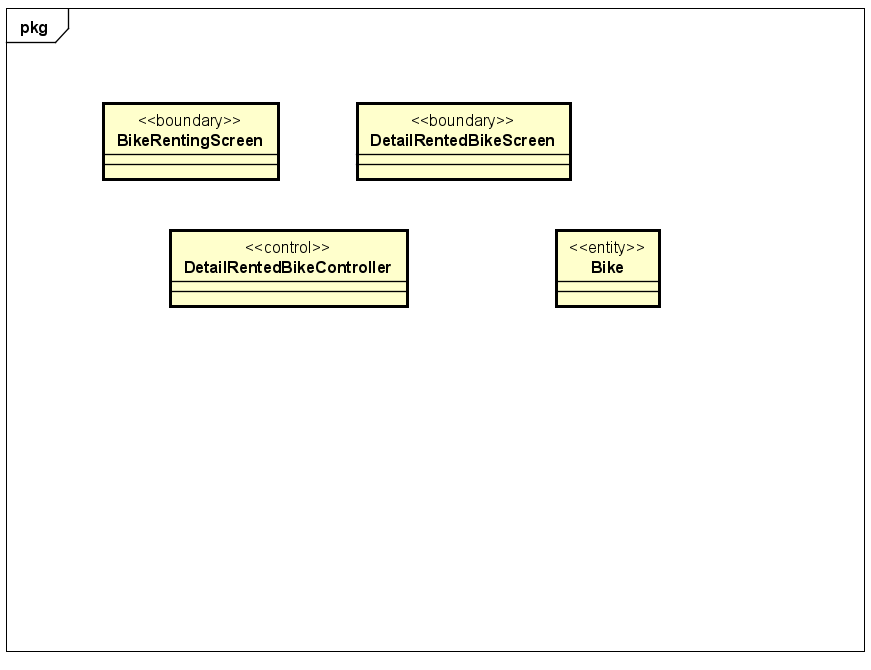
**NHÓM 16**

**I.THIẾT KẾ KIẾN TRÚC**

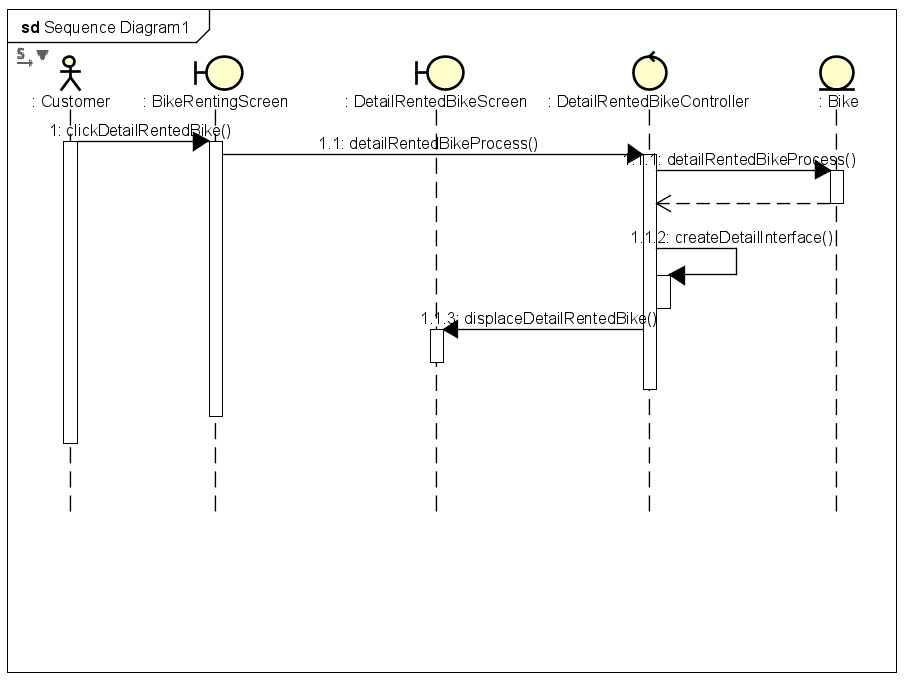
## 1. Xem thông tin xe đang thuê

### a) Phân tích lớp



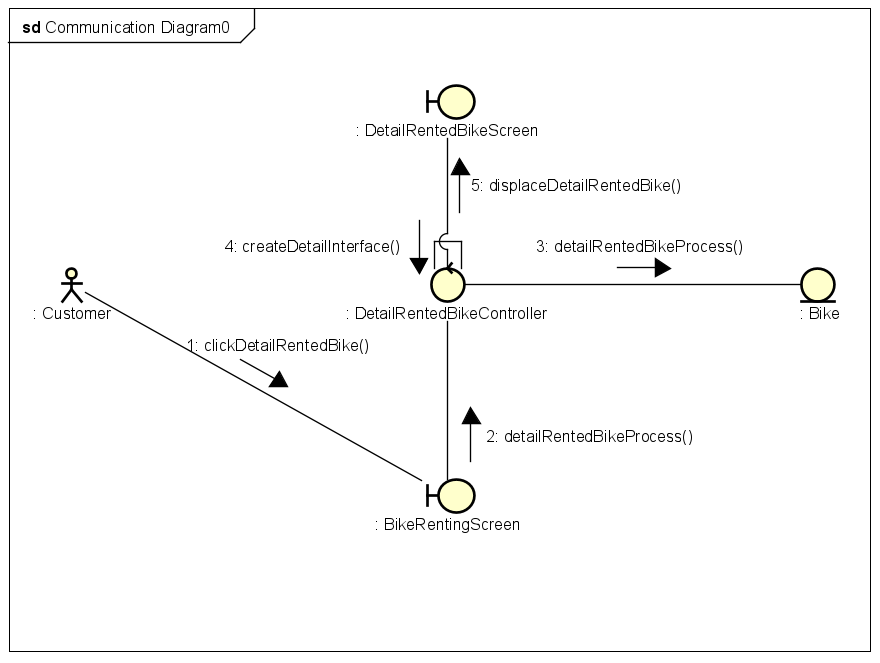
b) Phân phối hành vi trong use case tới các lớp

### Biểu đồ trình tự

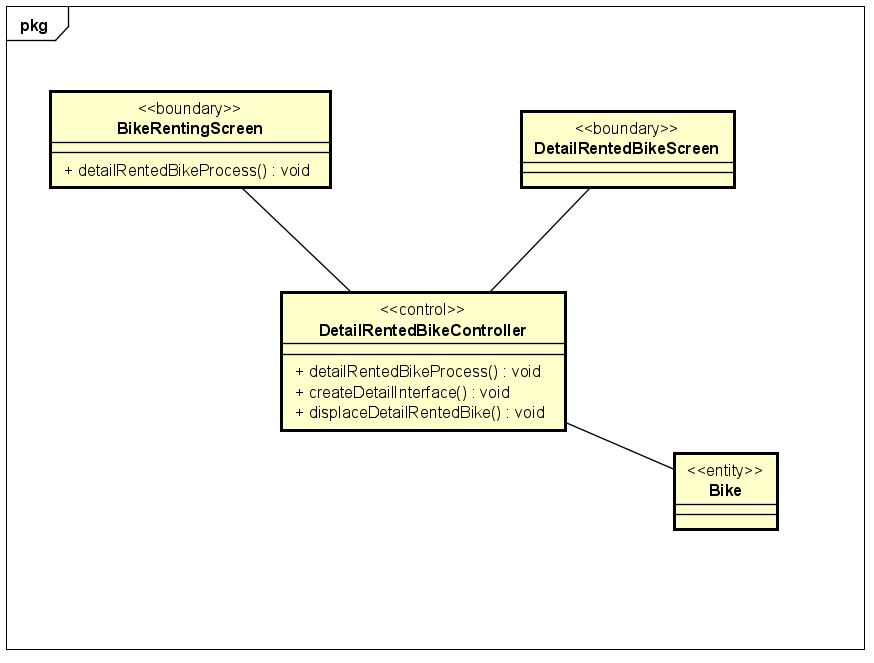


### 

### Biểu đồ hoạt động

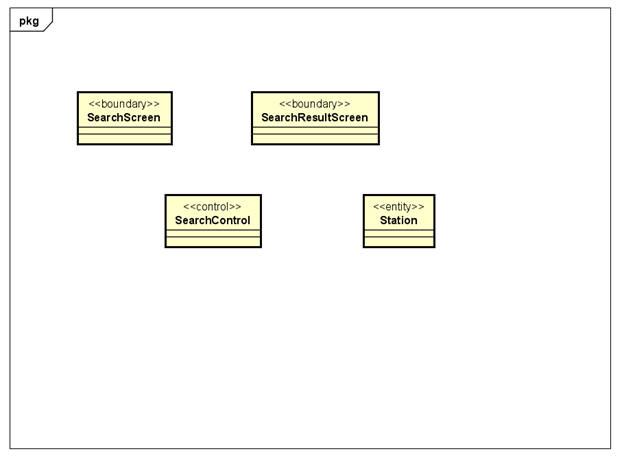


### Biểu đồ phân tích lớp



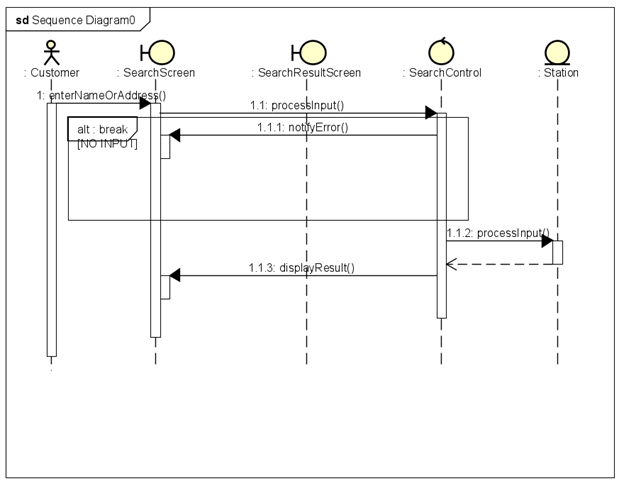
## 2. Tìm kiếm bãi xe

### Phân tích lớp



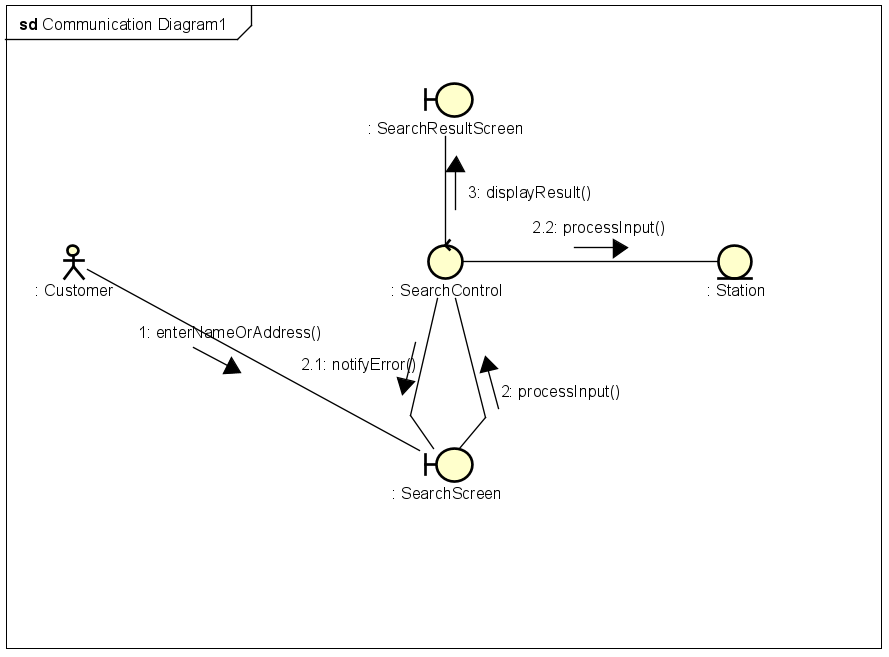
### Phân phối hành vi trong use case tới các lớp.

### Biểu đồ trình tự

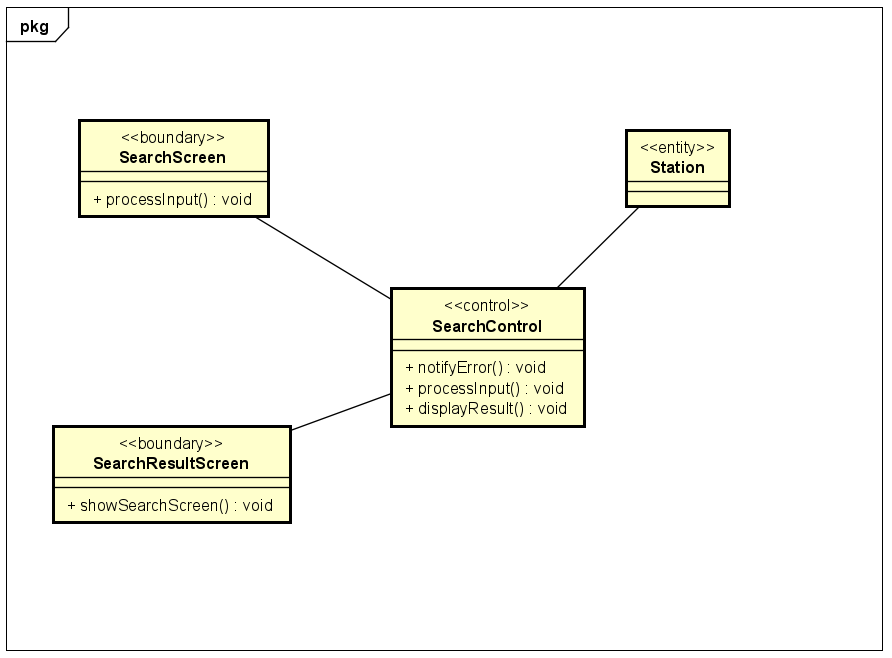


### 

### Biểu đồ hoạt động

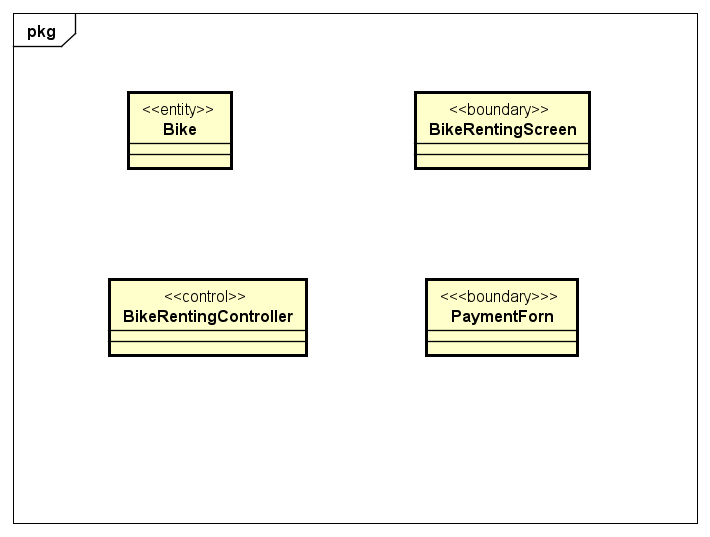


### Biểu đồ lớp phân tích



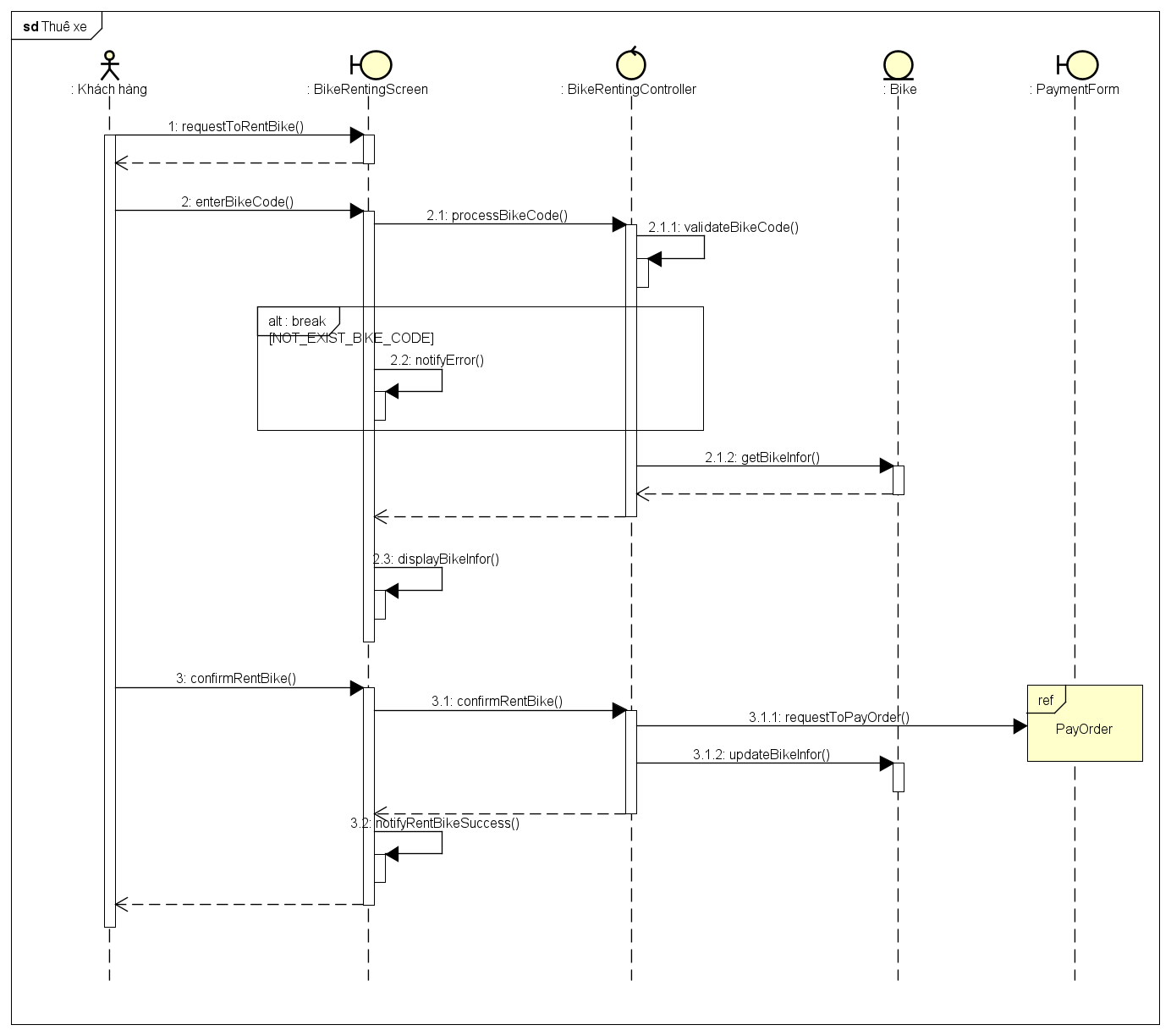
## 3. Thuê xe

### Phân tích lớp

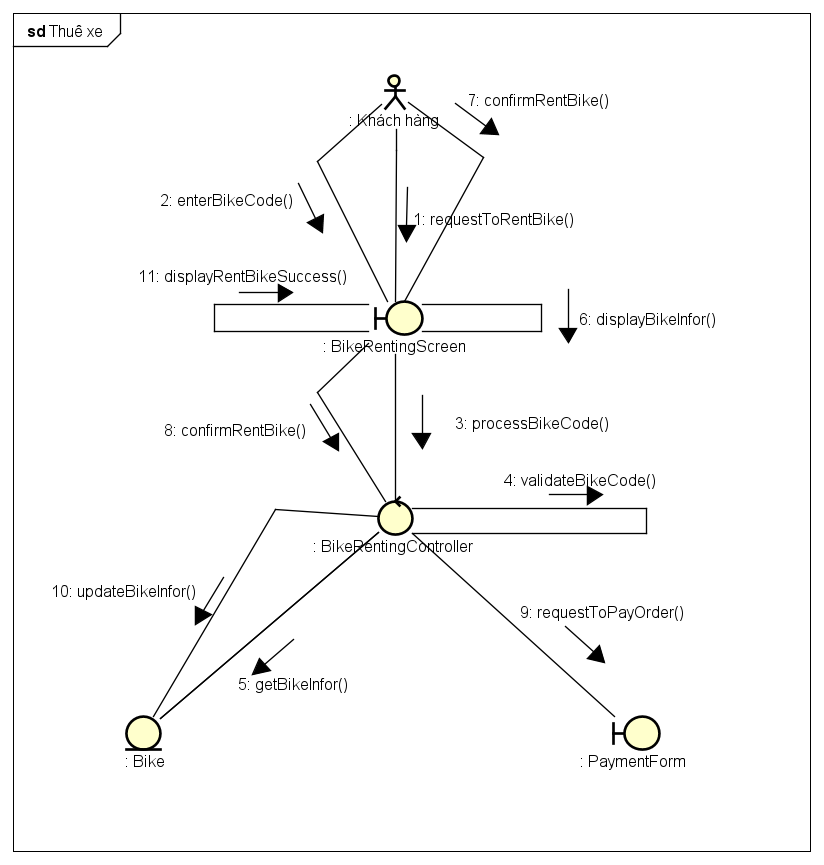


### Phân phối hành vi trong use case tới các lớp.

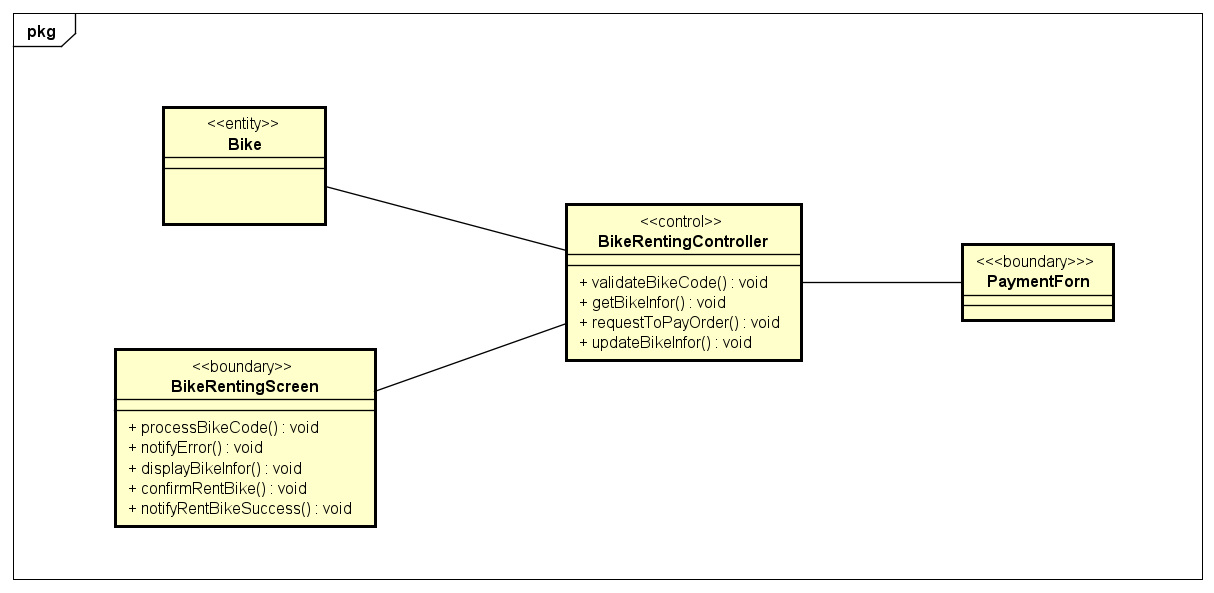
### Biểu đồ trình tự



### Biểu đồ giao tiếp

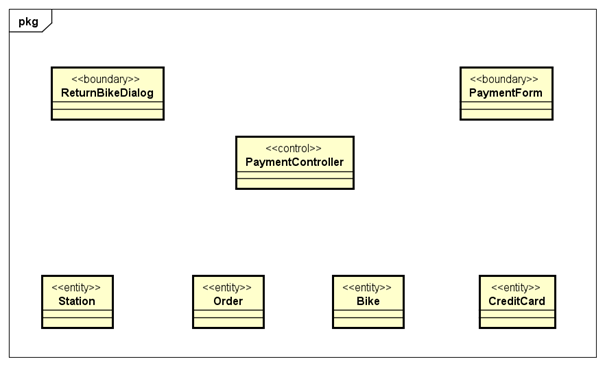


1. Biểu đồ lớp phân tích



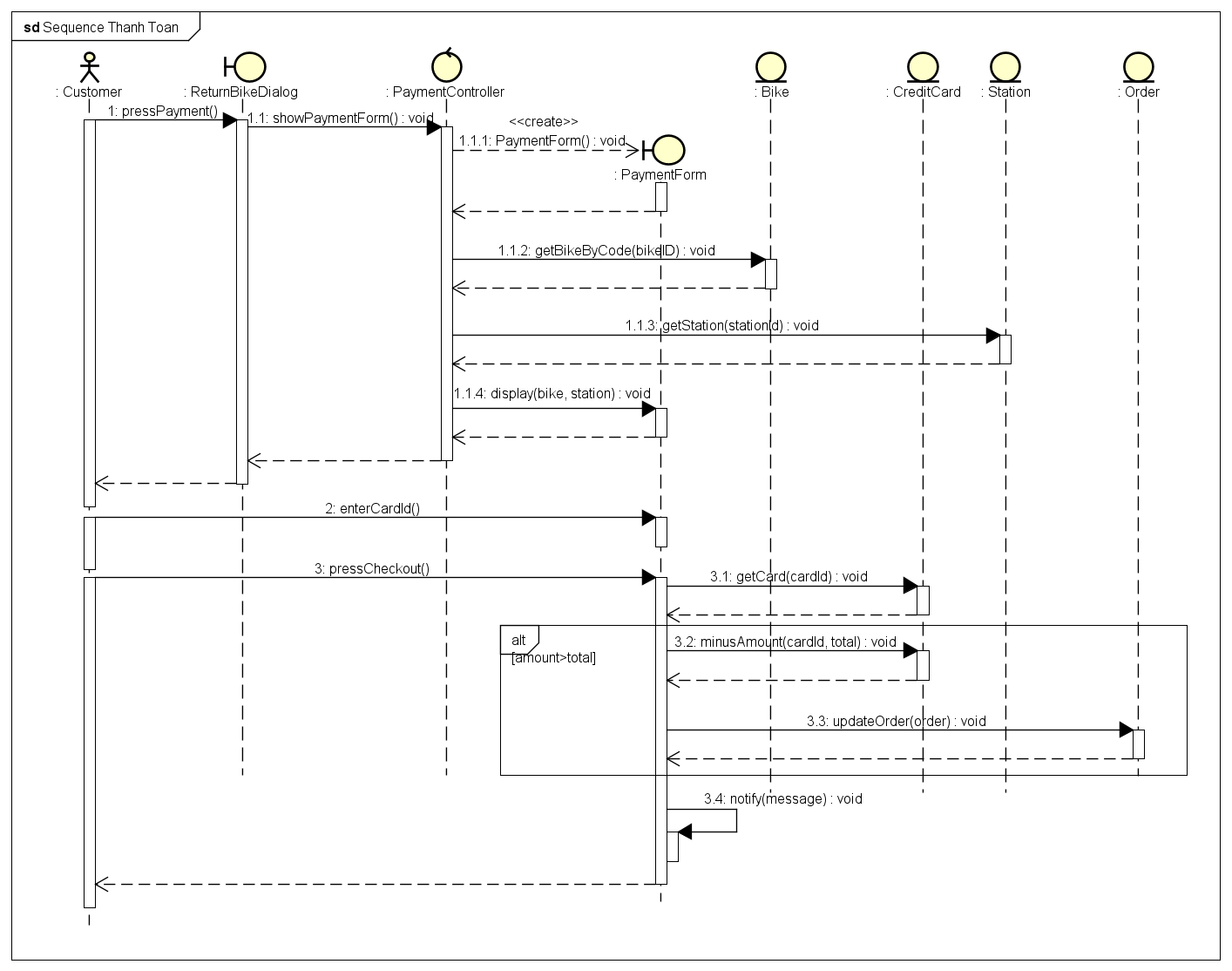
## 4. Thanh Toán

### Phân tích lớp

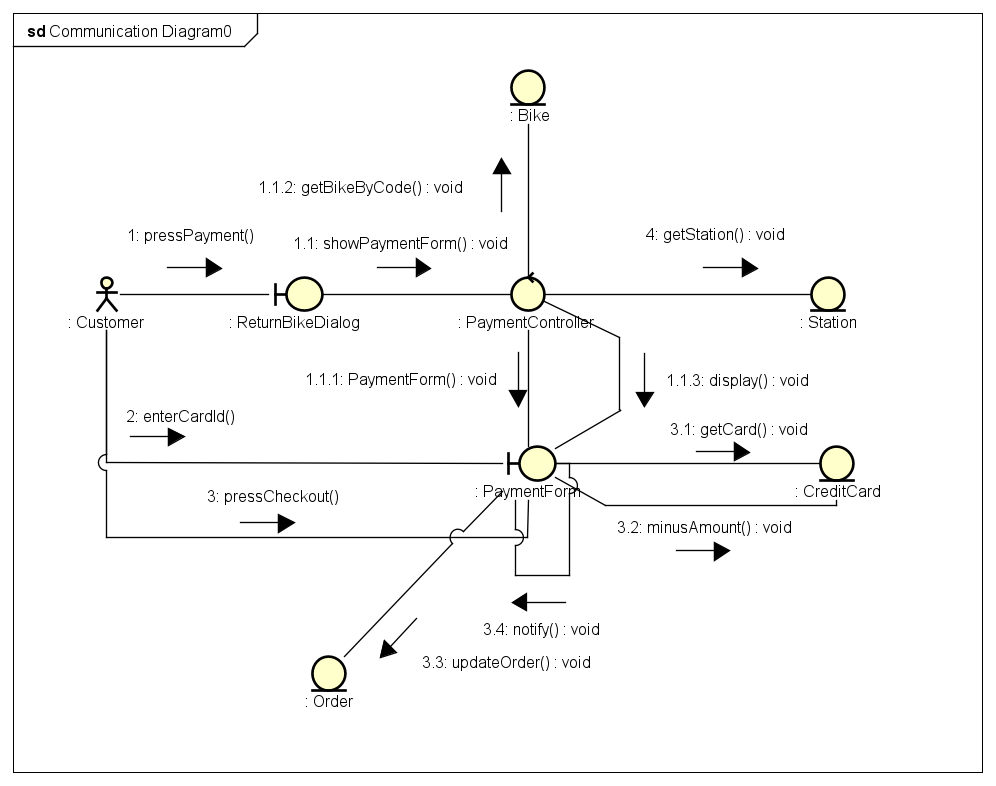


### Phân phối hành vi trong use case tới các lớp.

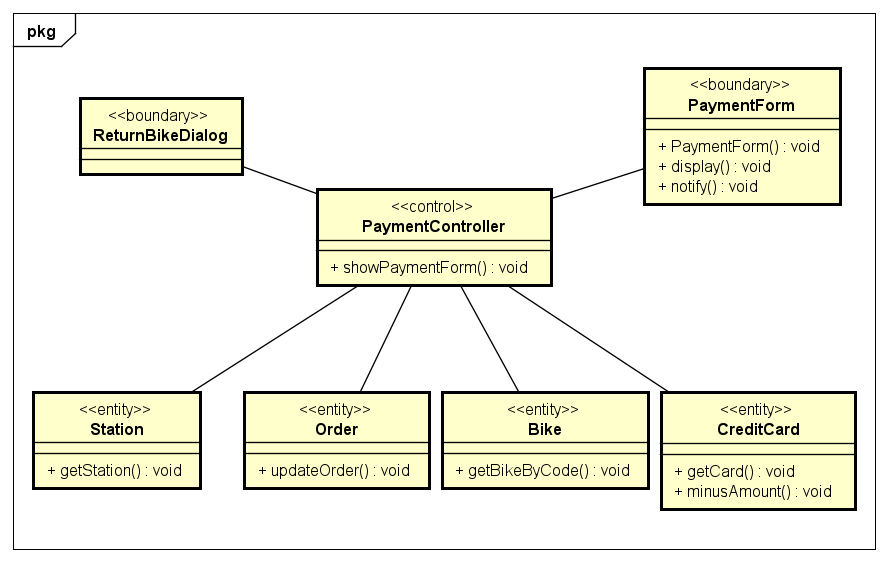
* Biểu đồ trình tự

****

* Biểu đồ giao tiếp

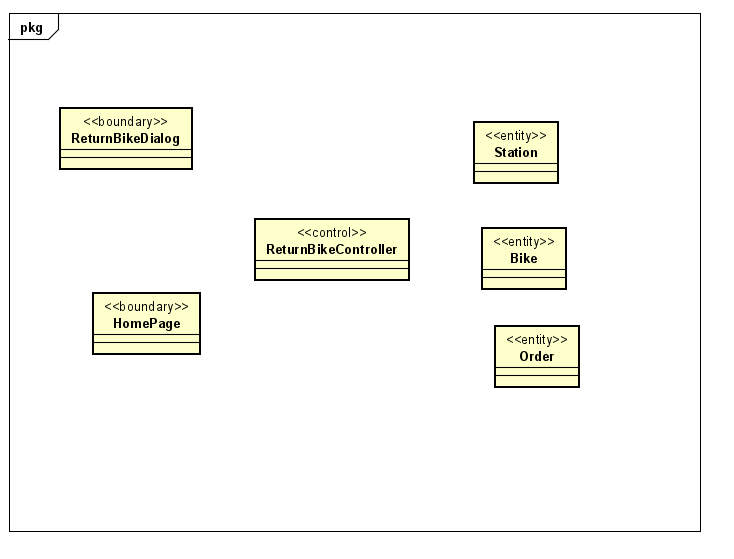
****

### Biểu đồ lớp phân tích



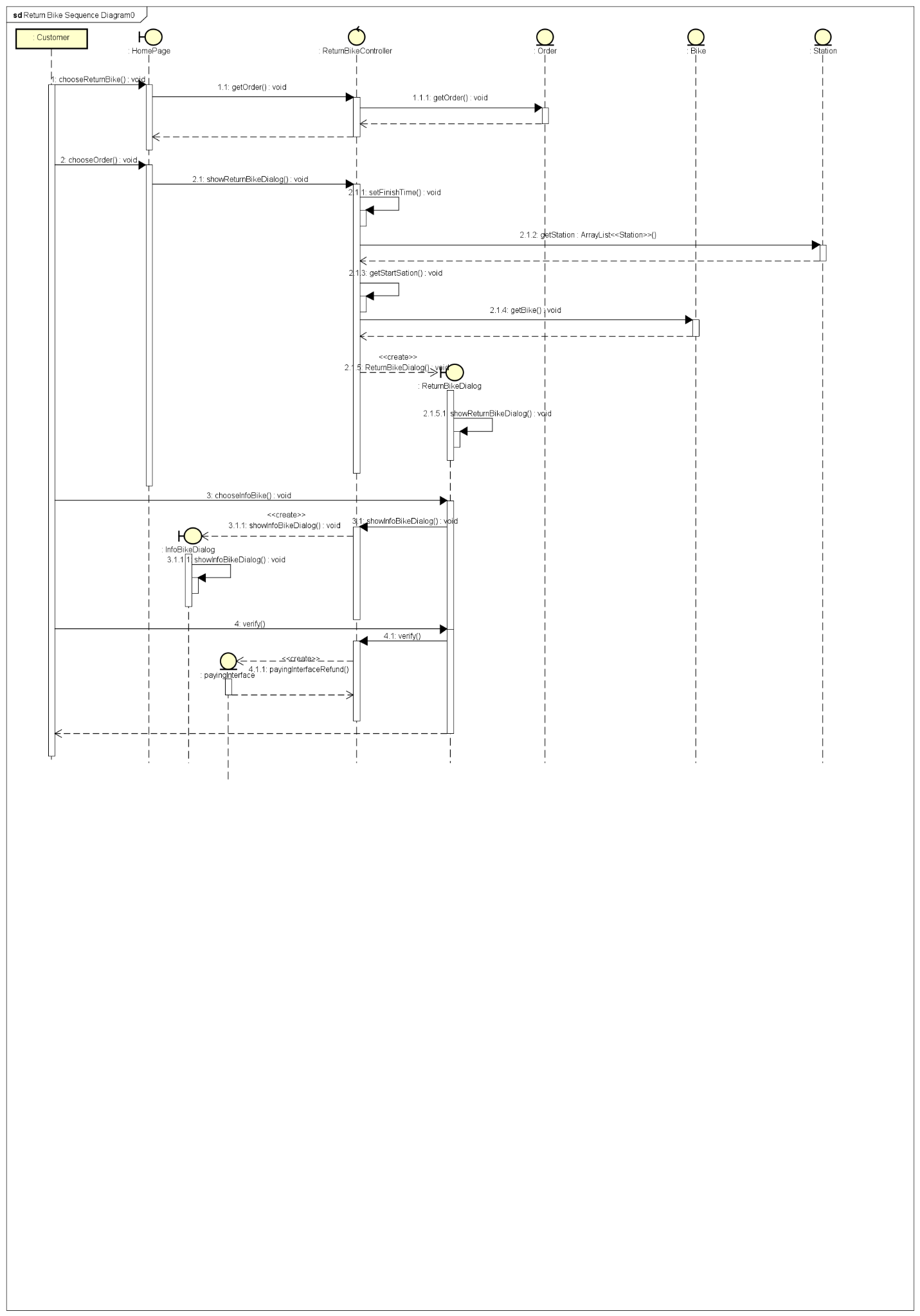
## 5. Trả xe

### Phân tích lớp

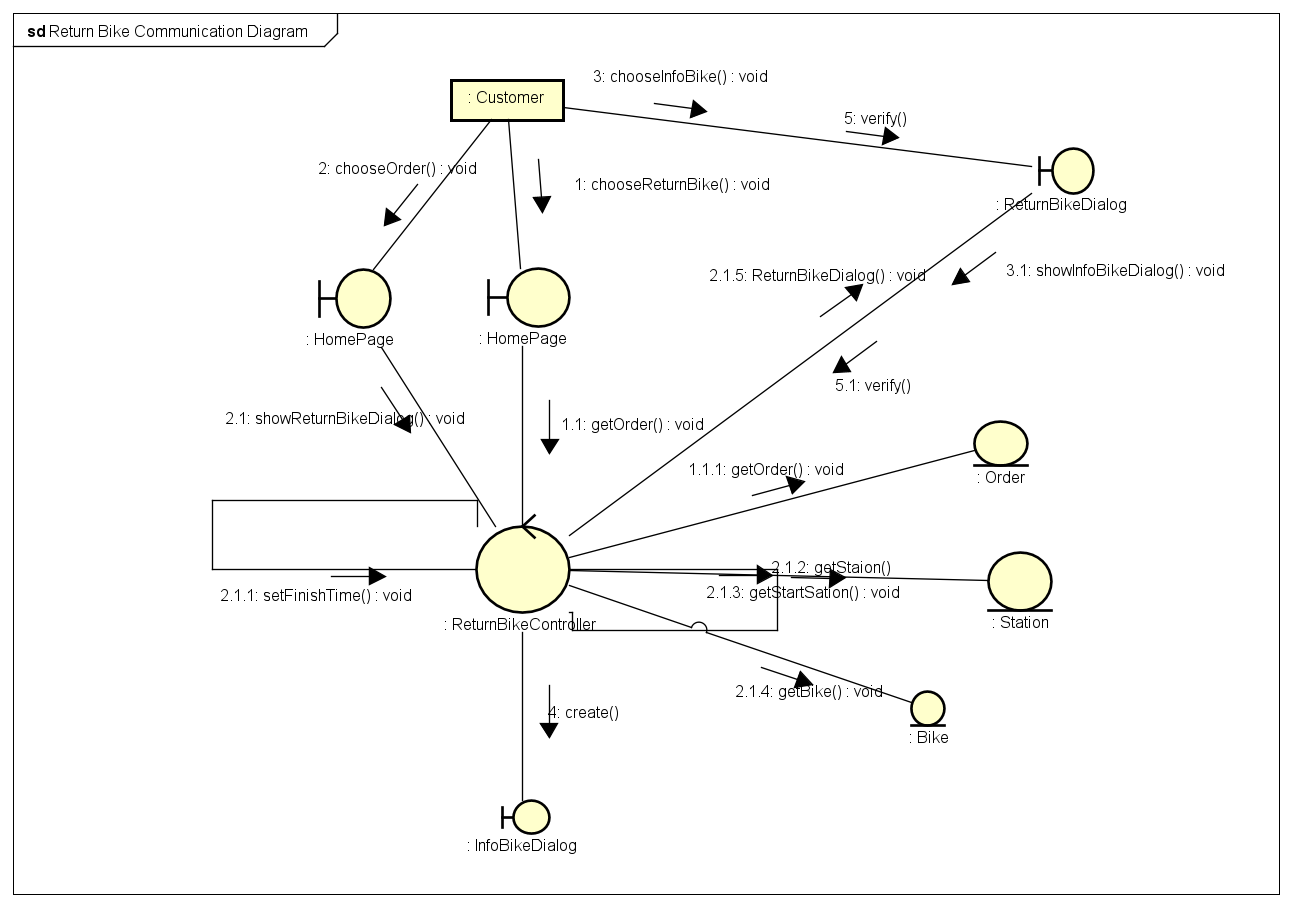


### Phân phối hành vi trong use case tới các lớp.

### Biểu đồ trình tự

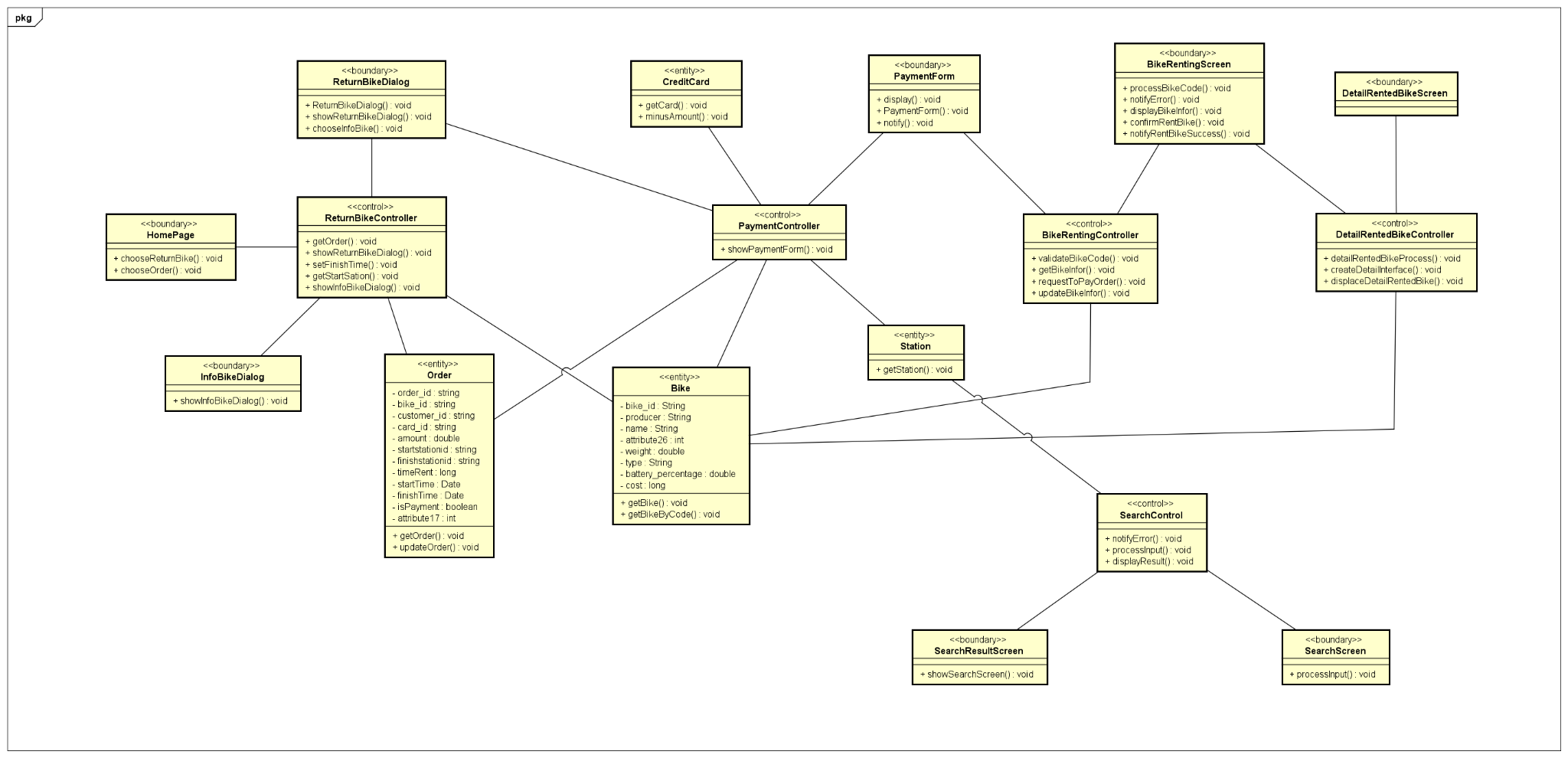


### Biểu đồ hoạt động



### Biểu đồ lớp phân tích

## 6. Biểu đồ lớp phân tích tổng hợp



**II. THIẾT KẾ LỚP VÀ MÔ HÌNH HÓA DỮ LIỆU**

### **1. Thiết kế lớp (Class Design)**

### **1.1. Bước đầu tạo các lớp thiết kế**

Trong phần này, chúng ta sẽ ánh xạ các thành phần thiết kế (design elements, ví dụ: lớp – class, nhóm các lớp – group of classes, gói – package, subsystem) từ các lớp phân tích (analysis classes). Mỗi lớp thiết kế nên chỉ phục vụ tốt một mục đích duy nhất. Chúng ta sẽ xác định các lớp thiết kế dựa vào biểu đồ lớp kiến trúc và khuôn mẫu (stereotype) của lớp đấy. Lưu ý rằng chúng ta chưa ứng dụng các mẫu thiết kế (design patterns) trong bài thực hành này.

##### **a)** **Thiết kế lớp boundary**

###### **Lớp boundary: Giao diện người dùng (User interface)**

Trong Case study, chúng ta sử dụng JavaFX để xây dựng giao diện. Do đó, từ góc nhìn kiến trúc, mỗi lớp boundary giao diện người dùng tương đương với lớp thiết kế phụ trách xử lý sự kiện hoặc hành động của người dùng (được bắt bởi FXML tương ứng). Trong JavaFX, mặc dù các lớp thiết kế này thường được gọi là “controller” của các tệp FXML, chúng không thực sự đóng vai trò như lớp control trong UML. Do vậy, đa số các lớp xử lý sự kiện hiện tại đã khá đơn giản, và ánh xạ là 1-1.

###### **Lớp boundary: Giao diện hệ thống (System/device boundary)**

Trong bài thực hành trước, chúng ta đã “evolve”/ánh xạ lớp boundary cho liên ngân hàng Interbank thành một subsystem. Tuy nhiên, subsystem này thiết kế chưa được tốt: InterbankSubsystemController quá phức tạp, đồng thời một phần InterbankBoundary vẫn có thể tái sử dụng. Hiện tại trong mô tả của Case Study, chỉ có nhắc đến một hệ thống thông tin web bên ngoài (Interbank), nhưng thực tế có rất nhiều hệ thống như thế mà cần giao tiếp với REST APIs sử dụng các phương thức HTTP. Trong khi các hệ thống này có giao thức kết nối giống nhau, một trong số đó có thể là một hệ thống mà AIMS Software cần giao tiếp trong tương lai. Vì vậy, vì mục đích tái sử dụng, chúng ta có thể cần một lớp mới, ví dụ là API, để phụ trách giao tiếp API qua phương thức HTTP GET và HTTP POST. Ngoài ra, chúng ta sẽ nhìn nhận các vấn đề ở lớp control trong phần sau.

##### **b)** **Thiết kế lớp entity**

Với bản mô tả hiện tại và đối với 2 use case “Pay Order” và “Place Order”, đa số các lớp entity trong thiết kế kiến trúc đã đơn giản, và có thể ánh xạ 1-1 với lớp thiết kế.

##### **c)** **Thiết kế lớp control**

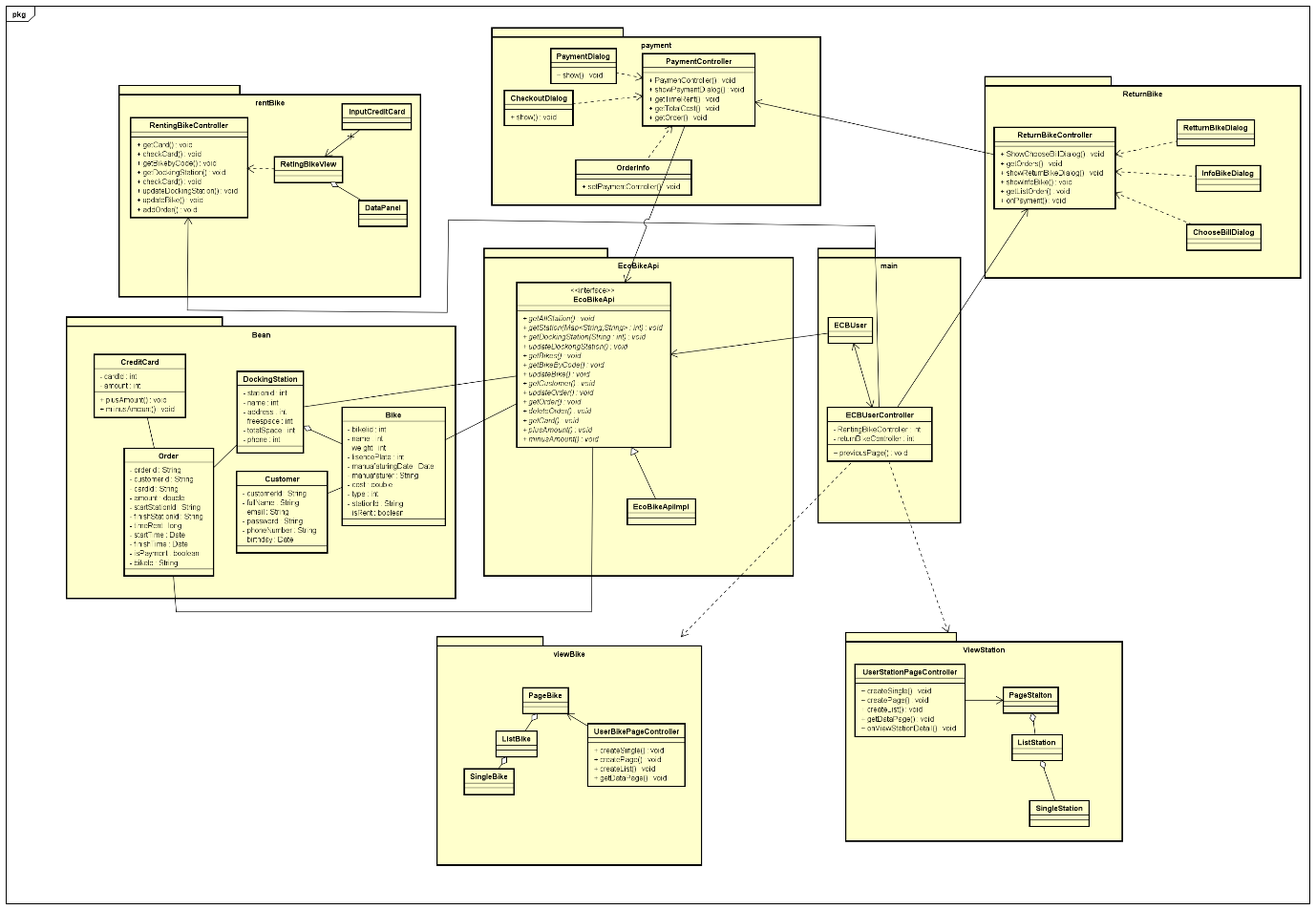
Tương tự, đa số các lớp control trong thiết kế kiến trúc đã đơn giản, và có thể ánh xạ 1-1 với lớp thiết kế. Tuy nhiên, InterbankSubsystemController đang phụ trách 2 tác vụ: (1) điều khiển luồng dữ liệu và (2) chuyển đổi dữ liệu (chuyển đổi dữ liệu sang định dạng yêu cầu và xử lý kết quả trả về). Do đó chúng ta cần ít nhất một lớp khác để phụ trách chuyển đổi dữ liệu, ví dụ JSON hoặc MyMap (tùy thuộc vào thiết kế mà lớp này có thể tái sử dụng cho các hệ thống thông tin web khác).

##### **d)** **Nhóm các lớp thiết kế**

Sau đây là một cách để nhóm các lớp thiết kế vào các package.

#### **1.2. Xác định mối quan hệ giữa các lớp**

Để phục vụ mục đích minh họa mối quan hệ giữa các lớp được dễ nhìn và rõ ràng, các lớp ở các hình sau không bao gồm các attribute cũng như operation/method.



III. **THIẾT KẾ GIAO DIỆN**

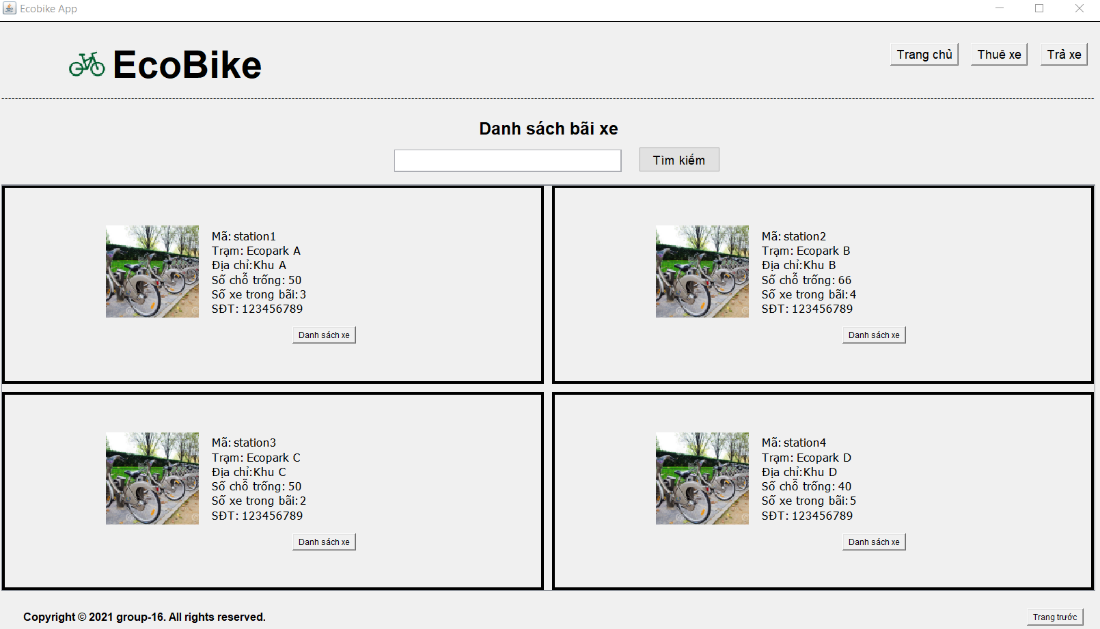
## **Tạo các ảnh màn hình**

* Thứ tự các màn hình trong hệ thống

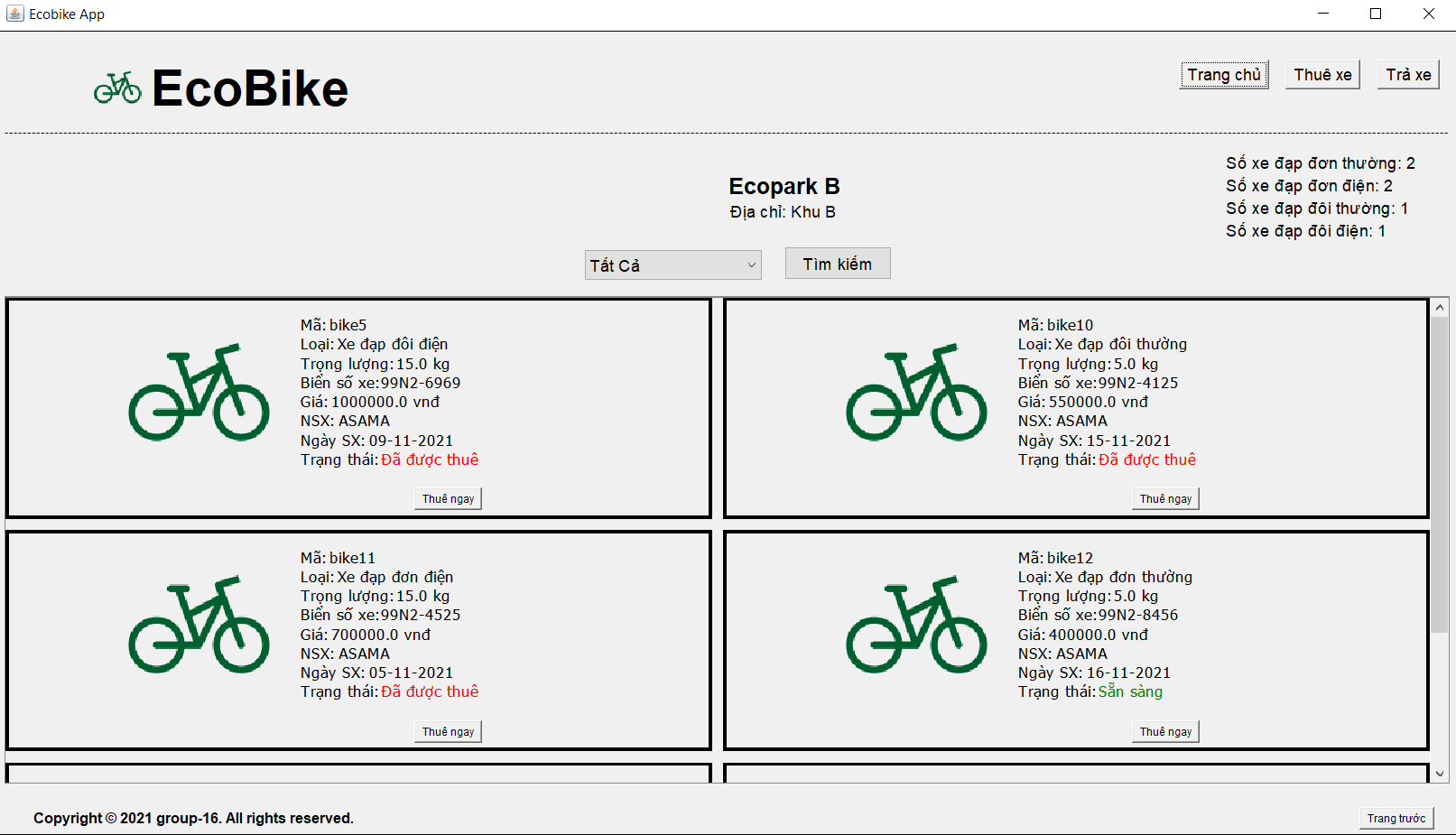
1. Splash screen
2. Trang chủ
3. Kết quả tìm kiếm bãi xe
4. Xem thông tin chi tiết bãi xe
5. Thuê xe
6. Thanh toán
7. Kết quả thanh toán
8. Xem thông tin xe đang thuê
9. Trả xe
10. Kết quả trả xe

* Các hình ảnh màn hình được tạo ra dưới đây được tạo ra bởi Scene Builder

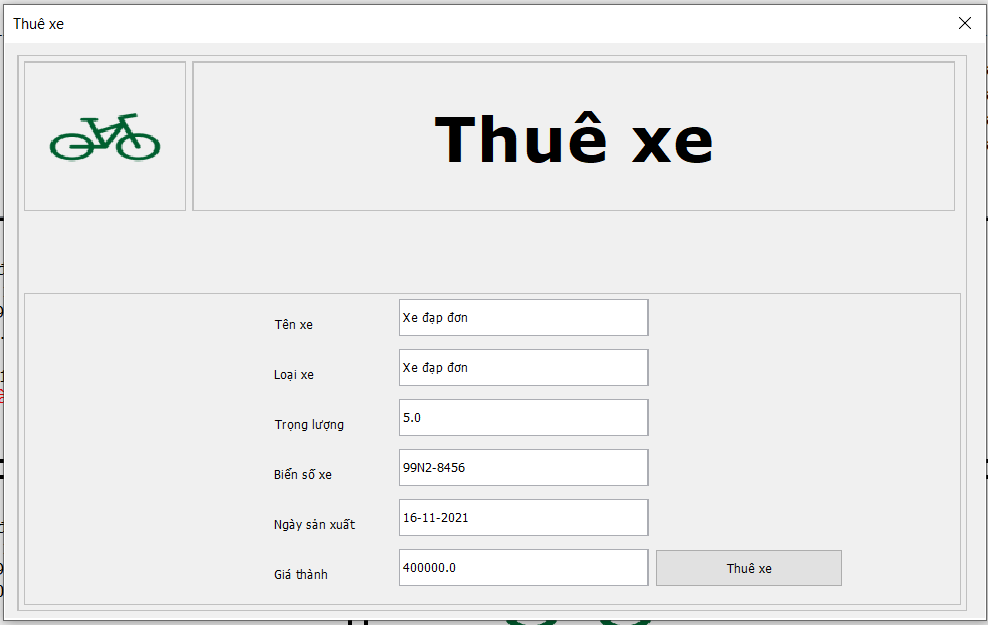
### Trang chủ



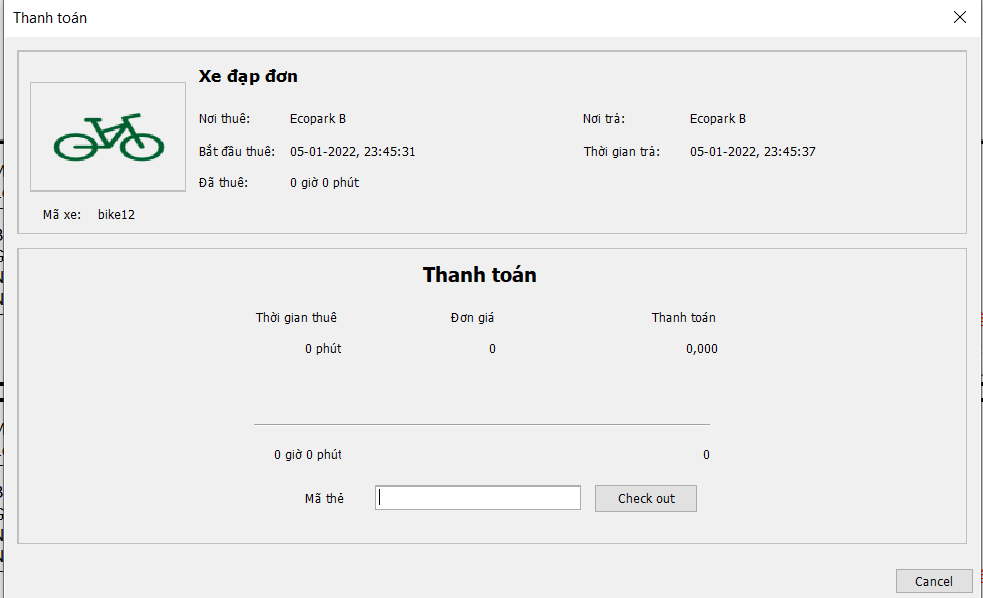
### Xem thông tin chi tiết bãi xe



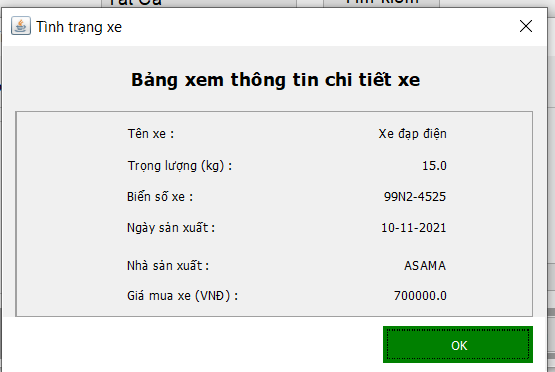
### Thuê xe



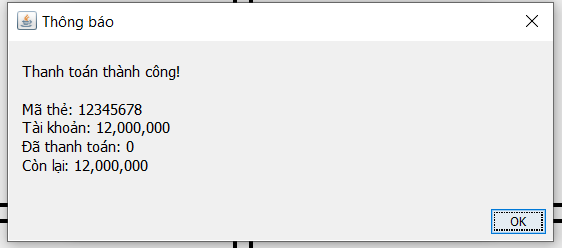
### Thanh toán



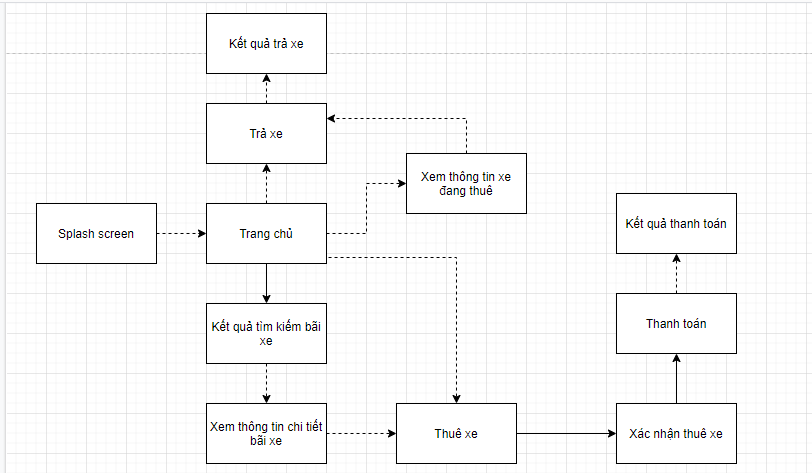
### Trả xe



### Kết quả trả xe



## **2.Tạo dịch chuyển màn hình**



3. Design Consideration

3.1 Coupling & cohesion

3.2 Design principle

3.3 Design patterns