhập sử dụng tài khoản Facebook. Bất kỳ người dùng nào cũng được cập nhật thông tin cá nhân của mình. Khi người dùng quên mật khẩu, có thể yêu cầu hệ thống cho phép mình thiết lập lại mật khẩu qua liên kết kèm token gửi qua email đã đăng ký.

Trong phạm vi môn học ta không xét các chức năng xác thực người dùng như đăng ký, đăng... nhập mà chỉ quan tâm các chức năng liên quan đến trả xe, thuê xe

### **1.3 Bảng chú giải thuật ngữ**

- API (Application Programming Interface): là các phương thức, giao thức kết nối với các thư viện và ứng dụng khác. API cung cấp khả năng cung cấp khả năng truy xuất đến một tập các hàm hay dùng.

### **1.4 Tài liệu tham khảo**

- “EcobikeRental”-Nguyễn Thị Thu Trang.
- Tài liệu đặc tả yêu cầu phần mềm

## **2. Mô tả chung**

### **2.1 Tổng quan chung**

Khi hệ thống khởi chạy, một danh sách các bãi xe hiện lên màn hình thay vì bản đồ. Khách hàng vẫn có thể xem các thông tin về bãi xe, loại xe.

Khi thuê xe, sau khi chọn bãi xe để thuê, khách hàng nhập mã vạch tương ứng của xe muốn thuê và hệ thống sẽ gọi đến một API để chuyển mã vạch thành mã xe trong hệ thống.

Tương tự như vậy, khi trả xe, sau khi chọn bãi xe để trả xe, khách hàng cũng nhập mã vạch tương ứng của xe cần trả thay vì đóng khóa xe.

Ngoài ra, khi hiển thị thông tin về xe, hệ thống cũng cần hiển thị thông tin về lượng pin còn lại của xe. Để đơn giản, chúng ta không cần quan tâm việc thay đổi giá trị của lượng pin này trong quá trình xe được mượn hay được trả tại bãi.

Biểu đồ use case tổng quan này là use case phức hợp của một nhóm các use case. Chi tiết về các use case phức này được đưa ra trong các biểu đồ phân rã ở phần sau.

### **2.2 Giả định / Ràng buộc / Rủi ro**

#### **2.2.1 Giả định**

Trong ứng dụng EcoBikeRental sẽ chỉ quan tâm đến các chức năng liên quan tới thuê xe/ trả xe. Khi thuê xe khách hàng nhập mã vạch tương ứng của xe muốn thuê và hệ thống sẽ gọi đến một API để chuyển mã vạch thành mã xe trong hệ thống. Để trả xe thì hệ thống có tính năng cho phép khách hàng chọn bãi để trả xe. Khách hàng sẽ sử dụng thẻ tín dụng làm phương thức thanh toán. Để thực hiện các thao tác với thẻ tín dụng, hệ thống sẽ gọi đến một số API được cung cấp sẵn: API trừ tiền( sử dụng để cọc tiền khi thuê xe và thanh toán số tiền của một lần thuê), API cộng tiền( sử dụng để thực hiện hoàn trả lại tiền cọc cho khách hàng), API reset, API xem số dư ( sử dụng để xem số dư trong tài khoản)

#### **2.2.2 Ràng buộc**

- Yêu cầu về giao diện dễ sử dụng, thân thiện với người dùng. Làm nổi bật các tính năng thuê xe, trả xe, xem bãi, xem thông tin xe, ...
- Các chức năng thiết kế sao cho dễ thao tác. Cần có thông báo hướng dẫn lỗi sai để người dùng biết là lỗi gì và biết các sửa lỗi
- Thiết bị cần có internet, bật định vị GPS

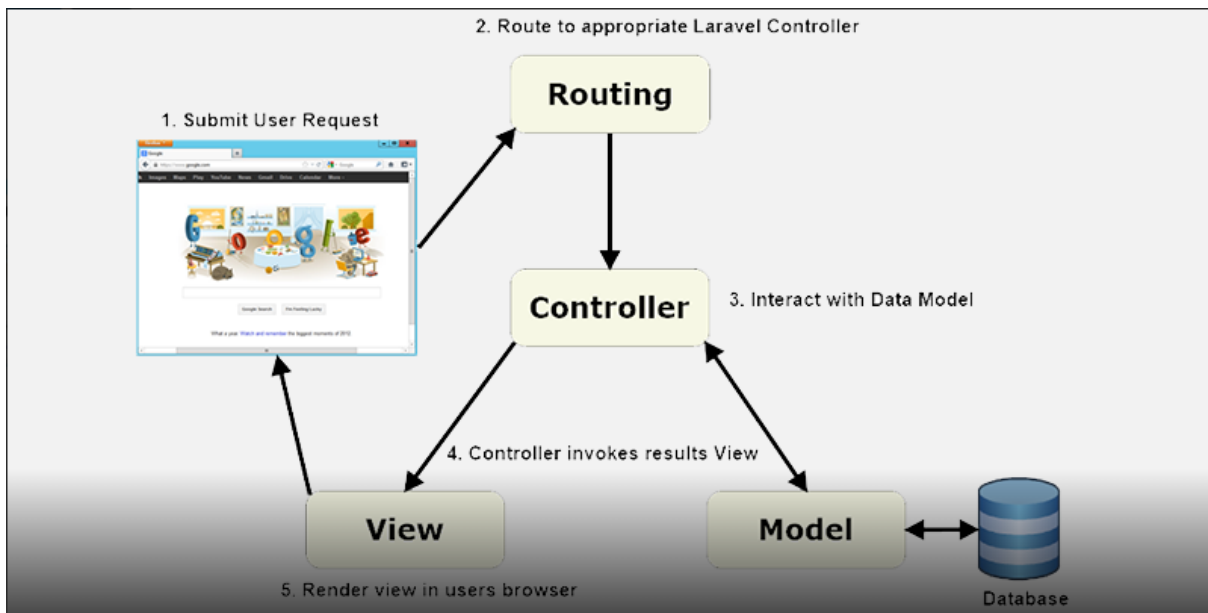
#### **2.2.3 Rủi ro**

- Các thông báo hướng dẫn lỗi sai không được hiển thị
- Giao diện không dễ sử dụng đối với người dùng

### 3. Kiến trúc hệ thống và thiết kế kiến trúc

#### 3.1 Mô hình kiến trúc

Website kiến trúc theo mô hình MVC model – view – controller

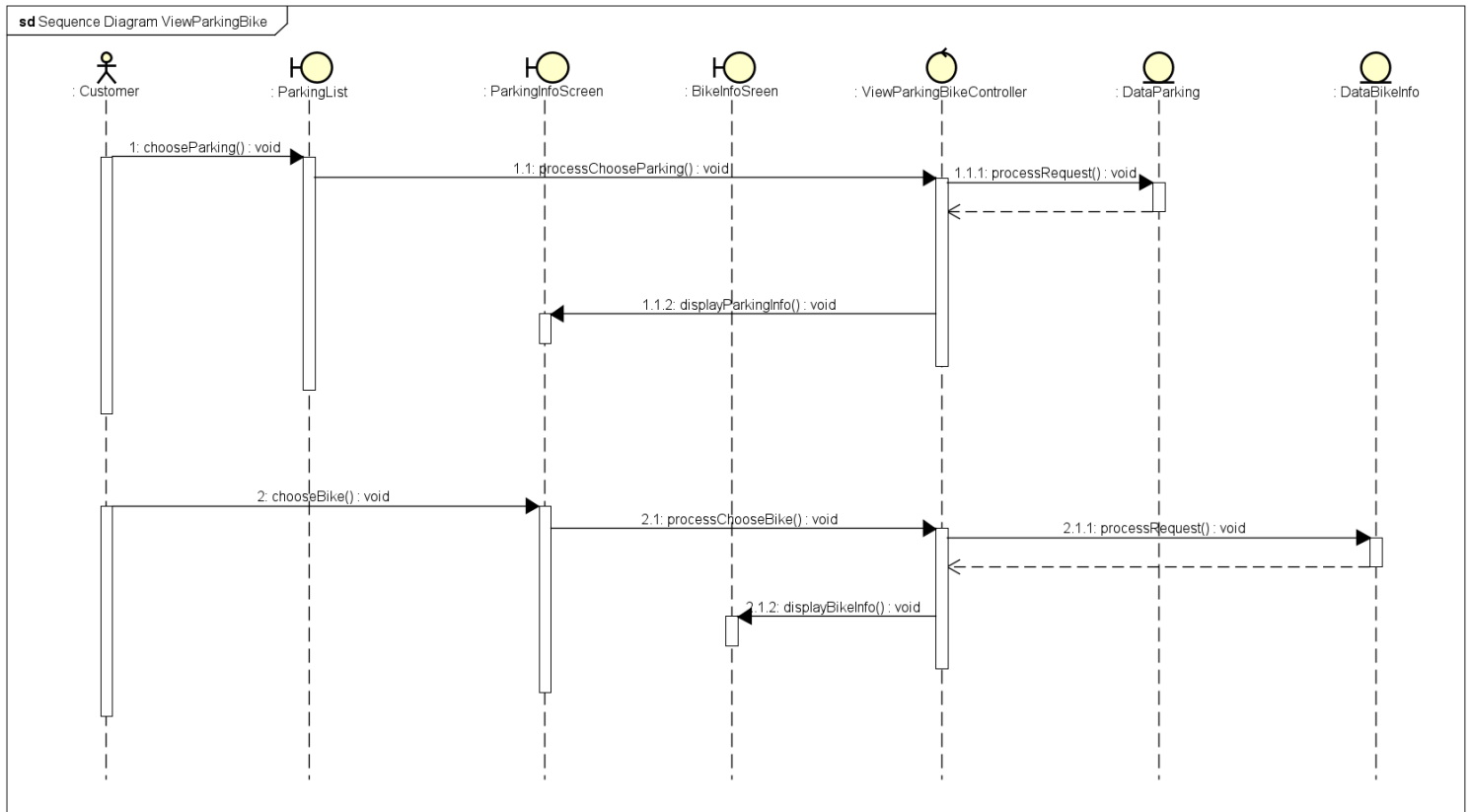


- Model :
  - Quản lý dữ liệu của ứng dụng
  - Biểu diễn, vận chuyển thông tin
  - Chứa các đối tượng mô tả dữ liệu
- View:
  - Tương tác với user
  - Lấy request từ người dùng và chuyển cho Controller xử lý
  - Show kết quả của Controller
  - View là hệ thống các frame, cửa sổ của ứng dụng; các trang giao diện web: html, jsp, bảng, biểu mẫu...
- Controller
  - Định nghĩa các hành vi, hoạt động, xử lý của hệ thống.
  - Đối chiếu hành động của người sử dụng từ View. Đồng thời tương tác Model để gọi View và hiển thị thông tin tương ứng cho người dùng.

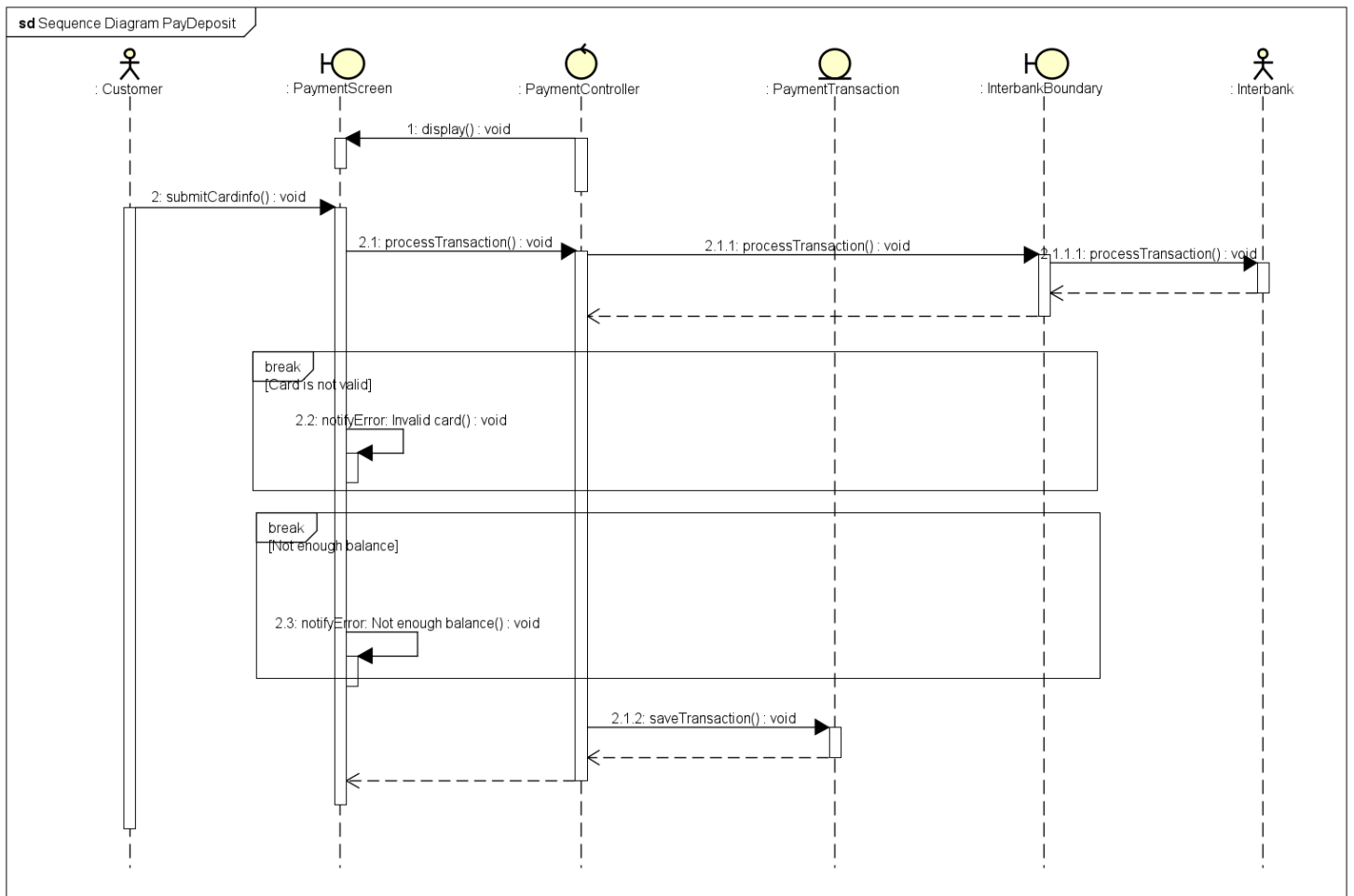
## 3.2 Biểu đồ tương tác

### 3.2.1 Biểu đồ trình tự

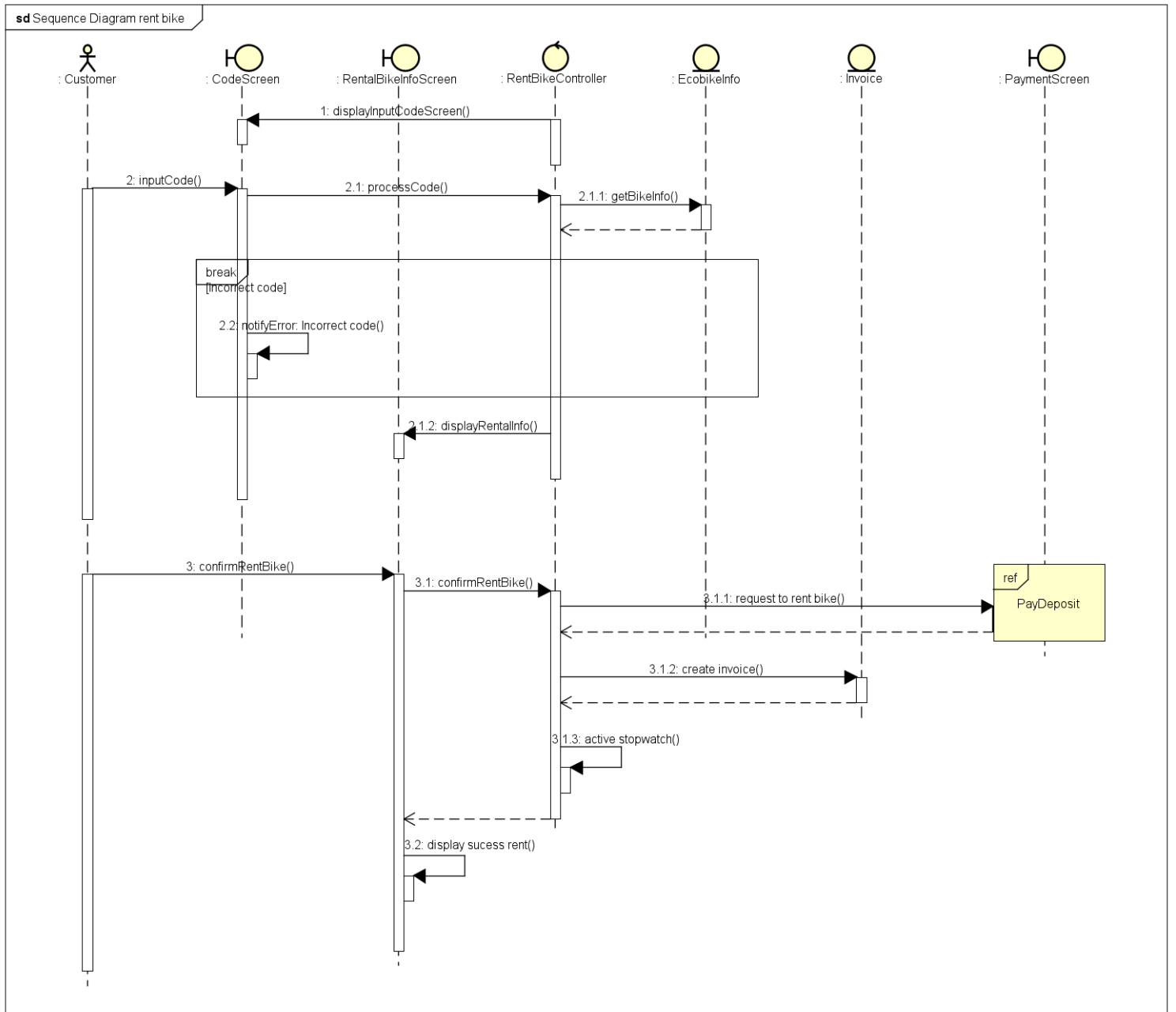
- Biểu đồ trình tự xem thông tin xe bãi xe



- Biểu đồ trình tự thanh toán

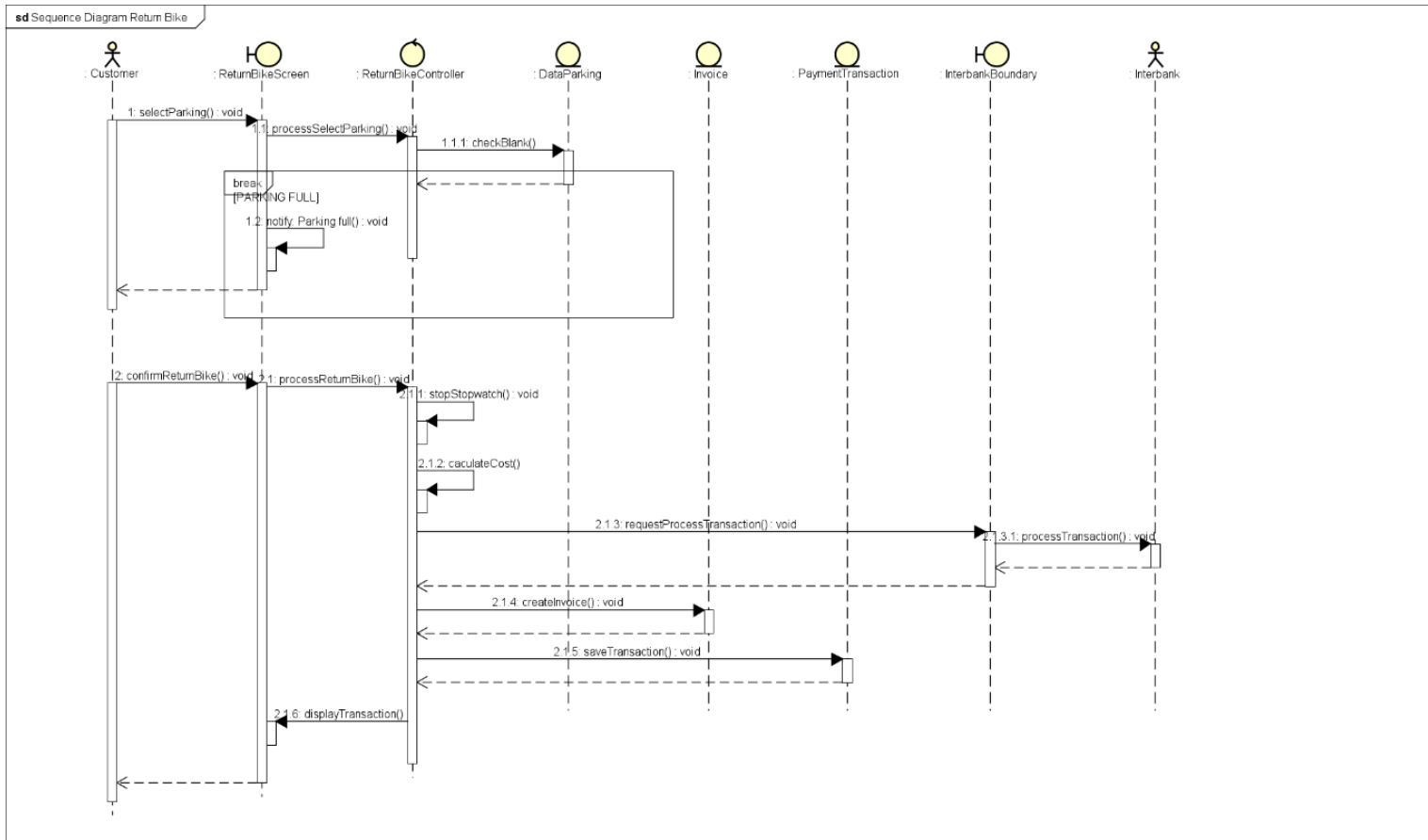


## - Biểu đồ trình tự thuê xe



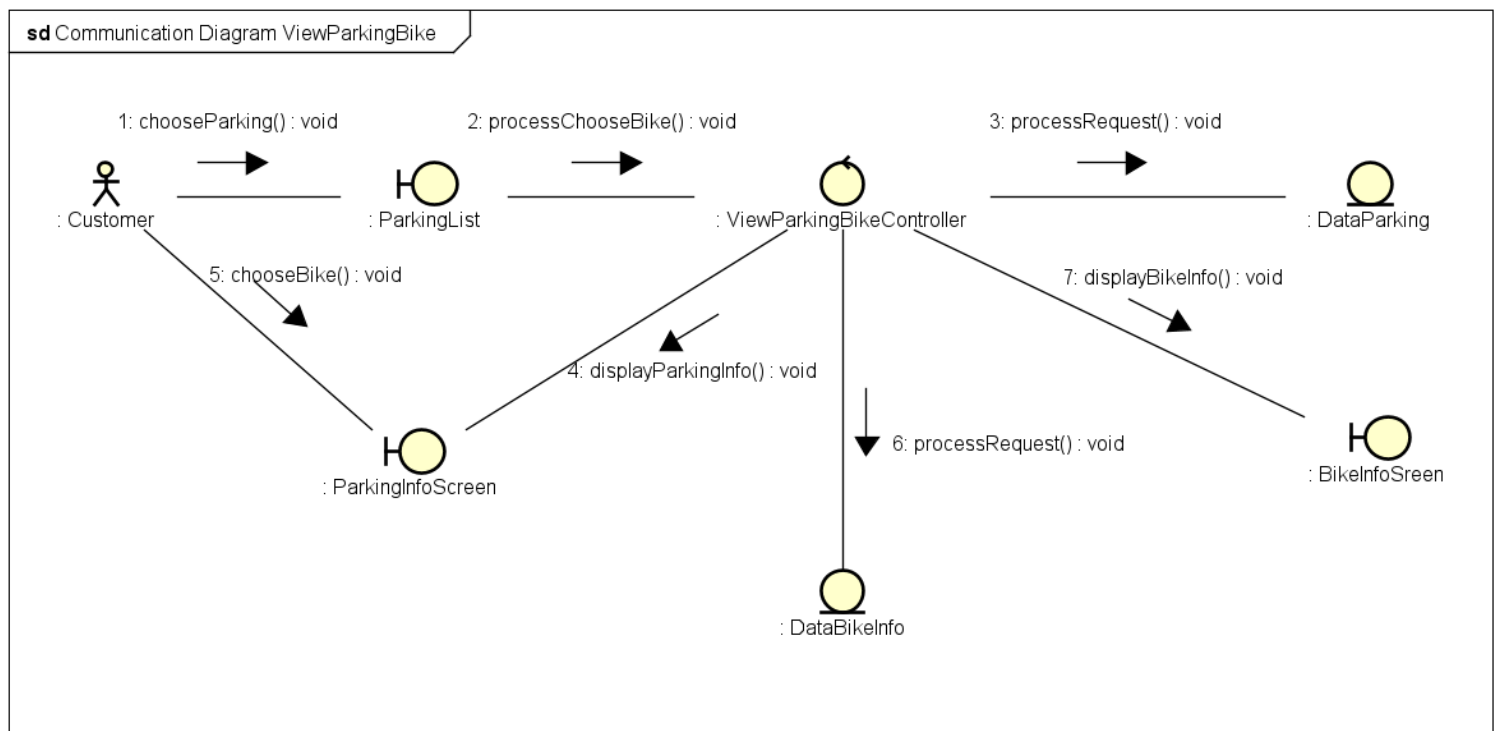


## - Biểu đồ trình tự trả xe

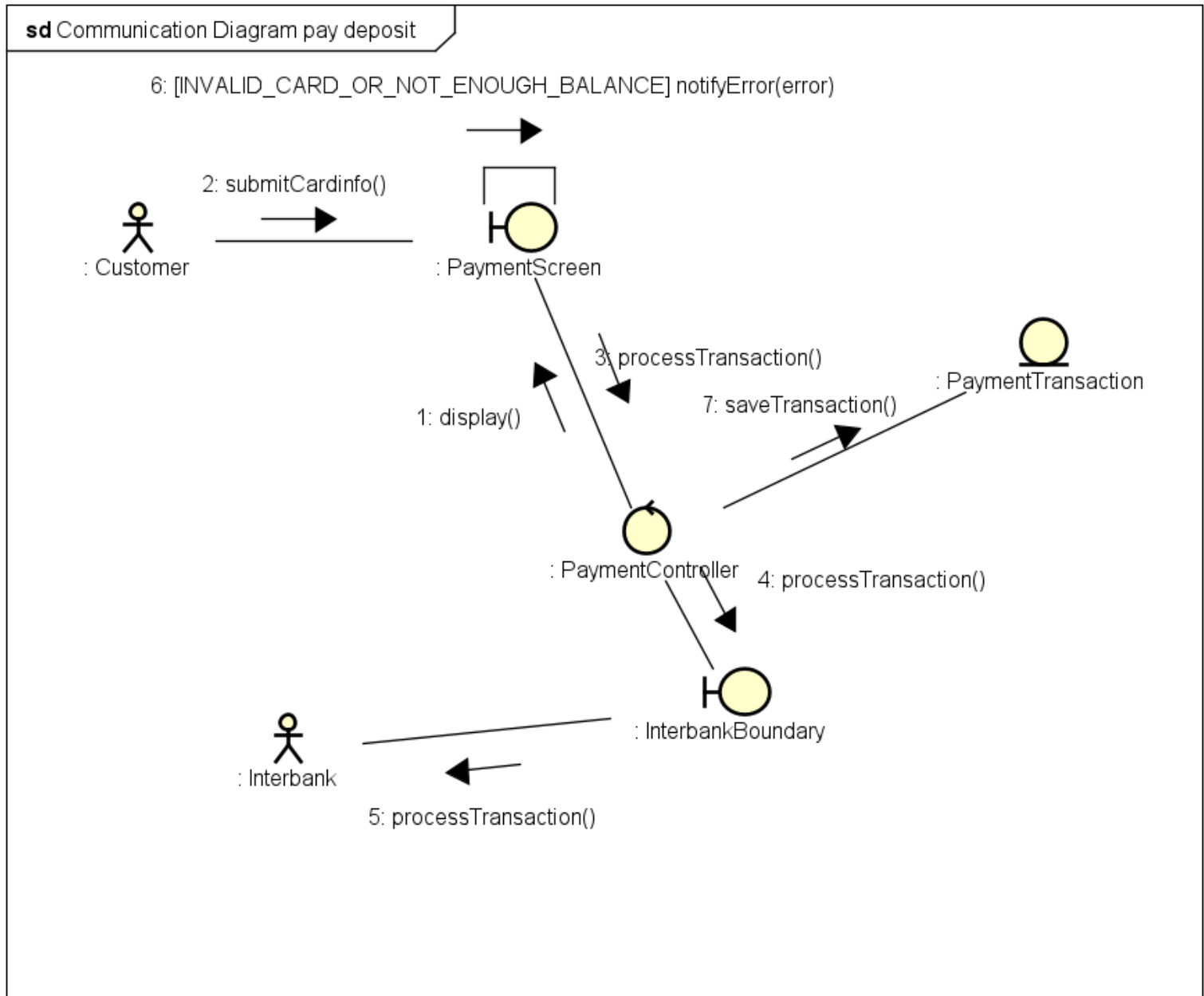


## 3.2.2 Biểu đồ giao tiếp

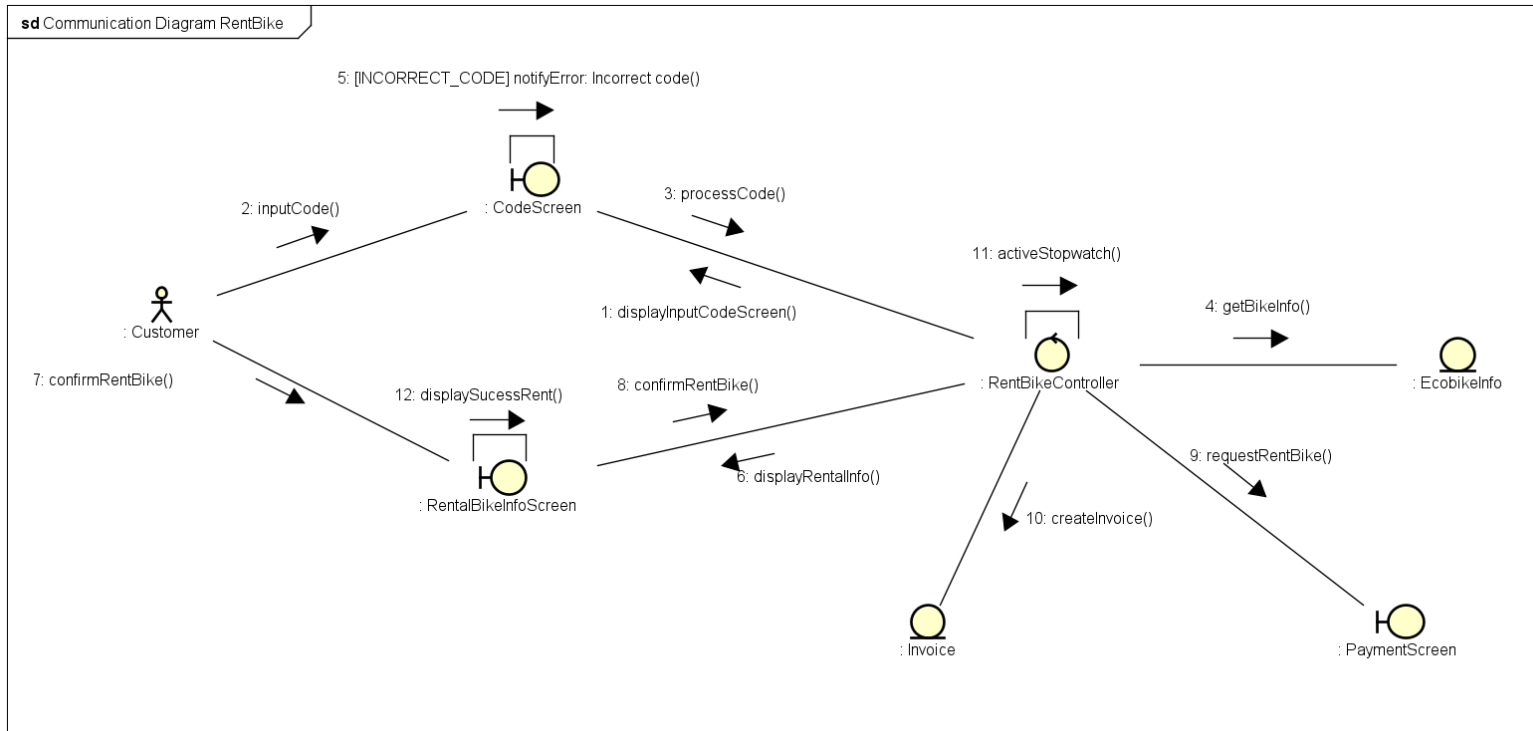
### - Biểu đồ giao tiếp xem thông tin xe bãi xe



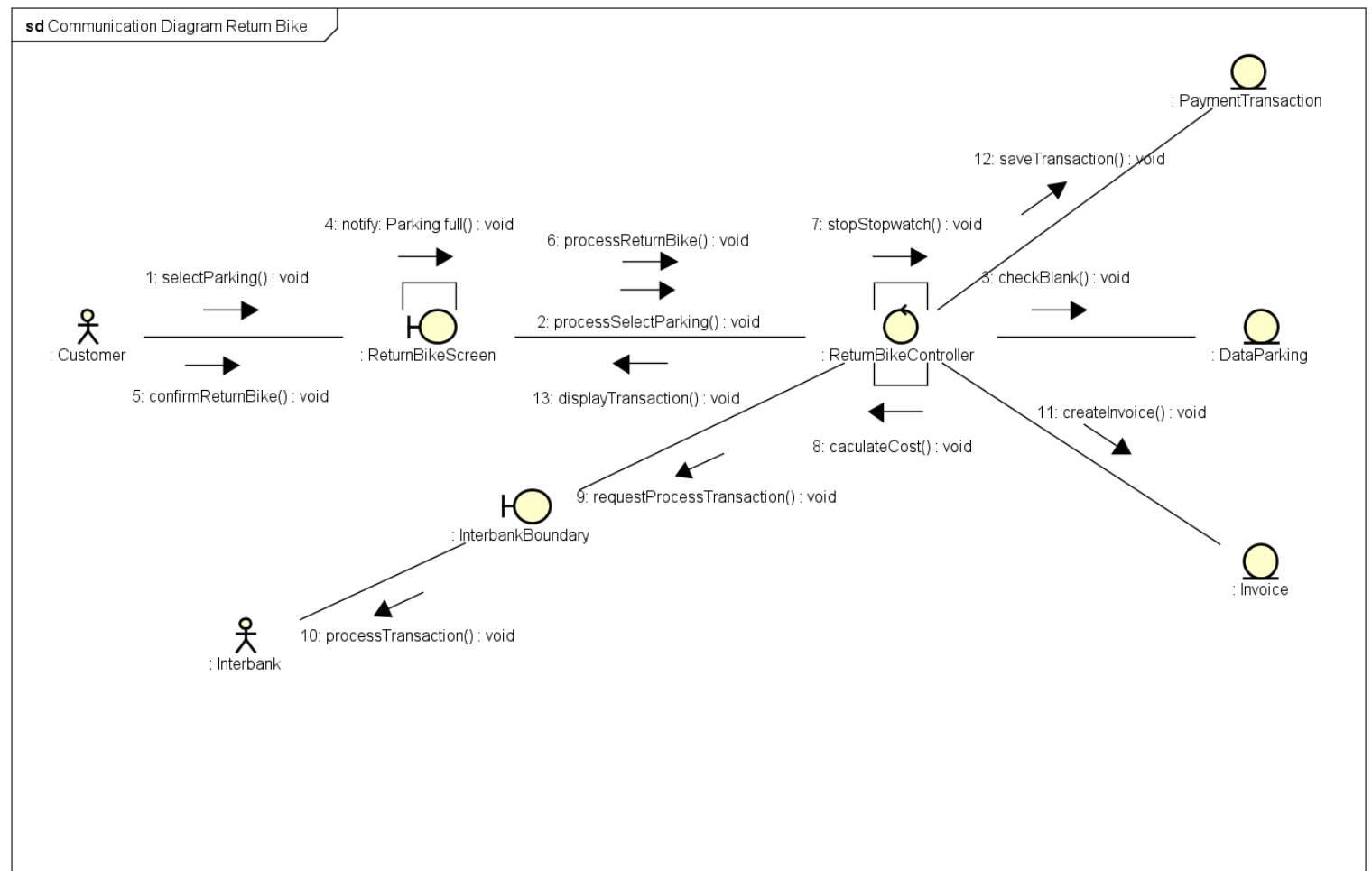
- Biểu đồ giao tiếp thanh toán



## - Biểu đồ giao tiếp thuê xe

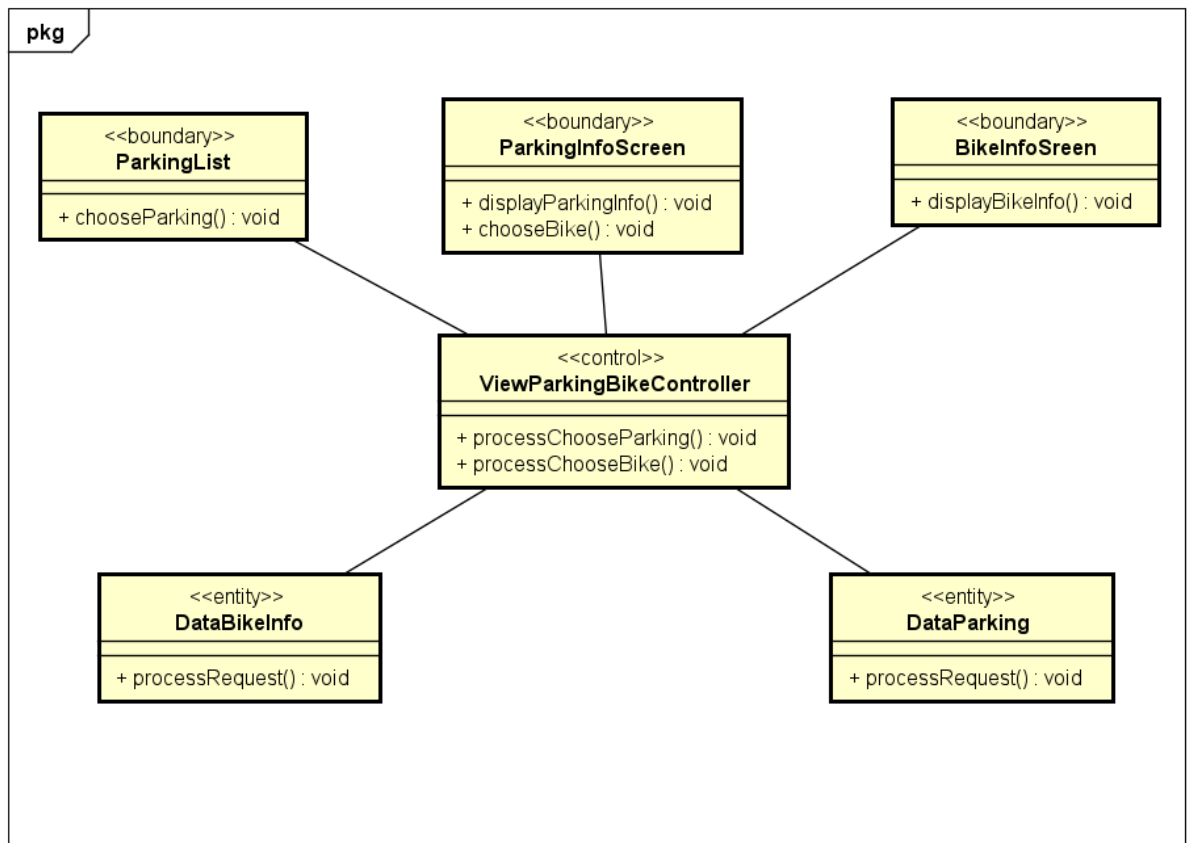


## - Biểu đồ giao tiếp trả xe

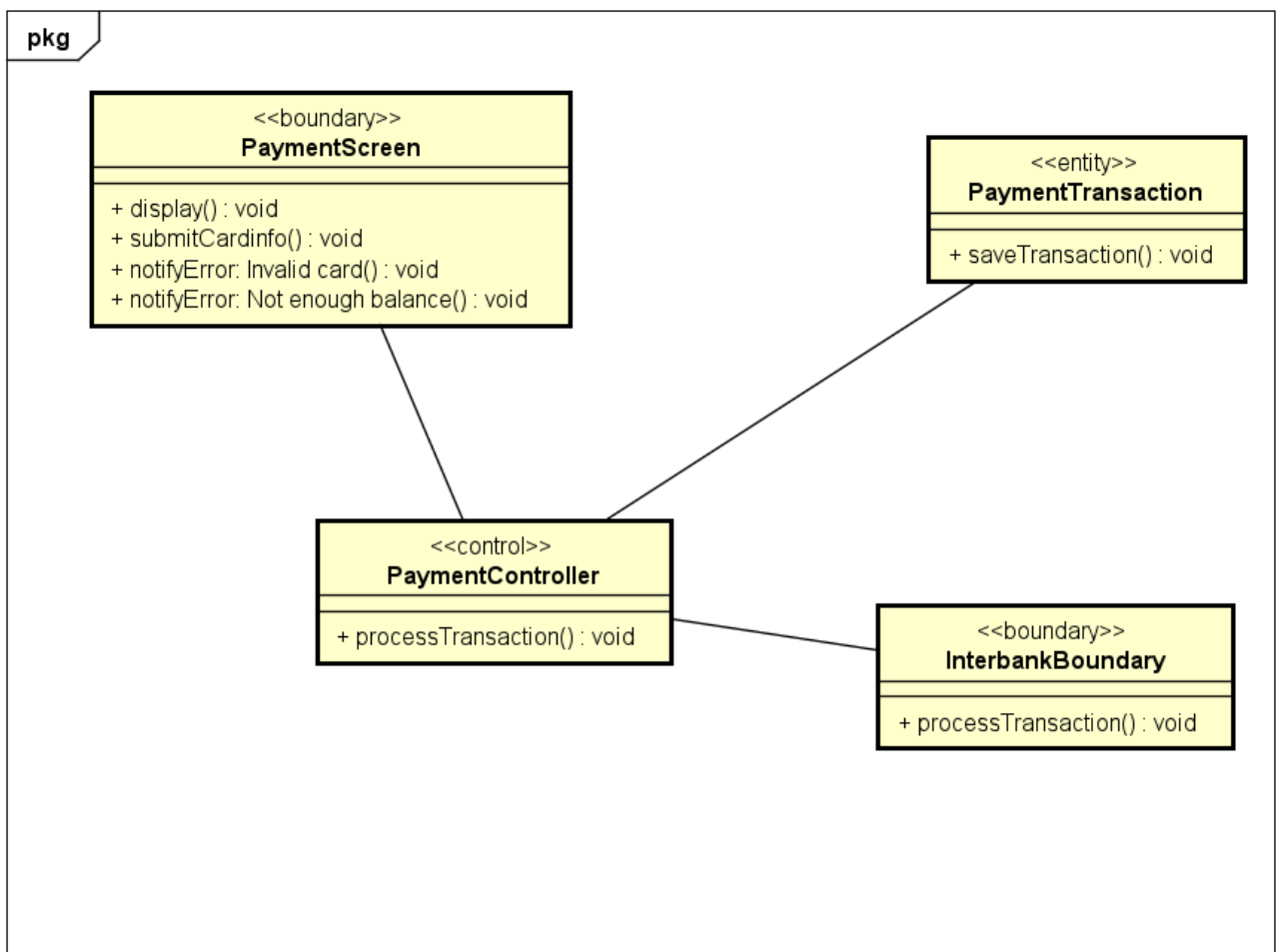


### 3.3 Biểu đồ lớp phân tích

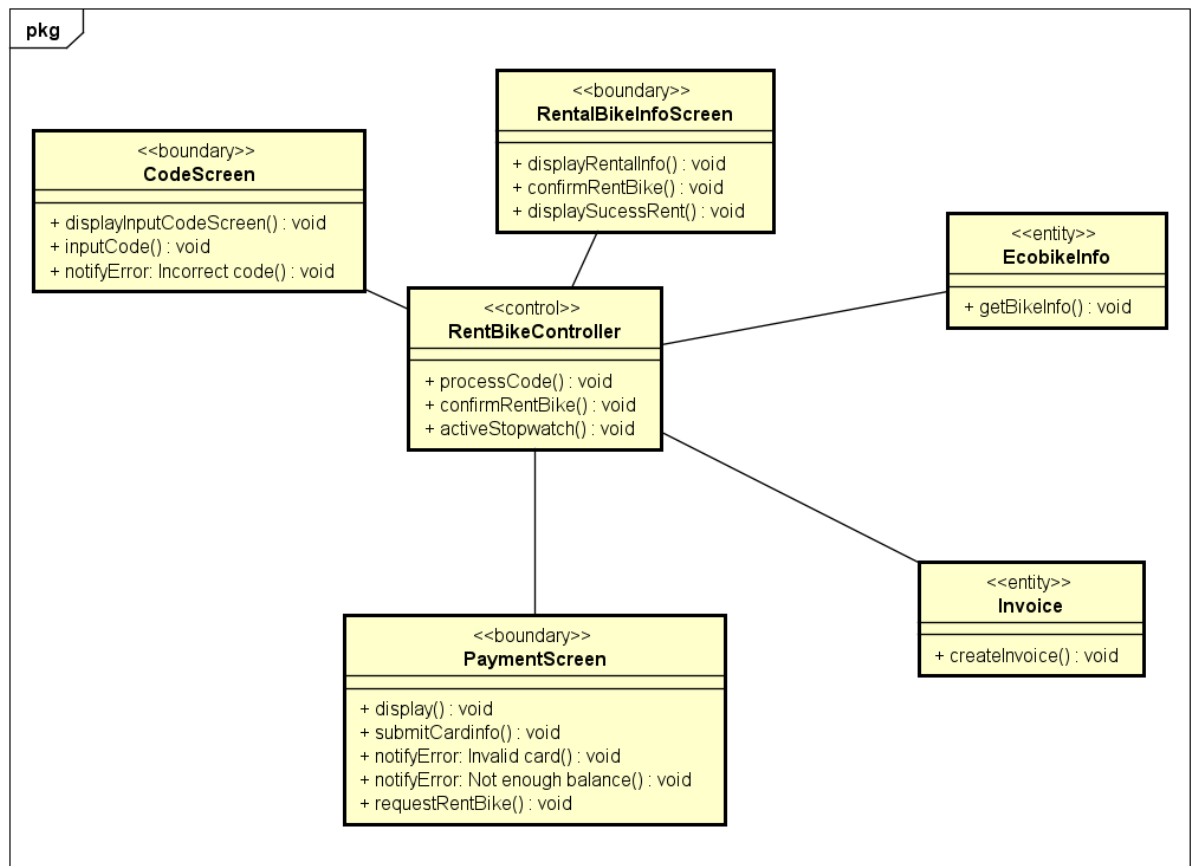
- Biểu đồ lớp phân tích xem thông tin xe bãi xe



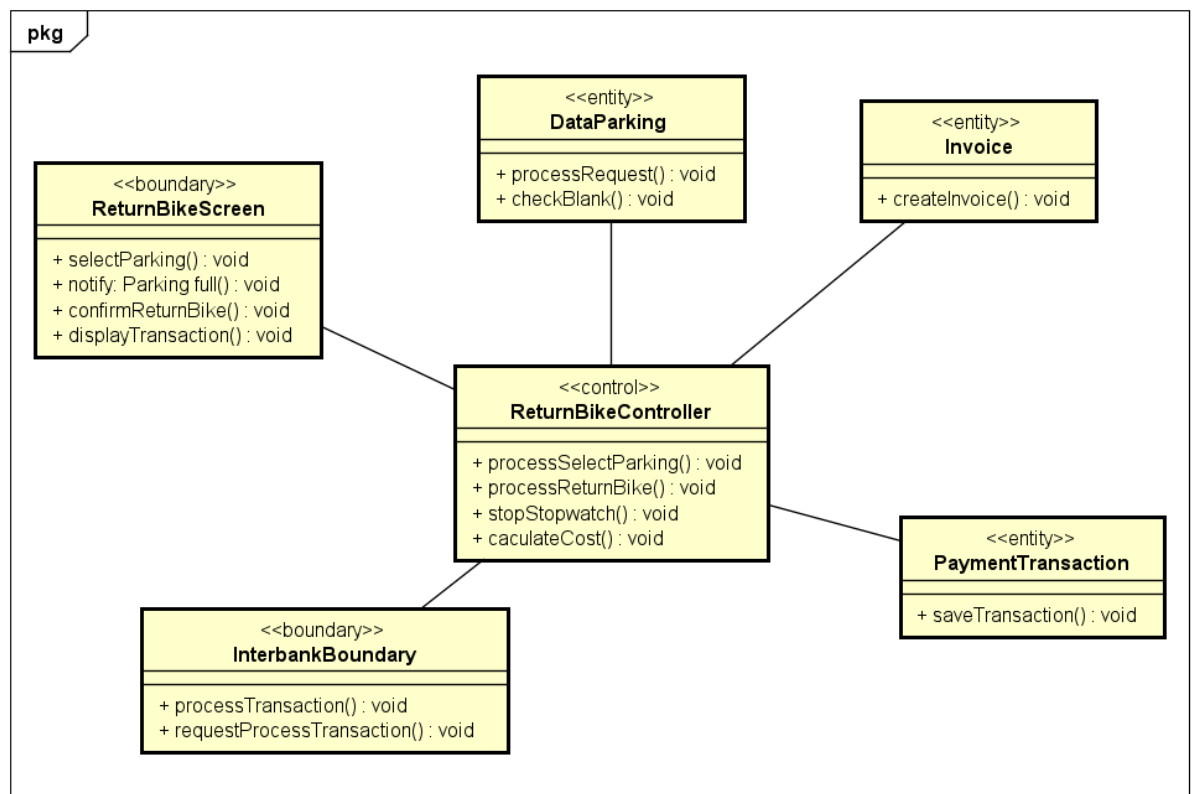
- Biểu đồ lớp phân tích thanh toán



- Biểu đồ lớp phân tích thuê xe



- Biểu đồ lớp phân tích trả xe



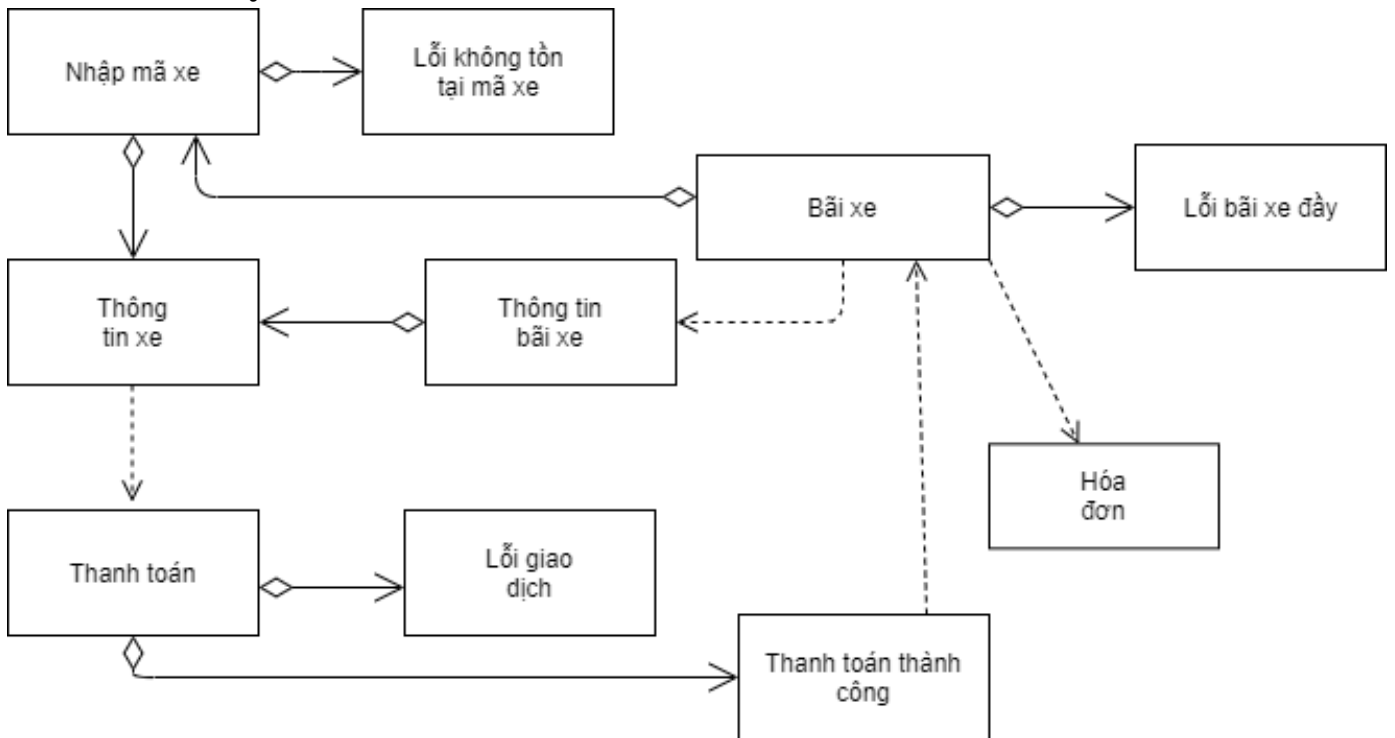
## 4. Thiết kế chi tiết

## 4.1 Thiết kế giao diện người dùng

### 4.1.1 Chuẩn hóa cấu hình màn hình

- Giao diện thiết kế màn hình thân thiện, dễ sử dụng
- Các tính năng nổi bật như tìm xe, thuê xe, trả xe, xem bãi, ...
- Chuẩn hóa: sử dụng các chuẩn hóa công nghệ đã tồn tại. Ví dụ: chuẩn giao diện W3C, Microsoft Windows
- Nguyên tắc cấu trúc: tổ chức giao diện người dùng một cách có chủ đích, dựa trên các mô hình rõ ràng, nhất quán và dễ nhận biết đối với người sử dụng
- Nguyên tắc hiển thị: Làm cho tất cả các tùy chọn và yếu tố cần thiết cho một tác vụ nhất định hiển thị cùng lúc mà không làm cho người sử dụng mất tập trung với thông tin không liên quan hoặc dư thừa.
- Nguyên tắc phản hồi: Thiết kế phải thông báo cho người sử dụng về các hành động hoặc giải thích về các thay đổi trạng thái, điều kiện và các lỗi hoặc các trường hợp ngoại lệ có liên quan và được người sử dụng quan tâm. Quá trình này phải thông qua ngôn ngữ rõ ràng, ngắn gọn và quen thuộc với họ.

### 4.1.2 Sơ đồ chuyển đổi màn hình



EcoBikeRental		Ngày tạo	Chấp nhận bởi	Đã đánh giá bởi	Người phụ trách																														
Đặc điểm màn hình	Màn hình chính: Danh sách bãi xe	23/11/2020	Lâm	Liên	Lan																														
<div><div>Bãi xe</div><div><div>← → ↻</div><div></div></div><div><div>DANH SÁCH BÃI XE</div><div><div>Thuê xe</div><div>Vui lòng trả xe nếu muốn thuê!</div><table><thead><tr><th>Tên bãi</th><th>Vị trí</th><th>Tổng số ô chứa</th><th>Số ô trống</th><th>Chi tiết bãi</th><th>Trả xe</th></tr></thead><tbody><tr><td>1</td><td>Đường 1</td><td>6</td><td>3</td><td><a href="#">Xem bãi</a></td><td><a href="#">Trả xe</a></td></tr><tr><td>2</td><td>Đường 2</td><td>9</td><td>2</td><td><a href="#">Xem bãi</a></td><td><a href="#">Trả xe</a></td></tr><tr><td>3</td><td>Đường 3</td><td>12</td><td>4</td><td><a href="#">Xem Bãi</a></td><td><a href="#">Trả xe</a></td></tr><tr><td>4</td><td>Đường 4</td><td>15</td><td>8</td><td><a href="#">Xem bãi</a></td><td><a href="#">Trả xe</a></td></tr></tbody></table></div></div></div>		Tên bãi	Vị trí	Tổng số ô chứa	Số ô trống	Chi tiết bãi	Trả xe	1	Đường 1	6	3	<a href="#">Xem bãi</a>	<a href="#">Trả xe</a>	2	Đường 2	9	2	<a href="#">Xem bãi</a>	<a href="#">Trả xe</a>	3	Đường 3	12	4	<a href="#">Xem Bãi</a>	<a href="#">Trả xe</a>	4	Đường 4	15	8	<a href="#">Xem bãi</a>	<a href="#">Trả xe</a>	Điều khiển	Hoạt động	Chức năng	
		Tên bãi	Vị trí	Tổng số ô chứa	Số ô trống	Chi tiết bãi	Trả xe																												
		1	Đường 1	6	3	<a href="#">Xem bãi</a>	<a href="#">Trả xe</a>																												
		2	Đường 2	9	2	<a href="#">Xem bãi</a>	<a href="#">Trả xe</a>																												
		3	Đường 3	12	4	<a href="#">Xem Bãi</a>	<a href="#">Trả xe</a>																												
		4	Đường 4	15	8	<a href="#">Xem bãi</a>	<a href="#">Trả xe</a>																												
Button Thuê xe	Click	Hiện thị màn hình nhập mã xe muốn thuê. Được hiện thị khi chưa có xe nào được thuê																																	
Khu vực hiển thị vui lòng trả xe	Initial	Được hiển thị khi đã có xe được thuê																																	
Khu vực danh sách bãi đậu xe	Initial	Hiện thị danh sách bãi đậu xe trong bãi																																	
Liên kết <a href="#">Xem bãi</a>	Click	Xem bãi xe được chọn, trả về màn hình thông tin bãi xe																																	
Liên kết <a href="#">Trả xe</a>	Click	Được hiển thị khi đã có xe được thuê, trả về màn hình hóa đơn																																	
Đạch tả màn hình	Thông tin bãi xe	23/11/2020	Lâm	Liên	Lan																														

← → 🔍

Thông tin bãi xe

Bãi xe

Tên bãi: Bãi xe 1  
Vị trí: Đường 1  
Số ô chứa trong bãi: 6  
Số ô trống: 3

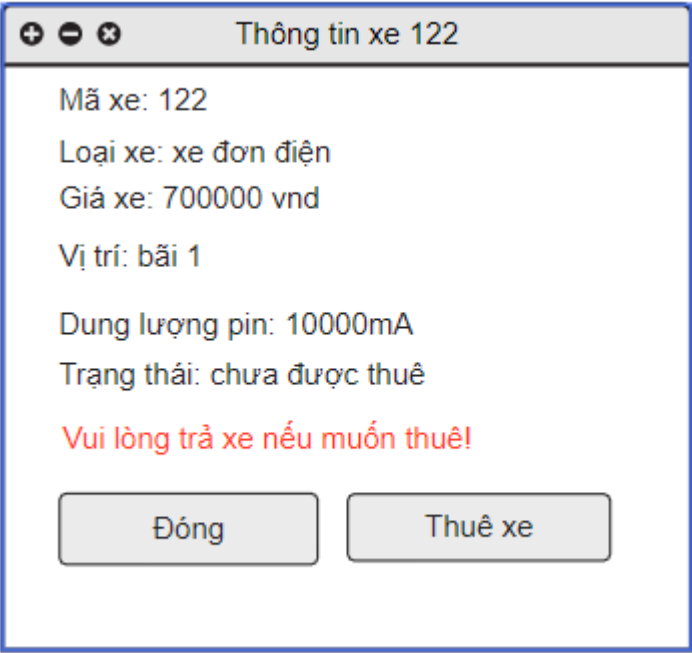
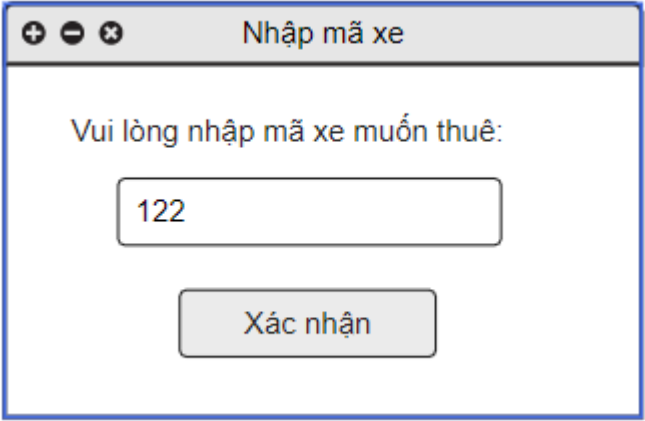
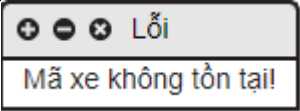
Thông tin bãi xe số 1

Danh sách xe có trong bãi

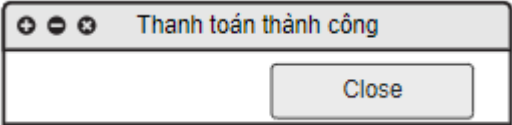
Mã xe	Loại xe	Giá xe	Dung lượng pin	Trạng thái thuê	Chi tiết xe
111	Xe đơn thường	400000 vnd	Xe không có pin	Xe không có pin	<a href="#">Xem</a>
122	Xe đơn điện	700000 vnd	10000mA	Xe không có pin	<a href="#">Xem</a>
133	Xe đôi thường	550000 vnd	Xe không có pin	Xe không có pin	<a href="#">Xem</a>

Đặc tả màn hình	Thông tin xe	23/11/2020	lâm	liên	lan
		Điều khiển	Hoạt động	Chức năng	
		Khu vực thông tin xe	Initial	Hiển thị thông tin xe đã chọn	
		Khu vực hiển thị vui lòng trả xe	Initial	Được hiển thị khi đã có xe được thuê	



		Button Đóng	Click	Hiển thị màn hình trước đó		
		Button Thuê xe	Click	Thuê xe, trả về màn hình thanh toán cọc xe. Hiện thị khi chưa có xe nào được thuê		
Đặc tả màn hình	Nhập mã thuê	23/11/2020	lâm	lâm	liên	
		Điều khiển	Hoạt động	Chức năng		
		Text Nhập mã xe		Nhập mã xe cần thuê		
		Button Xác nhận	Click	Xác nhận thuê xe, Trả về màn hình thông tin xe vừa nhập		
Đặc điểm màn hình	Lỗi mã xe không tồn tại	23/11/2020	lâm	lâm	liên	
		Điều khiển	Hoạt động	Chức năng		
		Khu vực thông báo	Initial	Thông báo mã xe không tồn tại		
Đặc điểm màn hình	Thanh toán cọc xe	23/11/2020	lâm	lâm	liên	

<div><div>Thanh toán</div><div><div>← → ↻</div><div></div></div><div>Thông tin thanh toán cọc xe</div><div><div>Tên chủ thẻ:<div>tên chủ thẻ</div></div><div>Mã thẻ:<div>mã thẻ</div></div><div>CvvCode:<div>cvv code</div></div><div>Ngày hết hạn:<div>ngày hết hạn thẻ</div></div><div>Số tiền giao dịch:<div>Số tiền giao dịch:</div></div><div><div>Chú ý:</div><div><div><div></div><div>Nếu bạn thuê xe vui lòng đặt cọc số tiền bằng 40% giá xe muốn thuê. Tiền cọc sẽ được trả lại vào tài khoản khi bạn trả xe</div></div><div><div></div><div>Nếu bạn trả xe vui lòng thanh toán đủ tổng số tiền trên hóa đơn</div></div></div><div>Nội dung giao dịch:</div><div><div>Hủy</div><div>Thanh toán</div></div></div></div></div>		<div>Điều khiển</div>	<div>Hoạt động</div>	<div>Chức năng</div>	
		<div>Khu vực các thông tin khách hàng</div>	<div>Initial</div>	<div>Hiển thị các trường thuộc tính thông tin khách hàng cần để thuê xe để nhập</div>	
		<div>Text-Các thông tin khách hàng cần để thuê xe</div>	<div>Initial</div>	<div>Nhập vào các thông tin cần để thuê xe</div>	
		<div>Khu vực hiển thị chú ý</div>	<div>Initial</div>	<div>Thông báo về số tiền cần giao dịch</div>	
		<div>Button Thanh toán</div>	<div>Click</div>	<div>Thanh toán hóa đơn thuê xe qua thẻ tín dụng</div>	
		<div>Button Hủy</div>	<div>Click</div>	<div>Hủy thuê xe, trả về màn hình danh sách bãi đậu xe</div>	
<div>Đặc tả màn hình</div>	<div>Lỗi giao dịch</div>	<div>23/11/2020</div>	<div>Lâm</div>	<div>lâm</div>	<div>liên</div>
<div><div><div><div>+</div><div>−</div><div>×</div></div><div>Lỗi giao dịch</div><div></div></div></div>		<div>Điều khiển</div>	<div>Hoạt động</div>	<div>Chức năng</div>	
		<div>Khu vực hiển thị lỗi</div>	<div>Initial</div>	<div>Thông báo thông tin lỗi giao dịch gồm các lỗi:</div>	

				<div>-thẻ không hợp lệ</div> <div>-thẻ không đủ số dư</div> <div>-Internal Server Error</div> <div>-Giao dịch bị nghi ngờ gian lận</div> <div>-Không đủ thông tin giao dịch</div> <div>-Thiếu thông tin version</div> <div>-Amount không hợp lệ</div> <div>-Amount không đủ</div>	
Đặc điểm màn hình	Thanh toán thành công	23/11/2020	lâm	lâm	liên
		Điều khiển	Hoạt động	Chức năng	
		Button Close	Click	Xác nhận xe đã được thuê, trả về màn hình danh sách bãi xe	
Đặc điểm màn hình	Lỗi bãi xe đầy	23/11/2020	lâm	lan	linh
		Điều khiển	Hoạt động	Chức năng	

<div><div>Lỗi</div><div>Bãi xe đầy, vui lòng chọn bãi khác</div></div>		Khu vực hiển thị lỗi	Initial	Được hiển thị khi mà trả xe về bãi đầy từ màn hình danh sách bãi xe	
Đặc điểm màn hình	Hóa đơn	23/11/2020	lâm	lan	linh
<div><div>Hóa đơn 111</div><div><div><div>Hóa đơn</div><div><div>Bãi trả xe: Bãi 1</div><div>Mã xe trả: 122</div><div>Loại xe Xe đơn điện</div><div>Thời gian bắt đầu: 10:00 01/11/2020</div><div>Thời gian kết thúc: 11:00 01/11/2020</div><div>Tổng số tiền: 100.000 VND</div><div>Số tiền đã cọc: 50.000 VND</div></div><div>Thông tin thanh toán trả xe</div><div><div>Tên chủ thẻ:<div>lên chủ thẻ</div></div><div>Mã thẻ:<div>mã thẻ</div></div><div>CvvCode:<div>cvv code</div></div><div>Ngày hết hạn:<div>ngày hết hạn thẻ</div></div><div>Số tiền giao dịch:<div>Số tiền giao dịch:</div></div><div><div>Chú ý:</div><div><div><div>• Nếu bạn thuê xe vui lòng đặt cọc số tiền bằng 40% giá xe muốn thuê. Tiền cọc sẽ được trả lại vào tài khoản khi bạn trả xe</div><div>• Nếu bạn trả xe vui lòng thanh toán đủ tổng số tiền trên hóa đơn</div></div><div>Nội dung giao dịch:<div>Nội dung giao dịch:</div></div></div><div><div>Hủy</div><div>Thanh toán</div></div></div></div></div></div></div>		Điều khiển	Hoạt động	Chức năng	
		Khu vực hiển thị	Initial	Hiển thị thông tin hóa đơn xe đã thuê	
		Khu vực các thông tin khách hàng	Initial	Hiển thị các trường thuộc tính thông tin khách hàng cần để thuê xe để nhập	
		Text-Các thông tin khách hàng cần để trả xe	Intial	Nhập vào các thông tin cần để trả xe	
		Khu vực chú ý	Initial	Thông báo về số tiền cần giao dịch	
		Button Hủy	Click	Trả về màn hình danh sách bãi xe	
		Button Thanh toán	Click	Thanh toán hóa đơn	

## 4.2 Mô hình dữ liệu

Mô hình hóa dữ liệu là quá trình tạo ra một mô hình dữ liệu. Khi tạo một mô hình dữ liệu, trước tiên phải xác định dữ liệu, các thuộc tính và mối quan hệ của nó với dữ liệu khác và xác định các ràng buộc hoặc giới hạn đối với dữ liệu

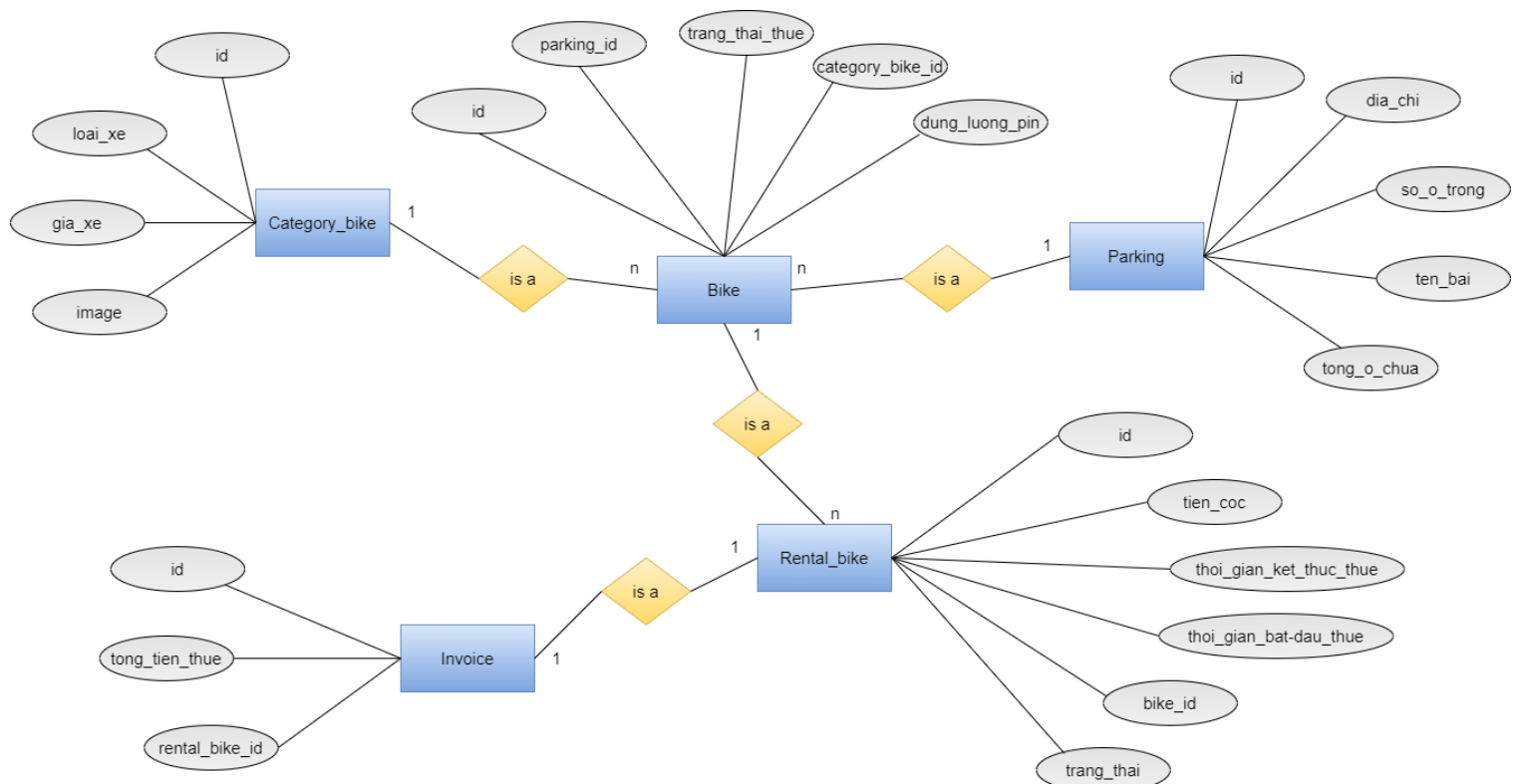
Mô hình dữ liệu xác định dữ liệu, các thuộc tính dữ liệu và các mối quan hệ hoặc liên kết với dữ liệu khác. Mô hình dữ liệu cung cấp một cái nhìn tổng quát, do người dùng định nghĩa về dữ liệu đại diện cho kịch bản và dữ liệu nghiệp vụ.

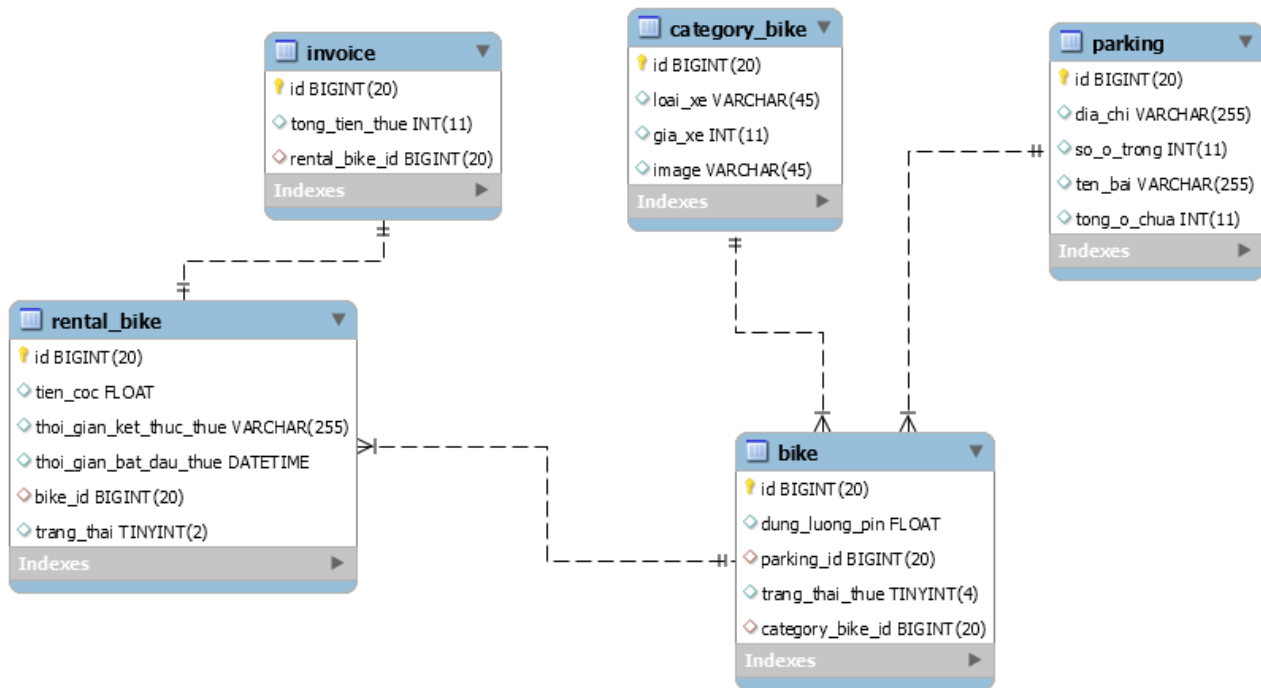
### 4.2.1 Mô hình hóa dữ liệu khái niệm

Mô hình dữ liệu khái niệm, còn được gọi là mô hình miền (Domain models), thiết lập các khái niệm và ngữ nghĩa cơ bản của một miền nhất định đối với nhiều đối tượng của các bên liên quan.

Các mô hình khái niệm được xây dựng dựa trên Kiến trúc tổng thể bằng cách sử dụng các mô hình Mối quan hệ thực thể hoặc Lớp UML.

### 4.2.2 Thiết kế cơ sở dữ liệu

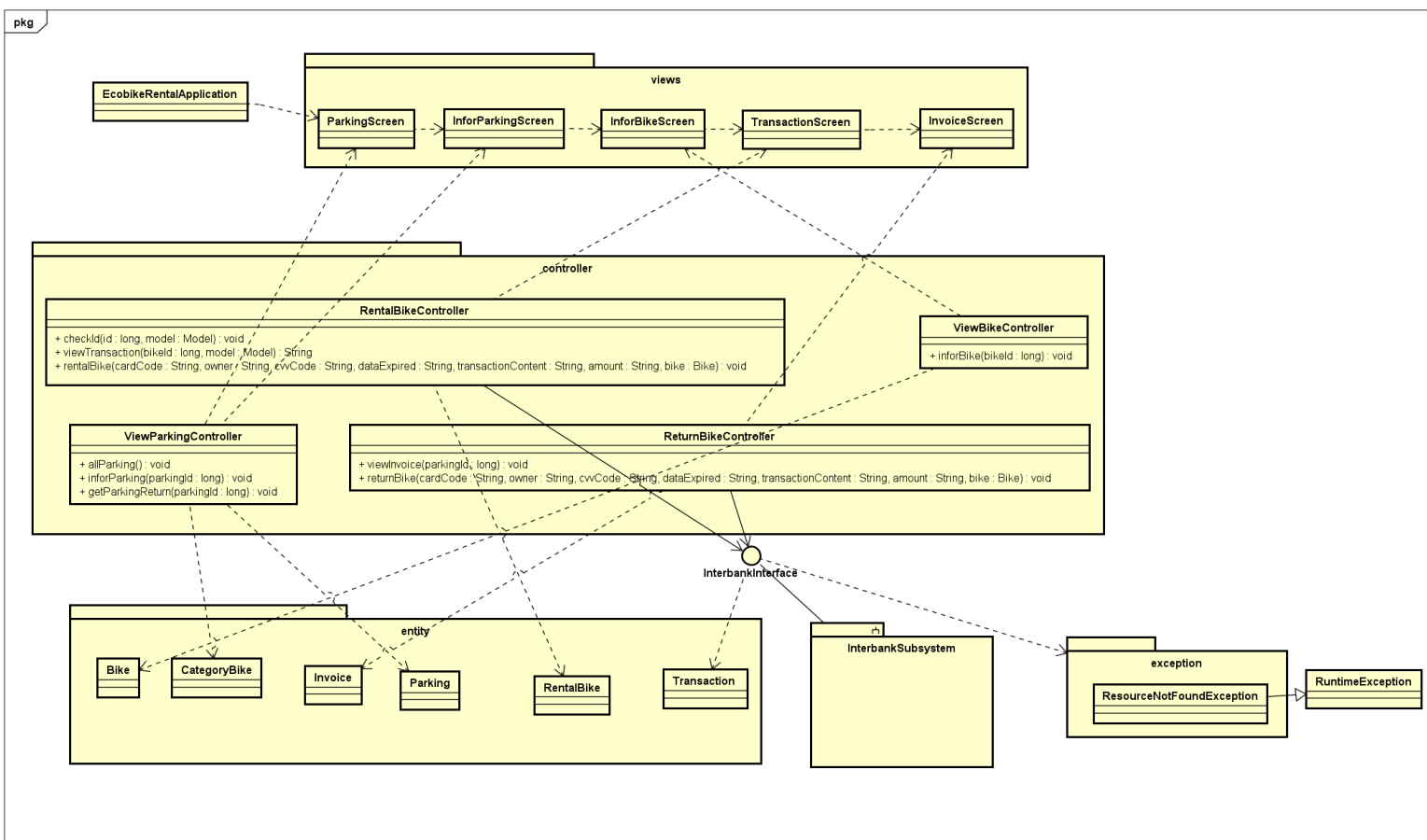




## 4.3 Non-Database Management System Files

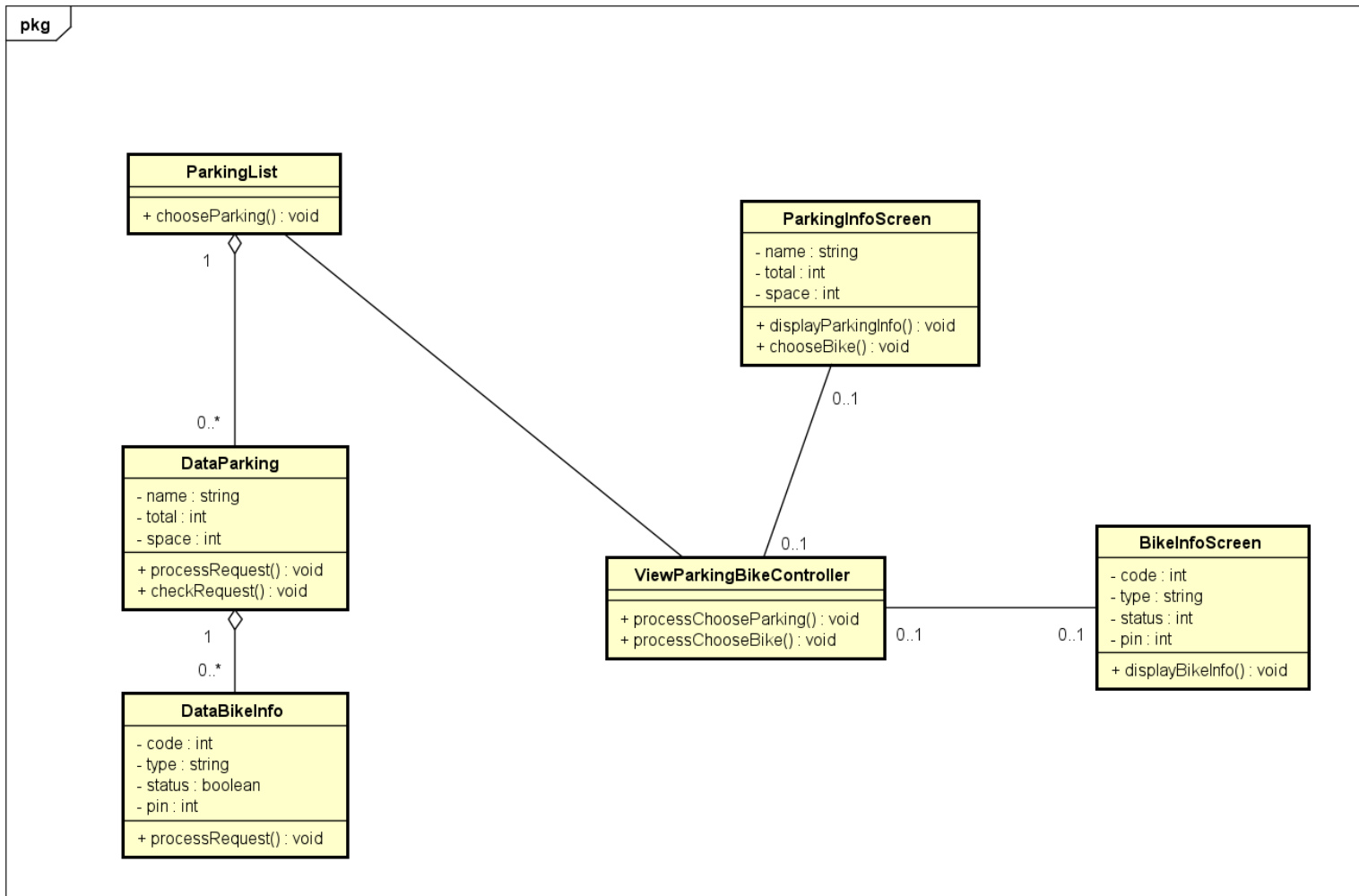
### 4.4 Thiết kế lớp

#### 4.4.1 Biểu đồ lớp tổng quát

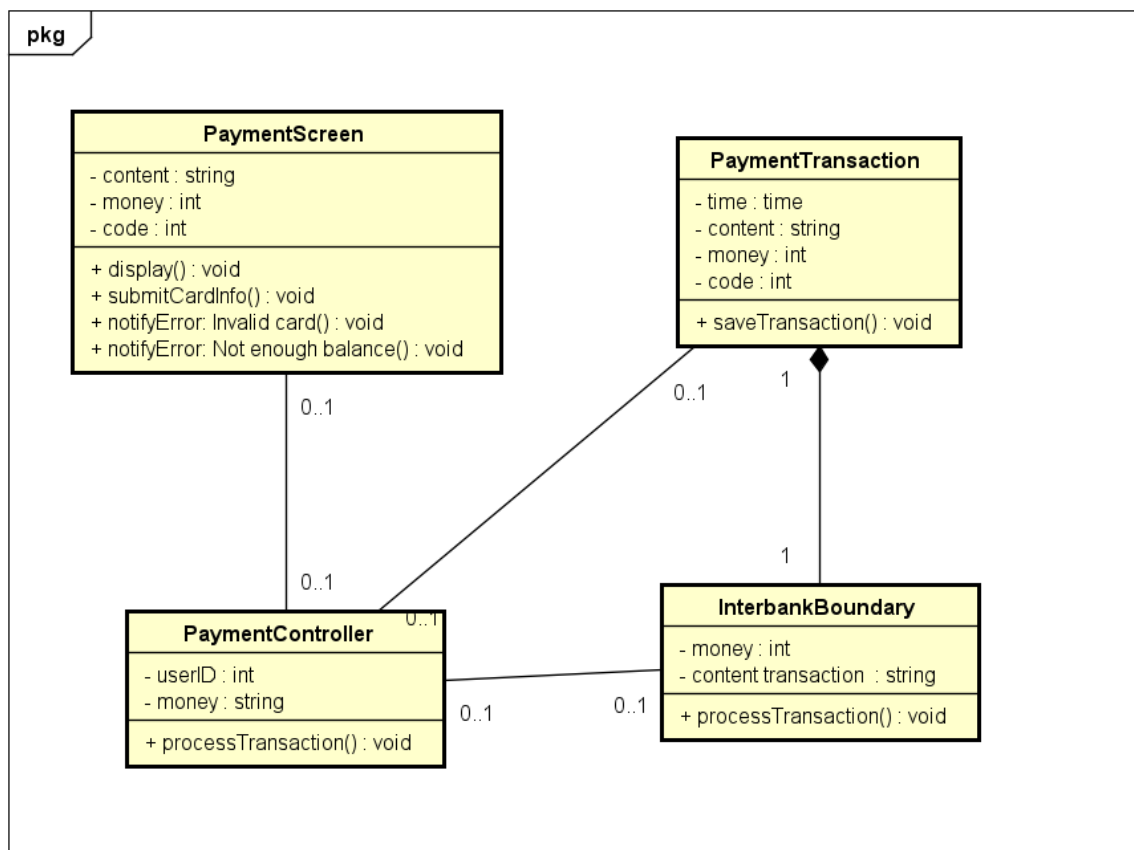


## 4.4.2 Biểu đồ lớp

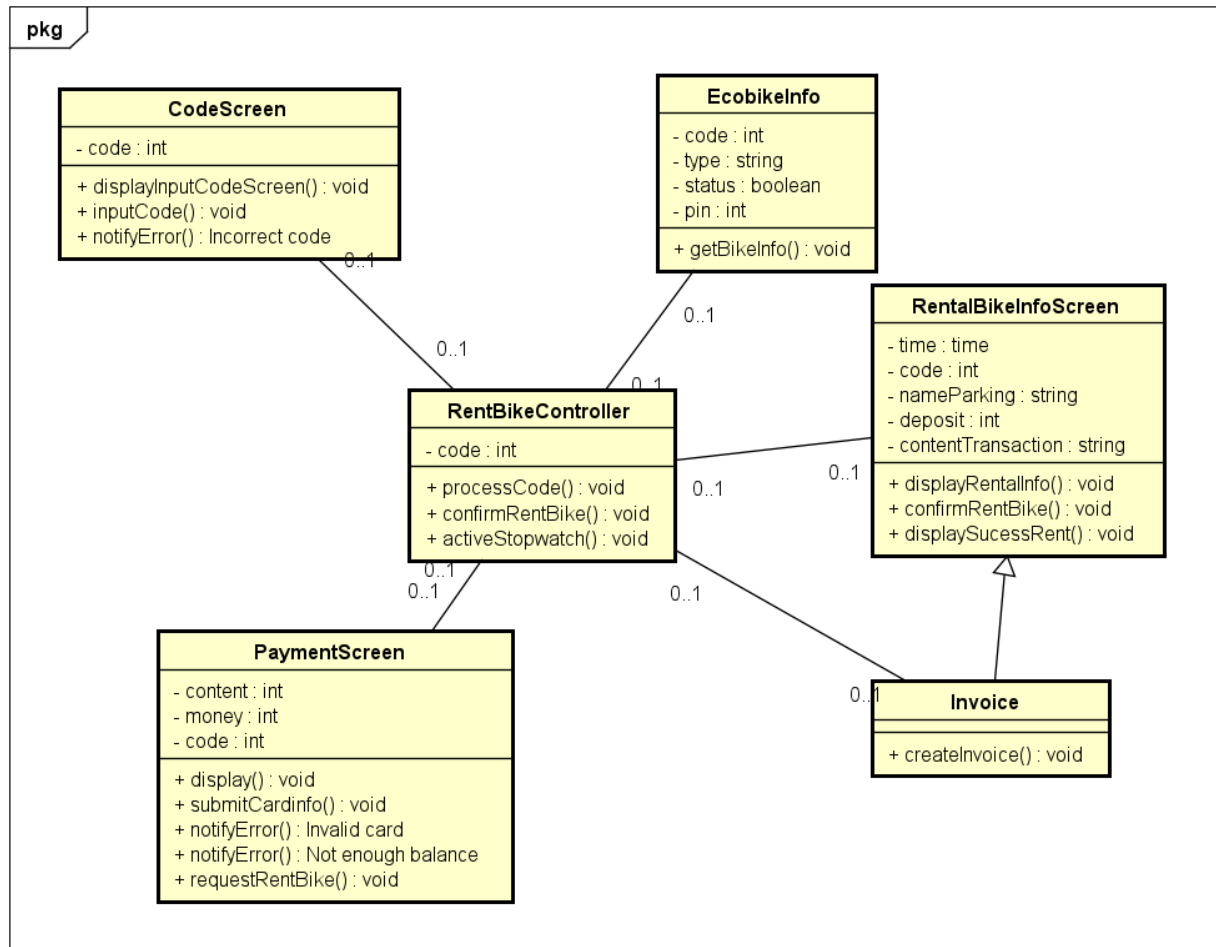
- Xem thông tin bãi xe loại xe



- Thanh toán



- Thuê xe



#### 4.4.3 Thiết kế lớp

```

public class Bike {
    /** id xe */
    @Id
    @GeneratedValue (strategy = GenerationType.IDENTITY)
    private Long id;
    /** thuộc tính dung lượng pin xe với xe dùng điện */
    @Column (name = "dung_luong_pin")
    private float energy;
    /** bãi xe nơi xe được đỗ */
    @ManyToOne (fetch = FetchType.LAZY, cascade = CascadeType.ALL)
    @JoinColumn (name = "parking_id")
    private Parking parking;
    /** loại xe */
    @ManyToOne (fetch = FetchType.LAZY, cascade = CascadeType.ALL)
    @JoinColumn (name = "category_bike_id")
    private CategoryBike categoryBike;
    /** trạng thái xe được thuê hay chưa ( true là đã thuê, false là chưa thuê) */
    @Column (name = "trang_thai_thue")
    private boolean status;
    /** thông tin thuê xe */
    @OneToMany (mappedBy = "bike", fetch = FetchType.LAZY, cascade = CascadeType.ALL)
    @Fetch (FetchMode.SUBSELECT)
    @JsonIgnore
    private Set<RentalBike> rentalBikes;
}
  
```



```

public class CategoryBike {
    /** id loại xe */
    @Id
    @GeneratedValue (strategy = GenerationType.IDENTITY)
    private Long id;
    /** tên loại xe */
    @Column (name = "loại_xe")
    private String categoryBike;
    /** giá tương ứng với từng loại xe */
    @Column (name = "gia_xe")
    private int priceBike;
    /** ảnh loại xe */
    @Column(name="image")
    private String image;
    /** danh sách các xe có cùng loại */
    @OneToMany(mappedBy = "categoryBike", fetch = FetchType.LAZY, cascade = CascadeType.ALL)
    @JsonIgnore
    private Set<Bike> bike;
}

```

```

public class Invoice {
    /** id hóa đơn */
    @Id
    @GeneratedValue (strategy = GenerationType.IDENTITY)
    private Long id;
    /** thông tin thuê xe để tạo hóa đơn tương ứng */
    @OneToOne
    @JoinColumn(name = "rental_bike_id")
    private RentalBike rentalBike;
    /** tổng số tiền thuê xe sau khi tính toán */
    @Column (name = "tong_tien_thue")
    private long rentMoney;
}

```

```

public class Parking {
    /** id bãi xe */
    @Id
    @GeneratedValue (strategy = GenerationType.IDENTITY)
    private Long id;
    /** tên bãi */
    @Column (name = "ten_bai")
    private String nameParking;
    /** địa chỉ bãi */
    @Column (name = "dia_chi")
    private String address;
    /** tổng số ô chứa trong bãi */
    @Column (name = "tong_o_chua")
    private int totalCellContains;
    /** số ô trống trong bãi */
    @Column (name = "so_o_trong")
    private int cellEmpty;
    /** danh sách xe trong bãi */
    @OneToMany (mappedBy = "parking", fetch = FetchType.LAZY, cascade = CascadeType.ALL)
    @Fetch (FetchMode.SUBSELECT)
    @JsonIgnore
    private Set<Bike> bike;
}

```

```

public class RentalBike {
    /** id thông tin thuê xe */
    @Id
    @GeneratedValue (strategy = GenerationType.IDENTITY)
    private Long id;
    /** thời gian xe bắt đầu thuê */
    @Column (name = "thoi_gian_bat_dau_thue")
    @Temporal (TemporalType.TIMESTAMP)
    @CreatedDate
    private Date timeStart;
    /** thời gian xe kết thúc thuê */
    @Column (name = "thoi_gian_ket_thuc_thue")
    @Temporal (TemporalType.TIMESTAMP)
    private Date timeEnd;
    /** tiền đặt cọc khi thuê xe */
    @Column (name = "tien_coc")
    private int deposits;
    /** trạng thái xe thuê đã được trả hay chưa (true: là đã trả, false: là chưa trả) */
    @Column (name = "trang_thai")
    private boolean status;
    /** xe được thuê */
    @ManyToOne (fetch = FetchType.LAZY, cascade = CascadeType.ALL)
    @JoinColumn (name = "bike_id")
    private Bike bike;
}

```

```

public class Transaction {
    /** mã aoi sử dụng */
    private String command;
    /** chủ tài khoản */
    private String owner;
    /** mã thẻ */
    private String cardCode;
    /** mã cvv */
    private String cvvCode;
    /** ngày hết hạn */
    private String dateExpired;
    /** nội dung giao dịch */
    private String transactionContent;
    /** số tiền giao dịch */
    private String amount;
    /** thời điểm tạo giao dịch */
    private Date createdAt;
}

```

## **5. Cân nhắc kiến trúc**

### **5.1 Mục tiêu và nguyên tắc**

#### **5.1.1 Mục tiêu phần mềm**

Giao diện dễ sử dụng, tốc độ xử lý dữ liệu nhanh, các module độc lập nhau, dễ phát triển cũng như sửa chữa, bảo trì hệ thống

#### **5.1.2 Nguyên tắc**

### **5.2 Chiến lược kiến trúc**

- Tái sử dụng các thành phần hiện có để phát triển các thành phần mới
- Xây dựng các mô đun tương đối độc lập để có thể dễ dàng mở rộng phần mềm
- Tối ưu việc sử dụng tài nguyên
- Đồng bộ hóa

### **5.3 Coupling and Cohesion**

#### **5.3.1 Coupling**

Để có một bản thiết kế tốt thì cần phải thiết kế sao cho coupling lỏng để khi có sự thay đổi ở một module thì ảnh hưởng ít nhất có thể đến các module khác. Coupling càng lỏng lẻo, càng tốt

- Content coupling: Xảy ra khi 1 module sử dụng code của 1 module khác, điều này sẽ không đúng với những khái niệm thiết kế cơ bản là đóng gói thông tin
- Common coupling: Xảy ra khi các module cùng truy cập vào cùng data global, và điều này có thể dẫn đến việc không thể kiểm soát được lỗi và sẽ xảy ra những lỗi không mong muốn
- Control coupling: là khi 1 module đang control luồng hoạt động của 1 module khác thông qua các tham số đầu vào, và nó có thể dẫn đến 1 vài trường hợp không mong muốn khi các tham số truyền vào không được đúng
- Stamp Coupling: xảy ra khi các module chỉ sử dụng các thành phần cấu trúc dữ liệu được chia sẻ với nhau, và trong trường hợp này, khi thay đổi 1 thuộc tính của 1 module có thể sẽ ảnh hưởng đến các module khác đang phụ thuộc vào nó
- Data coupling: xảy ra khi 2 hoặc nhiều module độc lập với nhau, tương tác, chia sẻ tài nguyên cho nhau và thông thường là qua tham số đầu vào

=>Kết luận: Do quá trình code mỗi bạn thực hiện một phần riêng lẻ nên sự liên kết giữa các module khá lỏng lẻo đạt đến datacoupling.

### 5.3.2 Conhesion

- Coincidental: Sự gắn kết ngẫu nhiên là khi các phần của mô-đun được nhóm lại một cách tùy tiện; mối quan hệ duy nhất giữa các bộ phận là họ đã được nhóm lại với nhau.
- Logical: Sự gắn kết logic là khi các phần của mô-đun được nhóm lại vì chúng được phân loại logic để làm việc tương tự, ngay cả khi chúng khác nhau theo bản chất.
- Temporal: Sự gắn kết thời gian là khi các phần của mô-đun được nhóm lại khi chúng được xử lý – các phần được xử lý tại một thời điểm cụ thể trong quá trình thực hiện chương trình
- Procedural: Sự gắn kết theo thủ tục là khi các phần của mô-đun được nhóm lại bởi vì chúng luôn tuân theo một trình tự thực thi nhất định
- Communicational: Sự liên kết giao tiếp là khi các phần của mô-đun được nhóm lại vì chúng hoạt động trên cùng một dữ liệu
- Sequential: Sự liên kết tuần tự là khi các phần của một mô-đun được nhóm lại bởi vì đầu ra từ một phần là đầu vào đến một phần khác giống như một dây chuyền lắp ráp
- Functional: Sự gắn kết chức năng là khi các phần của mô-đun được nhóm lại vì tất cả chúng đóng góp vào một nhiệm vụ duy nhất được định nghĩa rõ ràng của mô-đun

=>Kết luận:

- Hầu hết các class đều đạt mức Functional vì nó đều hướng đến một nhiệm vụ duy nhất.

### 5.4 Nguyên tắc thiết kế

Tuân theo nguyên tắc thiết kế SOLID:

- Single Responsibility: Một lớp chỉ chịu trách nhiệm một nhiệm vụ cụ thể nào đó.
- Open/Closed: Khi thêm mới chức năng, không sửa trực tiếp nội dung trên class đã viết mà tạo một class mới extends từ class đã có.

=> chưa đạt ở lớp Amount sẽ phải sửa nếu thêm loại xe khác với mức giá tính khác

- Liskov Substitution: class con có thể thay thế cho lớp cha
- Interface Segregation: tách nhỏ interface thành các interface con thực hiện các mục đích cụ thể.