HANOI UNIVERSITY OF SCIENCE AND TECHNOLOGY

School of Information and communications technology

Software Design Document

Version 1.0

Capstone Project

Subject: EcoBikeRental

Group 19

|  |  |  |
| --- | --- | --- |
| Lê Minh Tuấn | 20173442 | CNTT11 |
| Lý Bá Tuấn | 20173438 | CNTT.10 |
| Nguyễn Công Trưởng | 20173431 |  |
| Lê Thanh Tùng |  |  |

*Hanoi,* *12 / 2020*

Table of Contents

Table of Contents 1

1 Giới thiệu 4

1.1 Mục tiêu 4

1.2 Phạm vi 4

1.3 Glossary 4

2 Mô tả tổng quan 5

2.1 Tổng quan 5

2.2 Assumptions/Constraints/Risks 5

2.2.1 Assumptions 5

2.2.2 Constraints 5

3 System Architecture và Architecture Design 6

3.1 Mẫu kiến trúc sử dụng: Kiến trúc 3 tầng 6

3.2 Sequence Diagrams 6

3.2.1 Sequence Diagrams cho nghiệp vụ thuê xe 7

3.2.2 Sequence Diagrams cho nghiệp vụ trả xe 8

3.2.3 Sequence Diagrams cho nghiệp vụ xem thông tin trạm và xe 9

3.3 Analysis Class Diagrams 9

3.3.1 Analysis Class Diagrams for Rent bike business 9

3.3.2 Analysis Class Diagrams for Return bike business 11

3.3.3 Analysis Class Diagrams for View bike and station business 12

Detailed Design 13

3.4 User Interface Design 13

3.4.1 Screen Transition Diagrams 13

3.4.2 Screen Specifications 13

3.5 Data Modeling 25

3.5.1 Conceptual Data Modeling 25

3.5.2 Database Design 27

3.6 Class Design 32

3.6.1 General Class Diagram 32

3.6.2 Biểu đồ lớp 33

3.6.3 Class Design 37

4 Design Considerations 39

4.1 Mục tiêu và phương châm 39

4.2 Chiến lược 39

4.3 Coupling and Cohesion 40

4.4 Design Principles 40

4.5 Design Patterns 42

**Danh sách các hình**

[Hình 1. *Biểu đồ tuần tự nghiệp vụ thuê xe* 7](#_Toc59685384)

[Hình 2. *biểu đồ trình tự cho nghiệp vụ trả xe* 8](#_Toc59685385)

[Hình 3*. Biểu đồ trình tự nghiệp vụ xem thông tin xe và trạm xe* 9](#_Toc59685386)

[Hình 4. *Biểu đồ lớp phân tích nghiệp vụ thuê xe* 10](#_Toc59685387)

[Hình 5*. Biểu đồ lớp phân tích nghiệp vụ trả xe* 11](#_Toc59685388)

[Hình 6*. Biểu đồ lớp phân tích nghiệp vụ xem thông tin xe và trạm xe* 12](#_Toc59685389)

[Hình 7. *Biểu đồ dịch chuyển màn hình* 13](#_Toc59685390)

[Hình 8. *Sơ đồ thực thể liên kết* 26](#_Toc59685391)

[Hình 9.*Model diagrams* 27](#_Toc59685392)

[Hình 10. *Biểu đồ mô tả phụ thuộc chung giữa các package trong hệ thống* 32](#_Toc59685393)

[Hình 11. *Biểu đồ lớp của package Data* 33](#_Toc59685394)

[Hình 12. *Biểu đồ lớp cho package Business* 34](#_Toc59685395)

[Hình 13. *Biểu đồ lớp cho package Presentation* 34](#_Toc59685396)

[Hình 14. *Biểu đồ lớp cho package Exception* 35](#_Toc59685397)

[Hình 15. *Biểu đồ lớp cho package Util* 36](#_Toc59685398)

[Hình 16. *Biểu đồ lớp cho subsystem Interbank* 37](#_Toc59685399)

[Hình 17. *Ví dụ 1 áp dụng nguyên tắc Open/closed* 40](#_Toc59685400)

[Hình 18. *Ví dụ 2 áp dụng nguyên tắc Open/Closed* 41](#_Toc59685401)

[Hình 19. *Ví dụ áp dụng nguyên tắc Liskov* 41](#_Toc59685402)

[Hình 20. *Ví dụ áp dụng nguyên tắc Interface segregation* 41](#_Toc59685403)

[Hình 21. *Ví dụ áp dụng Design Pattern Singleton* 42](#_Toc59685404)

**Danh sách các bảng**

[Bảng 1. *Đặc tả chi tiết màn hình trang chủ (không thuê xe)* 14](#_Toc59685405)

[Bảng 2. *Đặc tả chi tiết màn hình Home (đang thuê xe)* 16](#_Toc59685406)

[Bảng 3. *Đặc tả chi tiết màn hình khi trả xe thành công* 18](#_Toc59685407)

[Bảng 4. *Đặc tả chi tiết màn hình trả xe* 18](#_Toc59685408)

[Bảng 5. *Đặc tả chi tiết màn hình Thuê xe* 20](#_Toc59685409)

[Bảng 6. *Đặc tả chi tiết màn hình chi tiết thông tin xe* 21](#_Toc59685410)

[Bảng 7. *Đặc tả chi tiết màn hình điền thông tin thanh toán* 23](#_Toc59685411)

[Bảng 8. *Đặc tả màn hình khi thuê xe thành công.* 25](#_Toc59685412)

[Bảng 9*. Chi tiết thiết kế bảng Bikes* 28](#_Toc59685413)

[Bảng 10. *Chi tiết thiết kế bảng Stations* 29](#_Toc59685414)

[Bảng 11.*Chi tiết thiết kế bảng Rental* 30](#_Toc59685415)

[Bảng 12. *Chi tiết thiết kế bảng Customer* 30](#_Toc59685416)

[Bảng 13. *Chi tiết thiết kế bảng Cards* 31](#_Toc59685417)

[Bảng 14. *Chi tiết thiết kế bảng Electric\_bike* 31](#_Toc59685418)

[Bảng 15. *Chi tiết thiết kế bảng Rental\_Transaction* 31](#_Toc59685419)

[Bảng 16. *Chi tiết thiết kế bảng Transactions* 32](#_Toc59685420)

# Giới thiệu

*Tài liệu cung cấp bản thiết kế hệ thống bao gồm các phần mô tả kiến trúc ngoài của hệ thống, thiết kế lớp, thiết kế cơ sở dữ liệu và giao diện người dùng dựa trên các mô tả trong tài liệu SRS.*

*Tài liệu là cơ sở để các thành viên trong cả nhóm có thể hiểu đề phối hợp làm việc với nhau một cách thống nhất. Ngoài ra tài liệu này còn dùng vào mục đích khi muốn mở rộng hệ thống*

## Mục tiêu

Tài liệu SDD này được xây dựng nhằm mục đích nêu lên thiết kế phần mềm của dự án, được sử dụng cho programmer, tester, maintainer, … Nó bao gồm việc thiết kế chi tiết cho kiển 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 để từ đó người đọc sẽ có cái nhìn rõ ràng hơn về phần mềm cần xây dựng và nó sẽ là tài liệu chính thức để từ đó những người xây dựng phần mềm có thể xây dựng nên phần mềm dựa vào tài liệu này.

## Phạm vi

Phần mềm được tạo ra để hỗ trợ cho hoạt động thuê xe đạp theo giờ tại khu đô thị Ecopark.

Người dùng có thể đăng kí tài khoản trên hệ thống EcoBike Rental, xác thực thông tin tài khoản, thiết lập phương thức thanh toán.

Khi mở ứng dụng người dùng có thể tìm trạm gửi xe dựa trên vị trí hiện tại, tiến hành xem các thông tin chi tiết về trạm xe: tên của trạm xe, địa chỉ, diện tích bãi, khoảng cách và thời gian đi bộ của khách hàng đến bãi,khi đến người dùng chọn xe phù hợp và tiến hành thuê xe.

Sau khi thuê, người dùng có thể xem các thông tin chi tiết về các thông số hiện tại của xe: thời lượng pin, số tiền phải trả,…

Khi tiến hành trả xe, hệ thống sẽ tự động trừ tiền và kết thúc hoạt động thuê xe.

# Mô tả tổng quan

## Tổng quan

*Tài liệu này mô tả các thiết kế chi tiết hệ thống thuê xe tự động EcoBikeRental, phục vụ hoạt động trong khu đô thị EcoPark. Tài kiệu thiết kế theo chiều phát triển dần dần của ứng dụng, đi từ việc lựa chọn mẫu kiến trúc thiết kế sau đó tới xây dựng các biểu đồ trình tự biểu thị luồng hoạt động của các nghiệp vụ trong hệ thống một cách cụ thể và rõ ràng nhất. Sau đó, sẽ phân tích các lớp, rồi xây dựng thực thể, xây dựng database, giao diện và cuối cùng là thiết kế lớp chi tiết cùng với các pattern sử dụng trong source code viết nên hệ thống.*

## Assumptions/Constraints/Risks

### Assumptions

*Các vấn đề có thể phát sinh về phần cứng, phần mềm, các mối quan hệ:*

### Constraints

*Các yêu cầu hệ thống về phần cứng, phần mềm:*

* *Hardware or software environment*
* *End-user environment*
* *Availability or volatility of resources*
* *Standards compliance*
* *Interoperability requirements*
* *Interface/protocol requirements*
* *Licensing requirements*
* *Data repository and distribution requirements*
* *Security requirements (or other such regulations)*
* *Memory or other capacity limitations*
* *Performance requirements*
* *Network communications*
* *Verification and validation requirements (testing)*
* *Other means of addressing quality goals*
* *Other requirements described in the Requirements Document*

# System Architecture và Architecture Design

Các bước thiết kế kiến trúc:

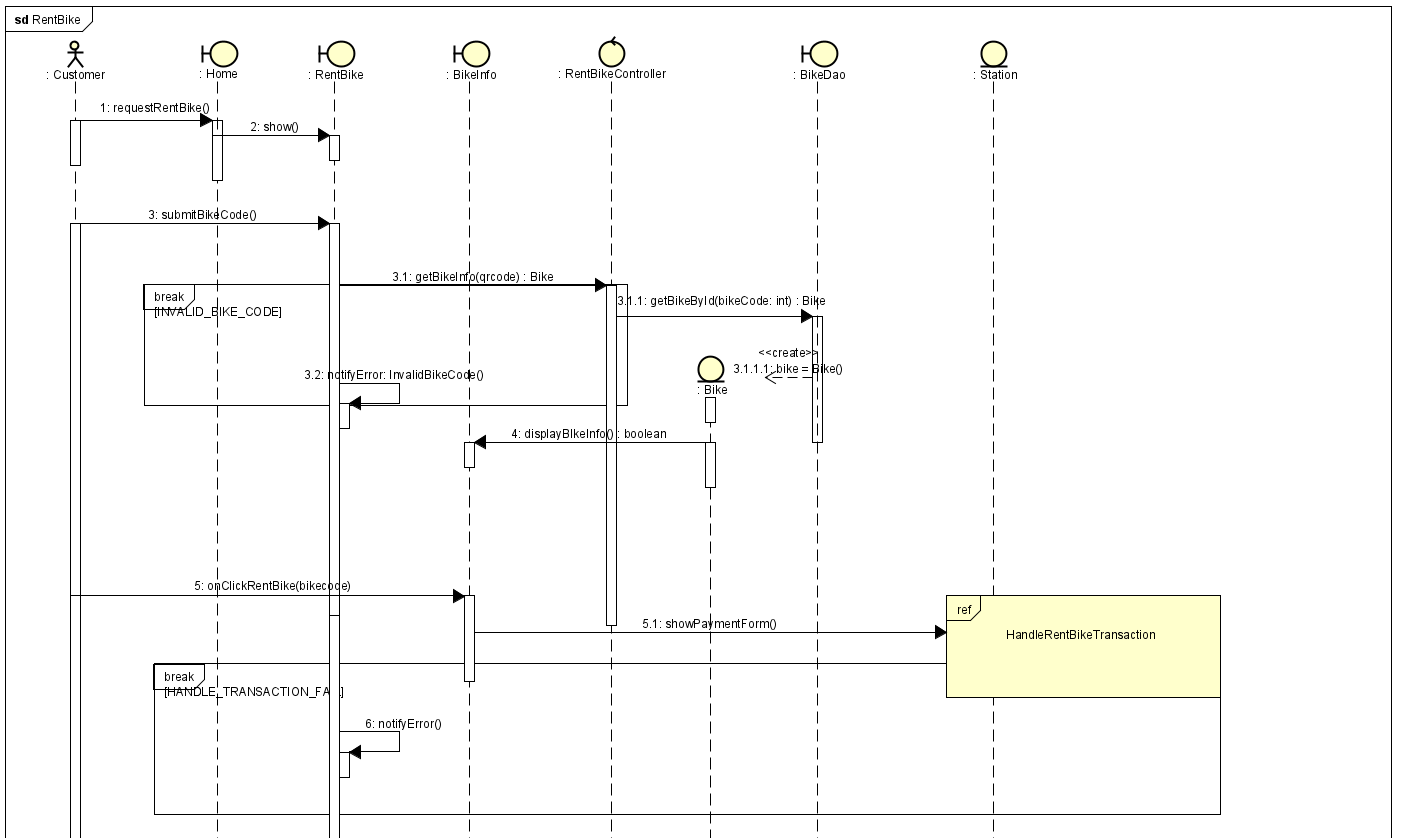
1. Xây dựng khung kiến trúc.
2. Từ tài liệu SRS, thiết kế luồng hoạt động tuần tự trong hệ thống qua biểu đồ trình tự (Sequence diagram).
3. Vẽ biểu đồ lớp phân tích sau khi hoàn thành biểu đồ trình tự.
4. Xây dựng database
5. Xây dựng giao diện, đặc tả chi tiết giao diện.
6. Từ biểu đồ lớp phân tích, bắt đầu thiết kế lớp chi tiết.

## Mẫu kiến trúc sử dụng: Kiến trúc 3 tầng

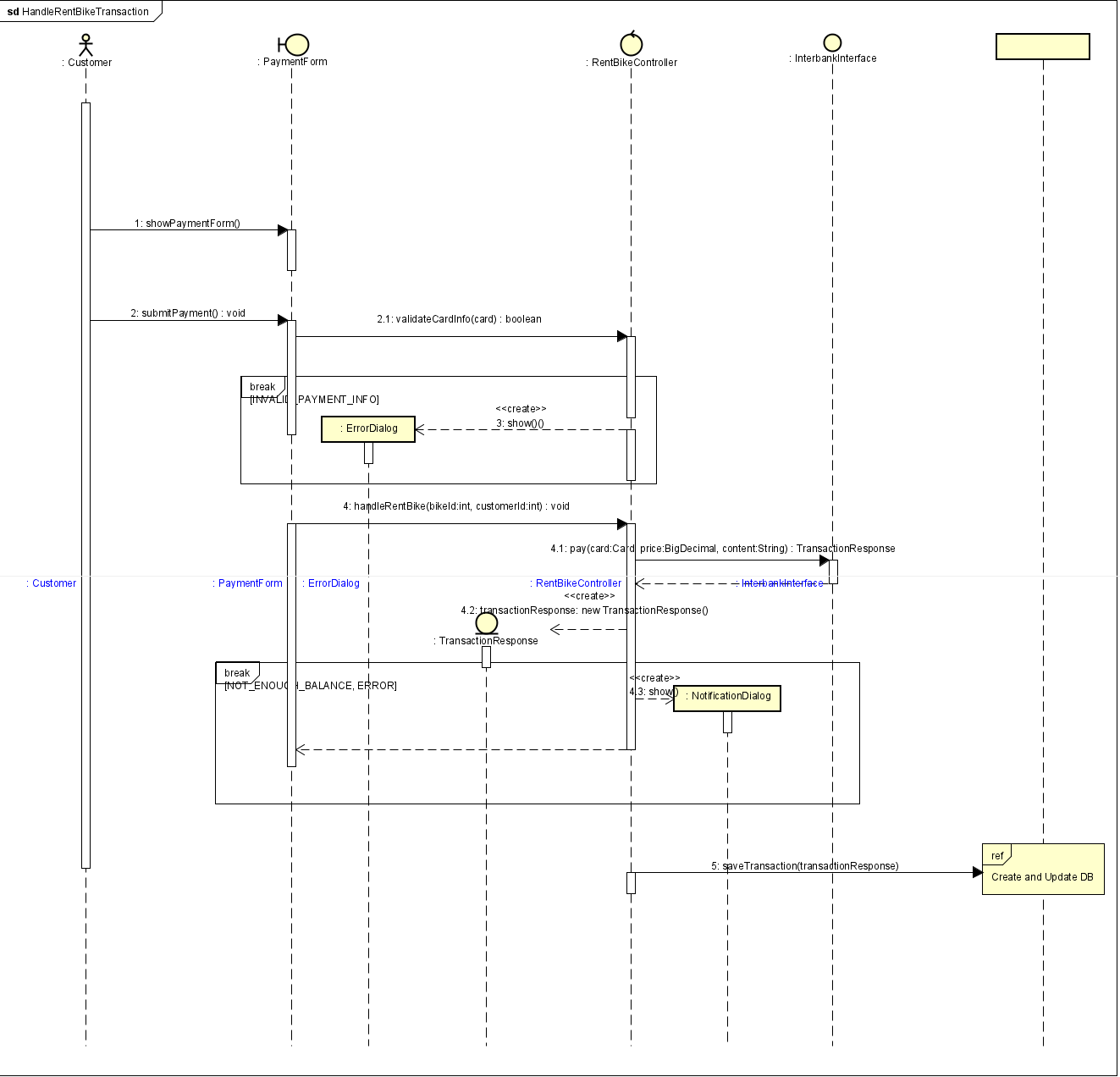
* *Mô tả: Kiến trúc 3 tầng (three layer) chia ứng dụng ra làm 3 tầng chính là tầng trình diễn (presentation), tầng nghiệp vụ (business) và tầng dữ liệu (data). Tầng ứng dụng xử lý các nghiệp vụ logic. Tầng dữ liệu để lưu trữ thông tin, biểu diễn, trao đổi thông tin với Database. Tầng trình diễn xử lý và hiển thị giao diện lên cho người dùng.*
* *Lý do lựa chọn:*
  + *Kiến trúc 3 tầng dễ hiểu, không khó trong việc triển khai đồng thời dễ bảo trì khi các tầng được phân chia rõ ràng, tái sử dụng tốt.*
  + *Dễ dàng phân chia khối lượng công việc cho các thành viên trong nhóm theo các chức năng mà ít bị xung đột lẫn nhau.*

## Sequence Diagrams

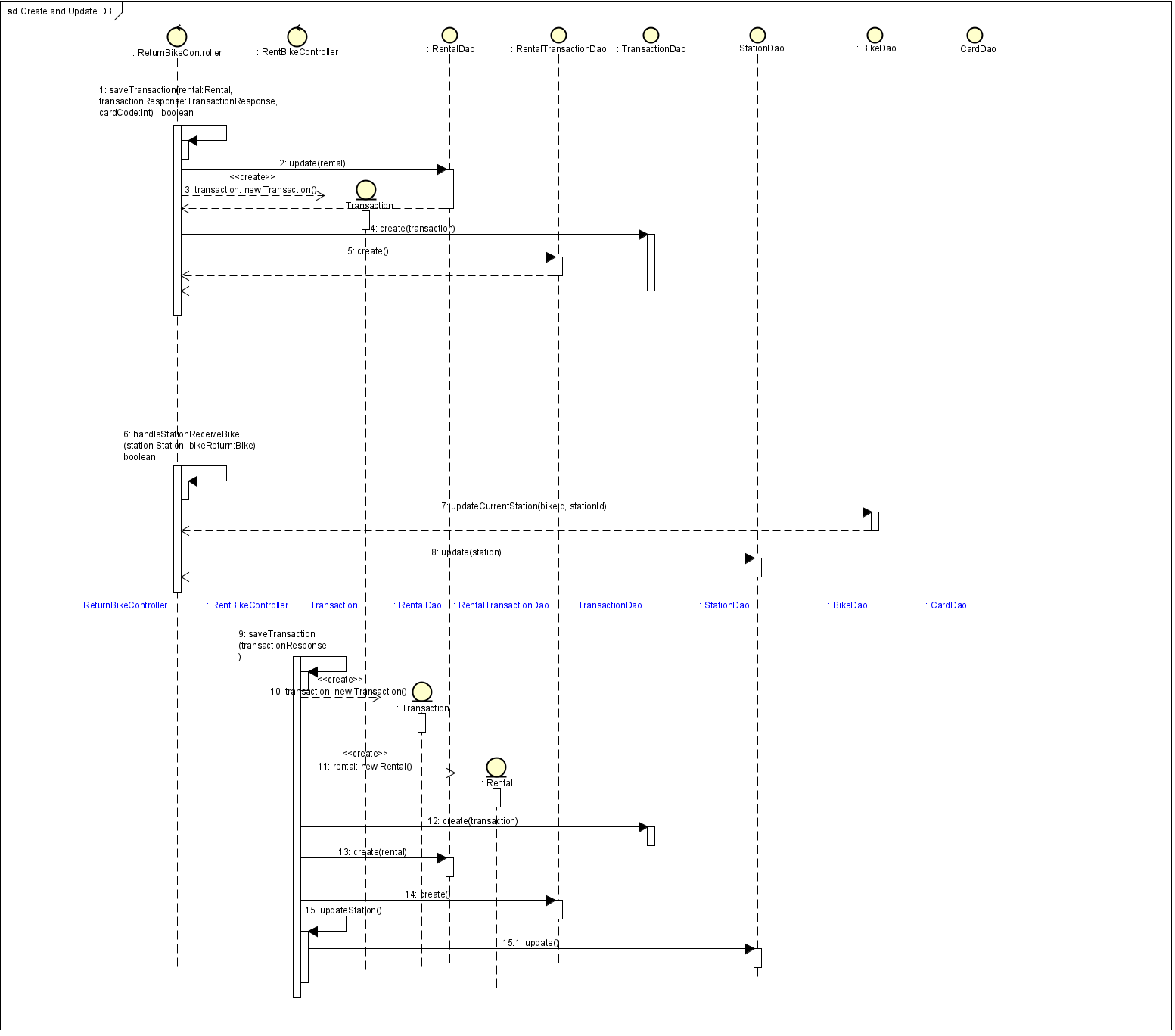
### Sequence Diagrams cho nghiệp vụ thuê xe



Hình . *Biểu đồ tuần tự nghiệp vụ thuê xe*

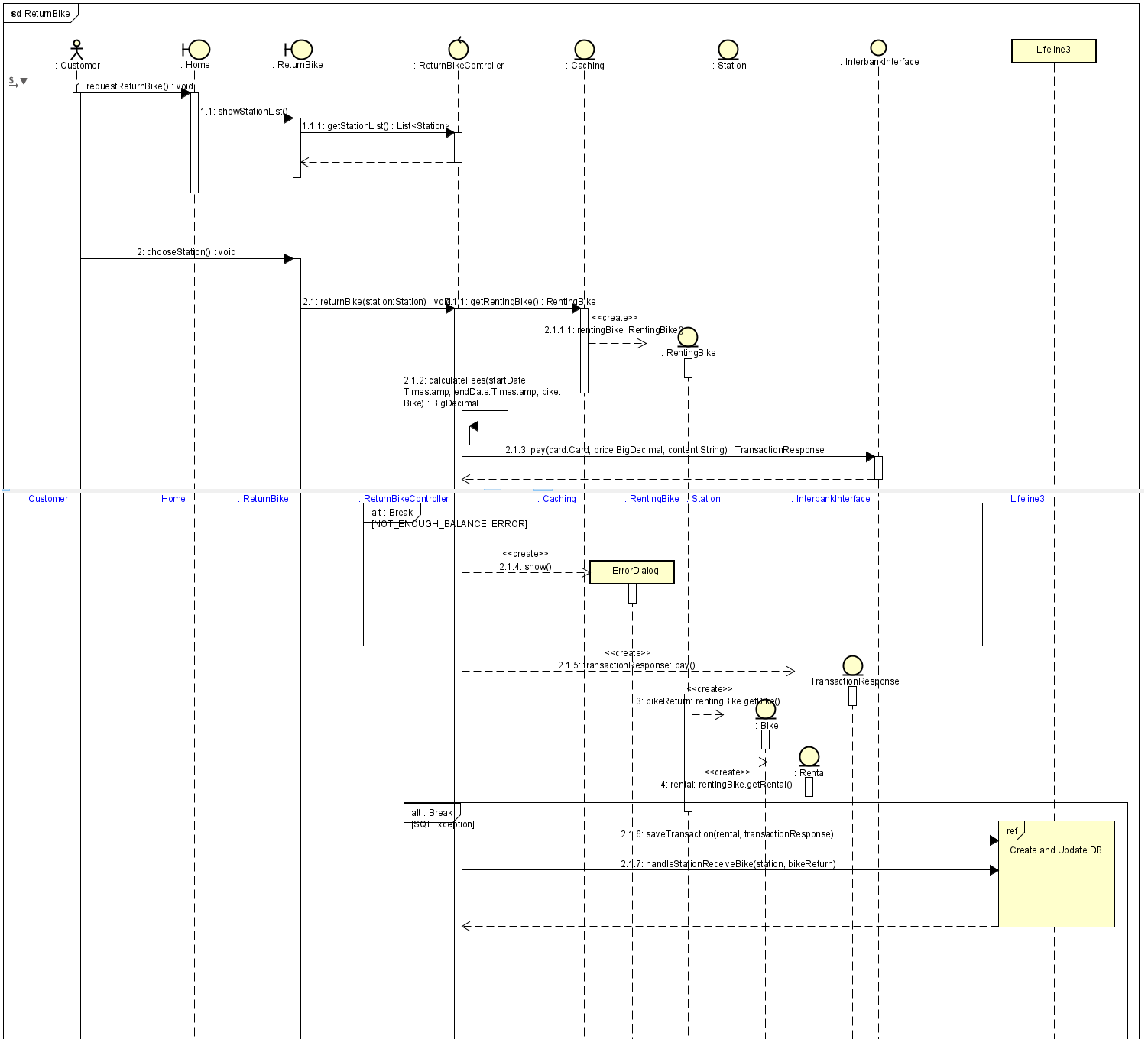


Hình . *HandleRentBikeTransaction*



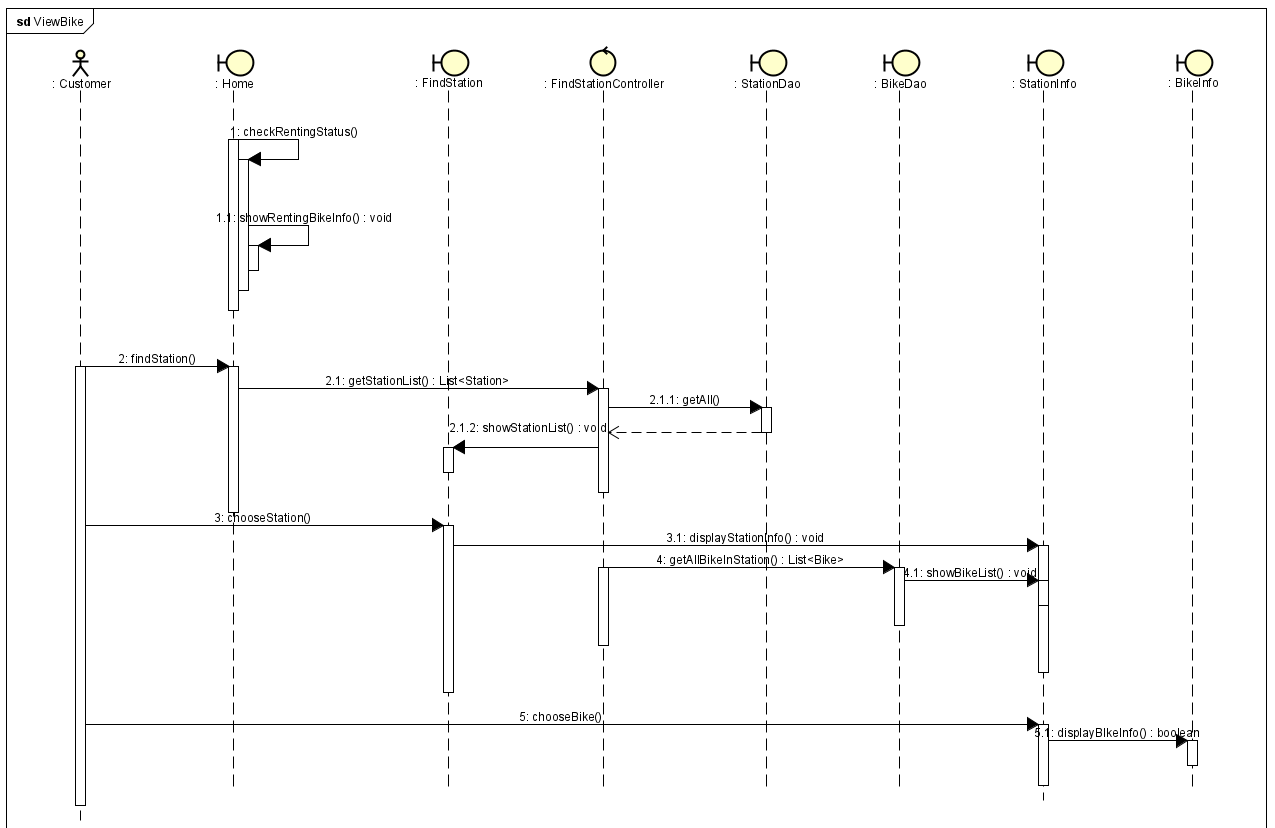
Hình . *Create and Update DB*

### Sequence Diagrams cho nghiệp vụ trả xe



Hình . *biểu đồ trình tự cho nghiệp vụ trả xe*

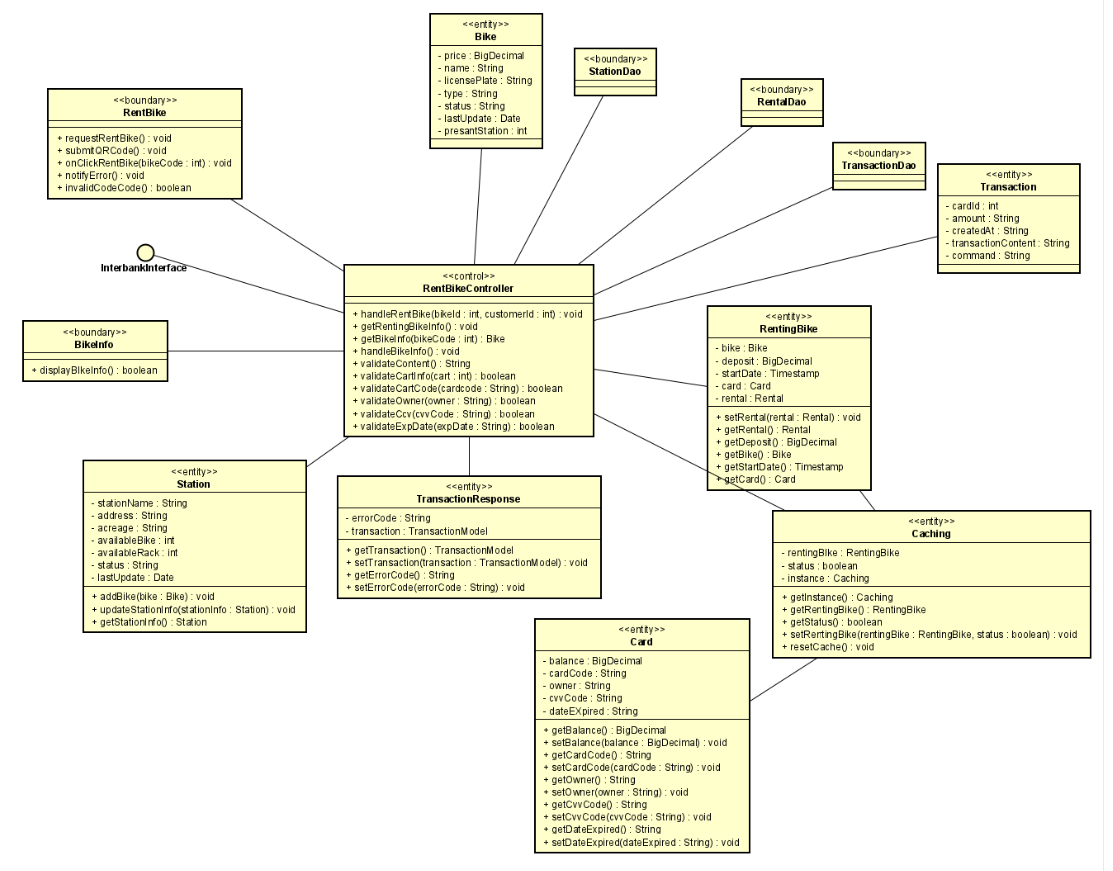
### Sequence Diagrams cho nghiệp vụ xem thông tin trạm và xe



Hình *. Biểu đồ trình tự nghiệp vụ xem thông tin xe và trạm xe*

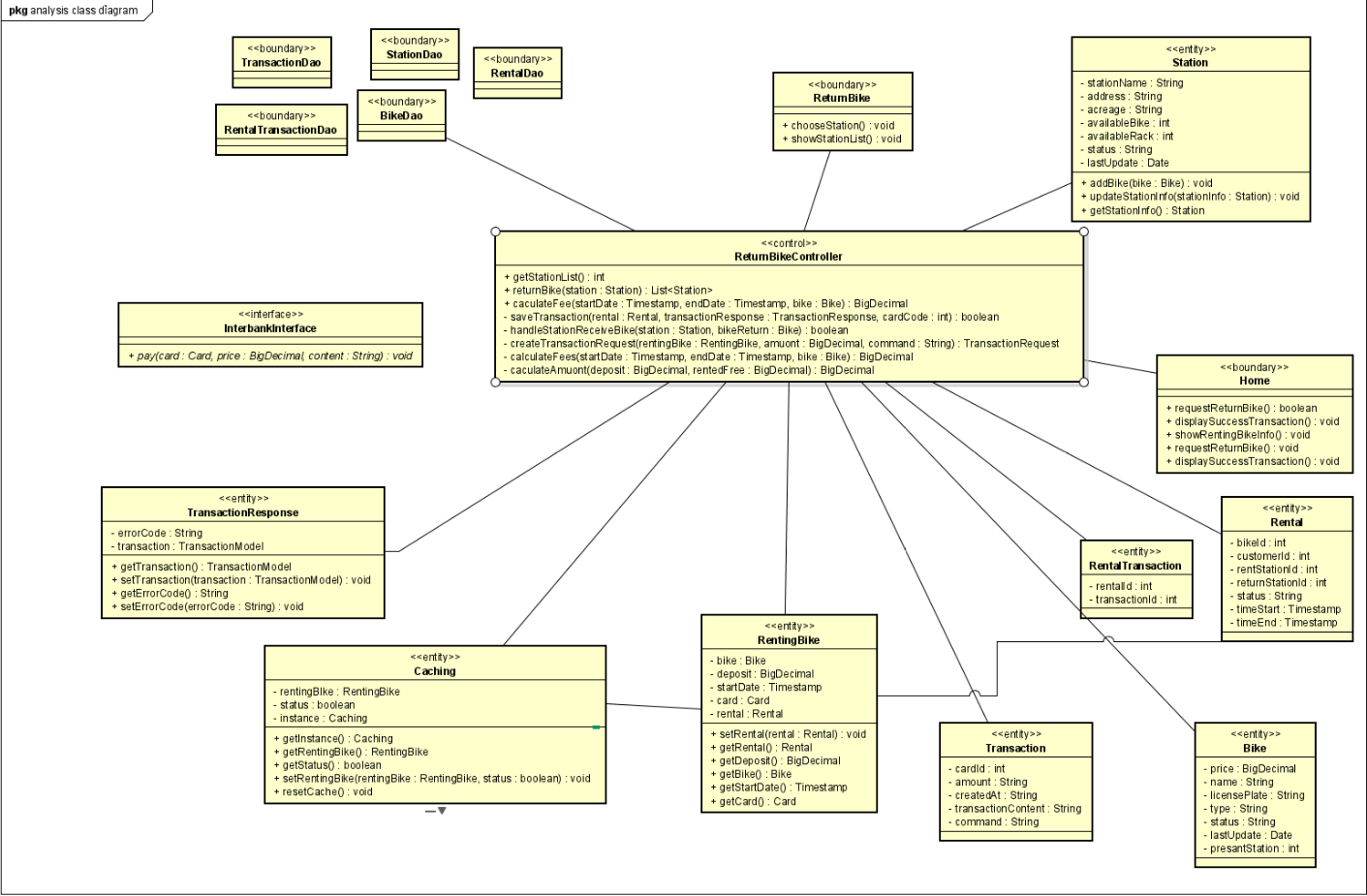
## Analysis Class Diagrams

### Analysis Class Diagrams for Rent bike business



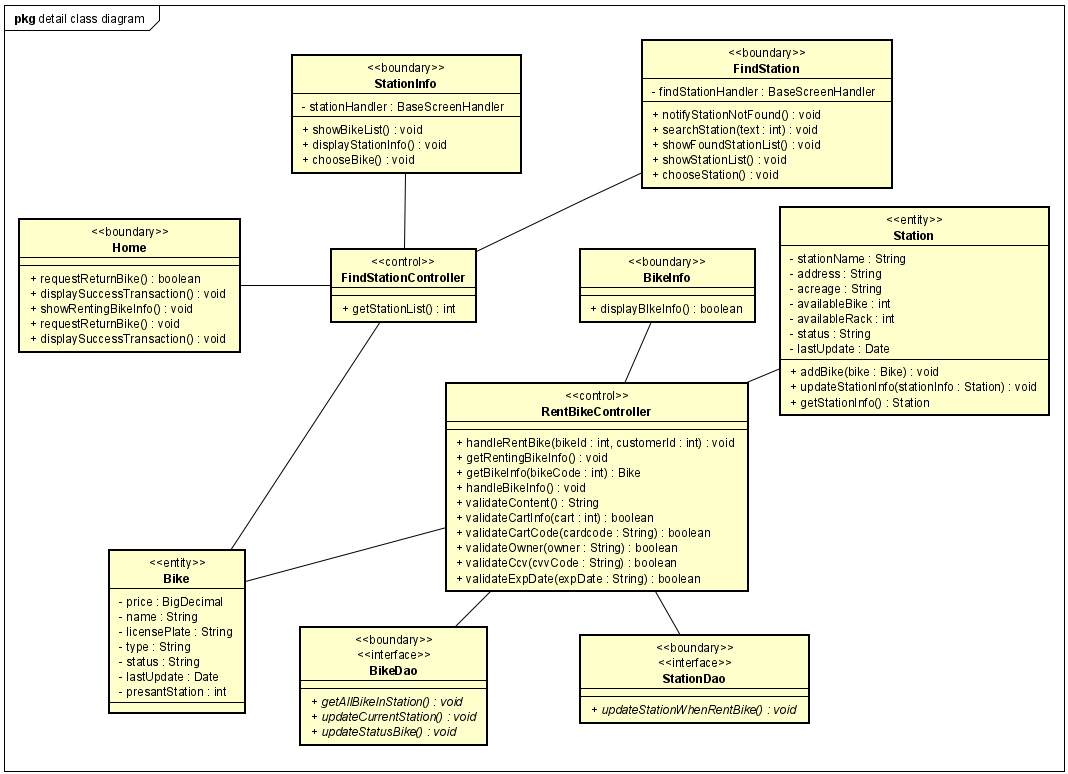
Hình . *Biểu đồ lớp phân tích nghiệp vụ thuê xe*

### Analysis Class Diagrams for Return bike business



Hình *. Biểu đồ lớp phân tích nghiệp vụ trả xe*

### Analysis Class Diagrams for View bike and station business

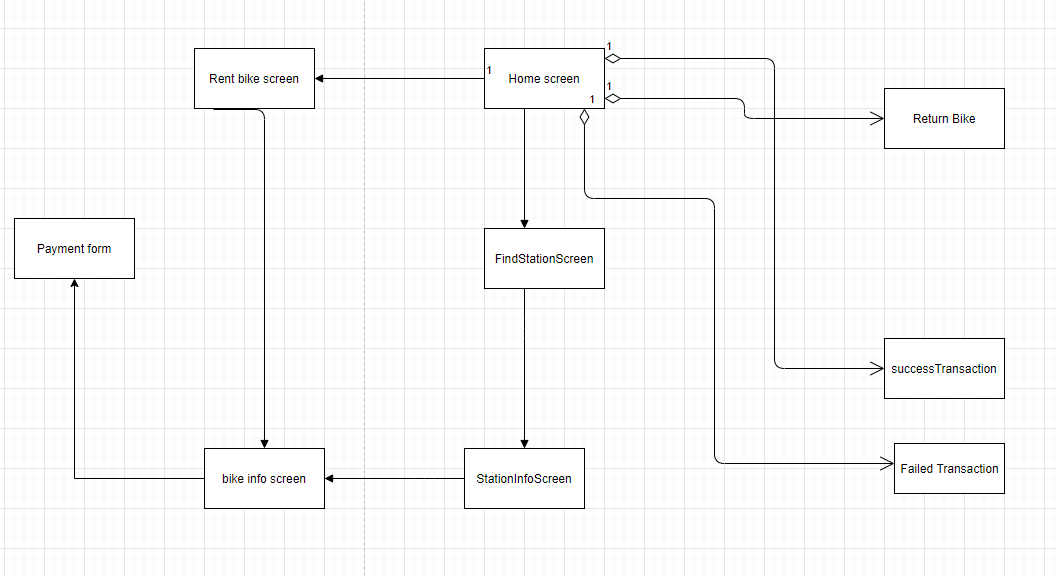


Hình *. Biểu đồ lớp phân tích nghiệp vụ xem thông tin xe và trạm xe*

# Detailed Design

## User Interface Design

### Screen Transition Diagrams



Hình . *Biểu đồ dịch chuyển màn hình*

Từ màn hình Home, người dùng có thể chuyển hướng sang các màn Rent Bike, màn FindStation, màn ReturnBike nếu đang thuê xe.

Từ màn hình FindStation, khi người dùng chọn một station trên danh sách station thì sẽ chuyển hướng sang màn StationInfo. Sau đó, khi chọn một bike trên danh sách Bike hiện có của Station, hệ thống sẽ chuyển sang màn BikeInfo. Và khi thuê xe thì sẽ được chuyển sang màn nhập thông tin thanh toán (PaymentForm)

### Screen Specifications

1. **Home Screen**
   1. **Màn Home khi không thuê xe**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| EcoBikeRental | | Date of creation | Approved By | Reviewed By | Person in charge |
| Screen specification | Home screen | 18/11/2020 |  |  | Ly Ba Tuan |
|  | | Control | Operation | Function | |
| Find Station Button | Click | Show list Station | |
|  |  |  | |
|  |  |  | |
|  |  |  | |
|  |  |  | |
|  |  |  | |
|  |  |  |  |  |  |

Bảng . *Đặc tả chi tiết màn hình trang chủ (không thuê xe)*

**Định nghĩa các trường thuộc tính:**

|  |  |
| --- | --- |
| Screen name | Home Screen |
| Item name | Number of  digits (bytes) | Type | Field attribute | Remarks |
| Title | 20 | String | Black | Left-justified |
| Tài khoản | 12 | String | Black | Right justified |

* 1. **Màn hình Home khi đang thuê xe**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| EcoBikeRental | | Date of creation | Approved By | Reviewed By | Person in charge |
| Screen specification | Home screen | 18/11/2020 |  |  | Ly Ba Tuan |
|  | | Control | Operation | Function | |
| Find Station Button | Click | Show list Station | |
| Rent Bike Button | Click | Go to Rent Bike Screen | |
| Info Button | Click | Show current bike info | |
| Return Bike Button | Click | Go to Return Bike Screen | |
|  |  |  | |
|  |  |  | |
|  |  |  |  |  |  |

Bảng . *Đặc tả chi tiết màn hình Home (đang thuê xe)*

**Định nghĩa các trường thuộc tính**

|  |  |
| --- | --- |
| Screen name | Home Screen |
| Item name | Number of  digits (bytes) | Type | Field attribute | Remarks |
| Title | 20 | String | Black | Left-justified |
| Tài khoản | 12 | String | Black | Right justified |
| Thời gian | 10 | DateTime | Black |  |
| Phí | 20 | Numeric | Black |  |
| Kiểu xe | 10 | String | Black |  |

* 1. **ReturnBikeSuccessfulHomeScreen**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| EcoBikeRental | | Date of creation | Approved By | Reviewed By | Person in charge |
| Screen specification | Return bike screen | 18/11/2020 |  |  | Ly Ba Tuan |
|  | | Control | Operation | Function | |
| Confirm button | Click | Close notification | |
|  |  |  | |
|  |  |  | |
|  |  |  | |
|  |  |  | |
|  |  |  | |
|  |  |  |  |  |  |

Bảng . *Đặc tả chi tiết màn hình khi trả xe thành công*

**Định nghĩa các trường thuộc tính**

|  |  |
| --- | --- |
| Screen name | Home Screen |
| Item name | Number of  digits (bytes) | Type | Field attribute | Remarks |
| message | 20 | String | Black | Left-justified |

1. **Return Bike Screen**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| EcoBikeRental | | Date of creation | Approved By | Reviewed By | Person in charge |
| Screen specification | Home screen | 18/11/2020 |  |  | Ly Ba Tuan |
|  | | Control | Operation | Function | |
| Station Field | Click | Select Station to return bike | |
| Back | Click | Return preScreen | |
|  |  |  | |
|  |  |  |  |  |  |

Bảng . *Đặc tả chi tiết màn hình trả xe*

**Định nghĩa các trường thuộc tính:**

|  |  |
| --- | --- |
| Screen name | Return Bike Screen |
| Item name | Number of  digits (bytes) | Type | Field attribute | Remarks |
| Title | 20 | String | Black | Left-justified |
| Tên trạm xe | 12 | String | Black | Right justified |
| Vị trí | 100 | String | Black |  |
| Trạng thái | 10 | String | Black |  |

1. **Rent Bike Screen**
   1. **Type Bike’s code**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| EcoBikeRental | | Date of creation | Approved By | Reviewed By | Person in charge |
| Screen specification | Rent bike screen | 18/11/2020 |  |  | Ly Ba Tuan |
|  | | Control | Operation | Function | |
| Back | Click | Return preScreen | |
| Rent Bike Button | Click | Go to Bike Info to rent bike | |
| Input | Type | Type Bike’s code | |
|  |  |  | |
|  |  |  | |
|  |  |  |  |  |  |

Bảng . *Đặc tả chi tiết màn hình Thuê xe*

**Định nghĩa các trường thuộc tính**

|  |  |
| --- | --- |
| Screen name | Rent Bike Screen |
| Item name | Number of  digits (bytes) | Type | Field attribute | Remarks |
| Title | 20 | String | Black | Left-justified |

* 1. **Show Bike info**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| EcoBikeRental | | Date of creation | Approved By | Reviewed By | Person in charge |
| Screen specification | Rent bike screen | 18/11/2020 |  |  | Ly Ba Tuan |
|  | | Control | Operation | Function | |
| Back | Click | Return preScreen | |
| Rent Bike Button | Click | Go to Payment Screen to rent bike | |
|  |  |  | |
|  |  |  | |
|  |  |  |  |  |  |

Bảng . *Đặc tả chi tiết màn hình chi tiết thông tin xe*

**Định nghĩa các trường thuộc tính**

|  |  |
| --- | --- |
| Screen name | Rent bike Screen |
| Item name | Number of  digits (bytes) | Type | Field attribute | Remarks |
| Title | 20 | String | Black | Left-justified |
| Tên xe | 10 | String | Black | Right justified |
| Biển số xe | 20 | String | Black |  |
| Mã số xe | 20 | String | Black |  |
| Thời lượng pin | 4 | Numeric | Black |  |
| Đặt cọc | 10 | Numeric | Black |  |
| Loại xe | 20 | String | Black |  |

* 1. **Payment Form Screen**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| EcoBikeRental | | Date of creation | Approved By | Reviewed By | Person in charge |
| Screen specification | Rent bike screen | 18/11/2020 |  |  | Ly Ba Tuan |
|  | | Control | Operation | Function | |
| Find Station Button | Click | Show list Station | |
| Confirm Button | Click | Confirm Rent bike | |
| back | Click | Return preScreen | |
| Input Card number | Type | Type card number | |
| Input Owner | Type | Type owner | |
| Input CVV | Type | Type CVV number | |
| Input expiration date | Type | Type expiration date | |
| Input Bank | Type | Type bank | |
| Input comment | Type | Type comment | |
|  |  |  | |
|  |  |  |  |  |  |

Bảng . *Đặc tả chi tiết màn hình điền thông tin thanh toán*

**Định nghĩa các trường thuộc tính**

|  |  |
| --- | --- |
| Screen name | Rent bike Screen |
| Item name | Number of  digits (bytes) | Type | Field attribute | Remarks |
| Title | 20 | String | Black | Left-justified |
| Phí đặt cọc | 20 | Numeric | Black | Right justified |
| Giá thuê | 20 | Numeric | Black |  |

* 1. **Successful Rent bike Screen**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| EcoBikeRental | | Date of creation | Approved By | Reviewed By | Person in charge |
| Screen specification | Rent bike screen | 18/11/2020 |  |  | Ly Ba Tuan |
|  | | Control | Operation | Function | |
| Confirm button | Click | Confirm status rent bike | |
|  |  |  | |
|  |  |  | |
|  |  |  |  |  |  |

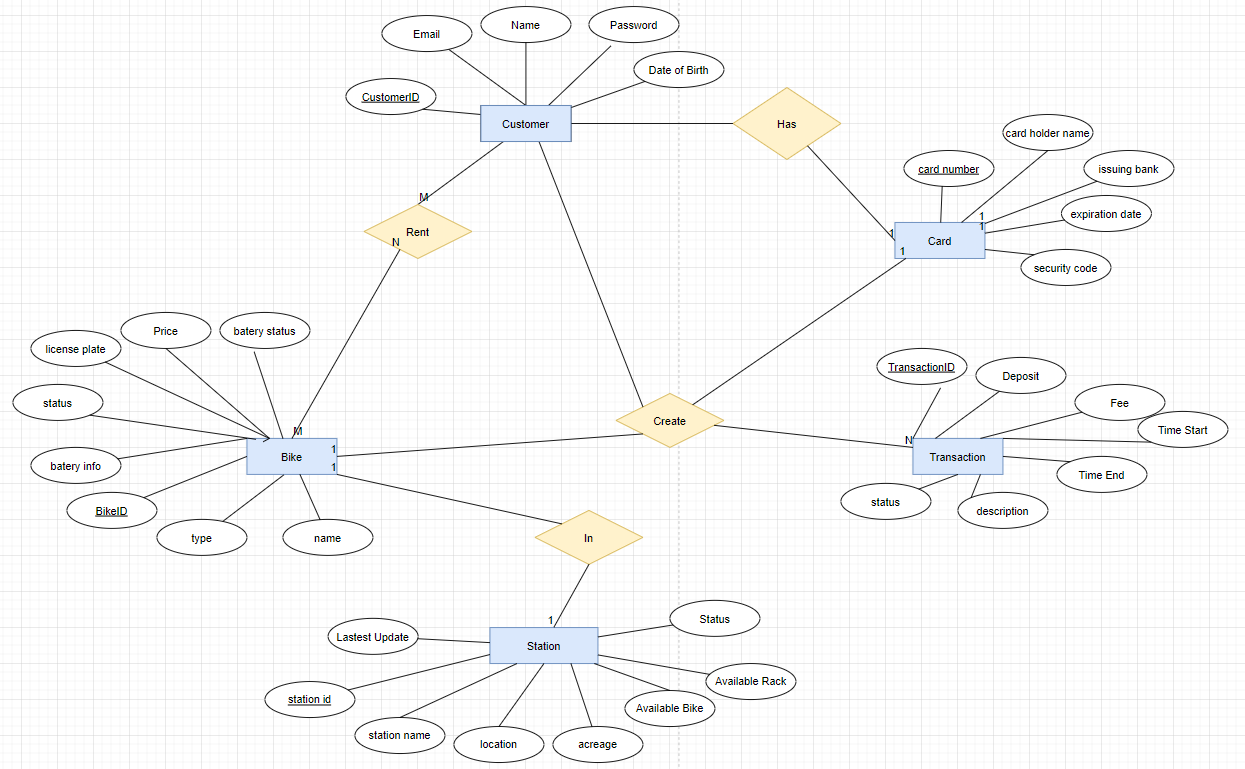
Bảng . *Đặc tả màn hình khi thuê xe thành công.*

**Định nghĩa các trường thuộc tính**

|  |  |
| --- | --- |
| Screen name | Rent bike Screen |
| Item name | Number of  digits (bytes) | Type | Field attribute | Remarks |
| message | 20 | String | Black | Left-justified |

## Data Modeling

### Conceptual Data Modeling



Hình . *Sơ đồ thực thể liên kết*

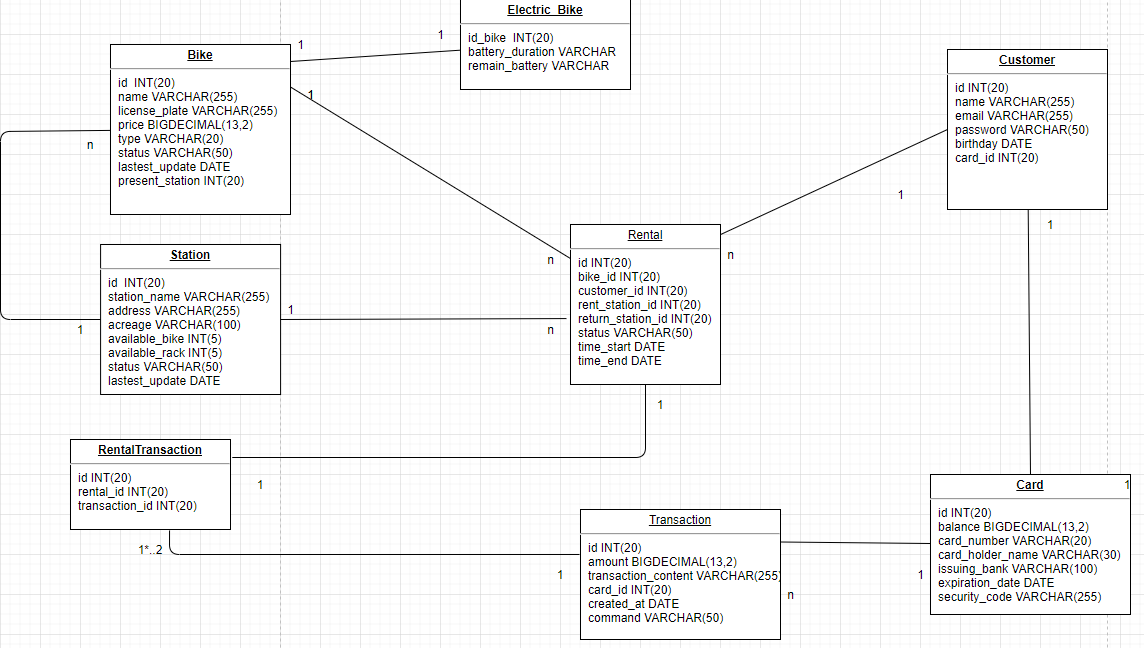
Hình … biểu diễn sơ đồ thực thể liên kết của database trong hệ thống. Bao gồm các quan hệ:

* Customer có Card
* Customer, Card, Bike kết hợp lại sẽ tạo thành một nhóm hoạt động thuê xe, trả xe và tạo Transaction.
* Bike sẽ nằm trong Station, 1 station có nhiều Bike.

### Database Design

#### Database Management Systems

#### Logical Data Model



Hình .*Model diagrams*

Mô tả:

Bike: lưu trữ thông tin chung của các loại xe đạp. Khi xe đạp có một tính năng mới nổi bật thì có thể tạo thêm các bảng Bike mới như bảng Electric\_Bike có khóa ngoại tham chiếu đến bảng Bike. Đặc biệt ở Bike, có trường status biểu thị tình trạng xe là: đang thuê (RENTING\_BIKE) hay có thể thuê bất cứ lúc nào (AVAILABLE). Trường present\_station là khóa ngoại tham chiếu tới id của bảng Station thể hiện thông tin trạm xe hiện tại mà Bike đang được đặt.

Station: lưu trữ thông tin của trạm xe.

Customer: lưu trữ thông tin khách hàng

Card: lưu trữ thông tin credit card của khách hàng thuê xe

Transaction: lưu trữ các thông tin giao dịch được thực hiện.

Rental: bảng này chứa thông tin thuê xe của khách hàng bao gồm id của xe được thuê, id khách hàng, id của trạm mà được lấy xe ra, id của trạm mà xe được trả về, status hiện tại (đang thuê hay đã trả xe), thời gian bắt đầu tính thuê xe, thời gian trả xe.

RentalTransaction: bảng này là bảng quan hệ giữa hai bảng Rental và Transaction, biểu thị quan hệ một nhiều (một record của Rental sẽ có tối đa 2 record của Transaction do trong mỗi hoạt động thuê xe thì chỉ xảy ra 2 giao dịch là lúc đặt cọc xe và lúc thanh toán).

#### Physical Data Model

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| *#* | *PK* | *FK* | *Tên cột* | *Kiểu dữ liệu* | *Giá trị mặc định* | *Bắt buộc* | *Mô tả* |
| 1 | x |  | id | int |  | x | Id của xe |
| 2 |  |  | bike\_name | text |  |  | Tên gọi của xe |
| 3 |  |  | license\_plate | text |  |  | Biến số xe |
| 4 |  |  | price | int |  |  | Giá thuê xe |
| 5 |  |  | bike\_type | text |  |  | Loại xe |
| 6 |  |  | status | text |  |  | Trạng thái xe |
| 7 |  |  | latest\_update | date |  |  | Ngày cập nhập cuối cùng |
| 8 |  | x | present\_station | int |  |  | Id trạm xe hiện tại |

Bảng *. Chi tiết thiết kế bảng Bikes*

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| *#* | *PK* | *FK* | *Tên cột* | *Kiểu dữ liệu* | *Giá trị mặc định* | *Bắt buộc* | *Mô tả* |
| 1 | x |  | id | int |  | x | Id của xe |
| 2 |  |  | station\_name | text |  |  | Tên gọi của trạm xe |
| 3 |  |  | address | text |  |  | Địa chỉ trạm xe |
| 4 |  |  | acreage | text |  |  | Diện tích trạm xe |
| 5 |  |  | available\_bike | int |  |  | Số lượng xe sẵn sàng |
| 6 |  |  | available\_rack | int |  |  | Số lượng chỗ trống |
| 7 |  |  | latest\_update | date |  |  | Ngày cập nhập cuối cùng |
| 8 |  |  | status | text |  |  | Trạng thái trạm xe |

Bảng . *Chi tiết thiết kế bảng Stations*

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| *#* | *PK* | *FK* | *Tên cột* | *Kiểu dữ liệu* | *Giá trị mặc định* | *Bắt buộc* | *Mô tả* |
| 1 | x |  | id | int |  | x | Id của lần thuê |
| 2 |  | x | bike\_id | int |  |  | Id của xe cho thuê |
| 3 |  | x | customer\_id | int |  |  | Id của người thuê |
| 4 |  |  | rent\_station\_id | int |  |  | Id trạm xe cho thuê |
| 5 |  |  | return\_station\_id | int |  |  | Id trạm xe trả |
| 6 |  |  | time\_start | timestamp |  |  | Thời gian bắt đầu thuê |
| 7 |  |  | time\_end | timestamp |  |  | Thời gian kết thuc thuê |
| 8 |  |  | status | text |  |  | Trạng thái lần thuê |

Bảng . *Chi tiết thiết kế bảng Rental*

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| *#* | *PK* | *FK* | *Tên cột* | *Kiểu dữ liệu* | *Giá trị mặc định* | *Bắt buộc* | *Mô tả* |
| 1 | x |  | id | int |  | x | Id của xe |
| 2 |  |  | customer\_name | text |  |  | Tên gọi của khách hàng |
| 3 |  |  | email | text |  |  | Địa chỉ email khách hàng |
| 4 |  |  | customer\_password | text |  |  | Mật khẩu khách hàng |
| 5 |  |  | birthday | date |  |  | Ngày sinh khách hàng |
| 6 |  | x | card\_id | int |  |  | Mã số thẻ |

Bảng . *Chi tiết thiết kế bảng Customer*

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| *#* | *PK* | *FK* | *Tên cột* | *Kiểu dữ liệu* | *Giá trị mặc định* | *Bắt buộc* | *Mô tả* |
| 1 | x |  | id | int |  | x | Id thẻ |
| 2 |  |  | balance | int |  |  | Số dư trong thẻ |
| 3 |  |  | card\_code | text |  |  | Số thẻ |
| 4 |  |  | owner | text |  |  | Tên chủ sở hữu |
| 5 |  |  | issuing\_bank | text |  |  | Tên ngân hàng |
| 6 |  |  | expiration\_date | date |  |  | Ngày hết hạn thẻ |
| 7 |  |  | security\_code | text |  |  | Mã bảo mật của thẻ |

Bảng . *Chi tiết thiết kế bảng Cards*

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| *#* | *PK* | *FK* | *Tên cột* | *Kiểu dữ liệu* | *Giá trị mặc định* | *Bắt buộc* | *Mô tả* |
| 1 | x | x | id\_bike | int |  | x | Id của xe điện |
| 2 |  |  | battery\_duration | text |  |  | Thời lượng pin |
| 3 |  |  | remain\_battery | text |  |  | Lượng pin còn lại |

Bảng . *Chi tiết thiết kế bảng Electric\_bike*

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| *#* | *PK* | *FK* | *Tên cột* | *Kiểu dữ liệu* | *Giá trị mặc định* | *Bắt buộc* | *Mô tả* |
| 1 | x |  | id | int |  | x | Id của transaction |
| 2 |  | x | rental\_id | int |  |  | Mã số của lần thuê |
| 3 |  | x | transaction\_id | int |  |  | Id giao dịch |

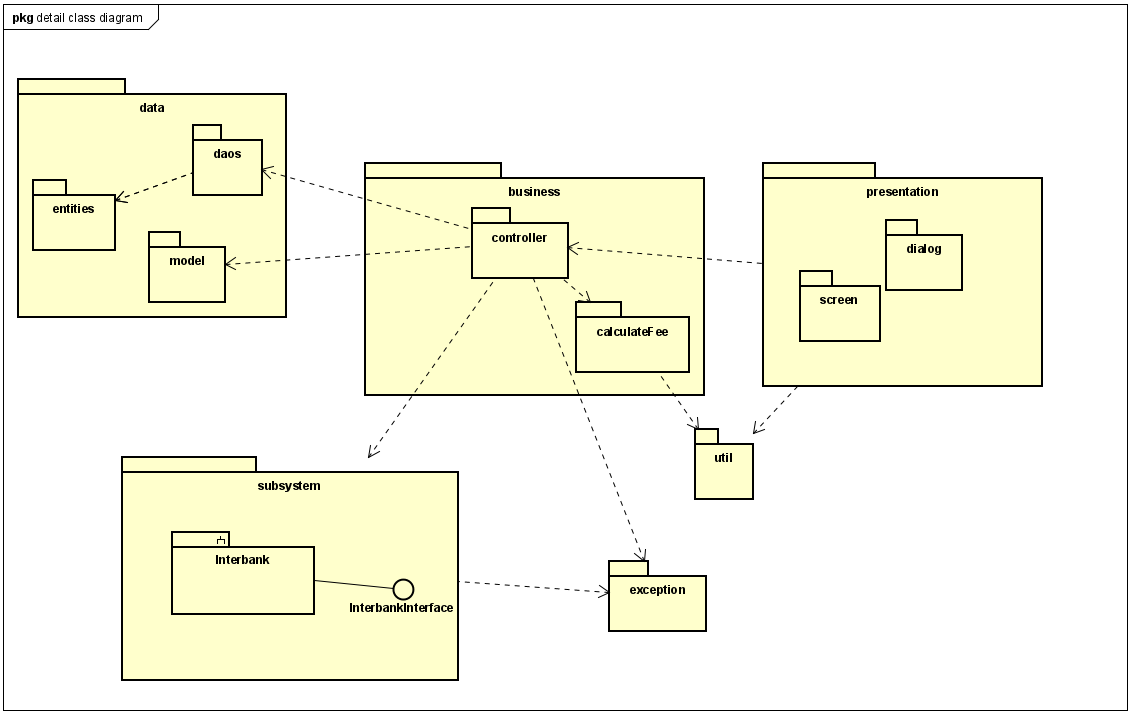
Bảng . *Chi tiết thiết kế bảng Rental\_Transaction*

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| *#* | *PK* | *FK* | *Tên cột* | *Kiểu dữ liệu* | *Giá trị mặc định* | *Bắt buộc* | *Mô tả* |
| 1 | x |  | id | int |  | x | Id giao dịch |
| 2 |  |  | amount | int |  |  | Số tiền |
| 3 |  |  | transaction\_content | text |  |  | Nội dung giao dịch |
| 4 |  | x | card\_code | text |  |  | Mã số thẻ |
| 5 |  |  | create\_at | timestamp |  |  | Thời điểm khởi tạo |
| 6 |  |  | command | text |  |  | Điều khiển |
| 7 |  |  | api\_id | text |  |  | Mã số api |

Bảng . *Chi tiết thiết kế bảng Transactions*

## Class Design

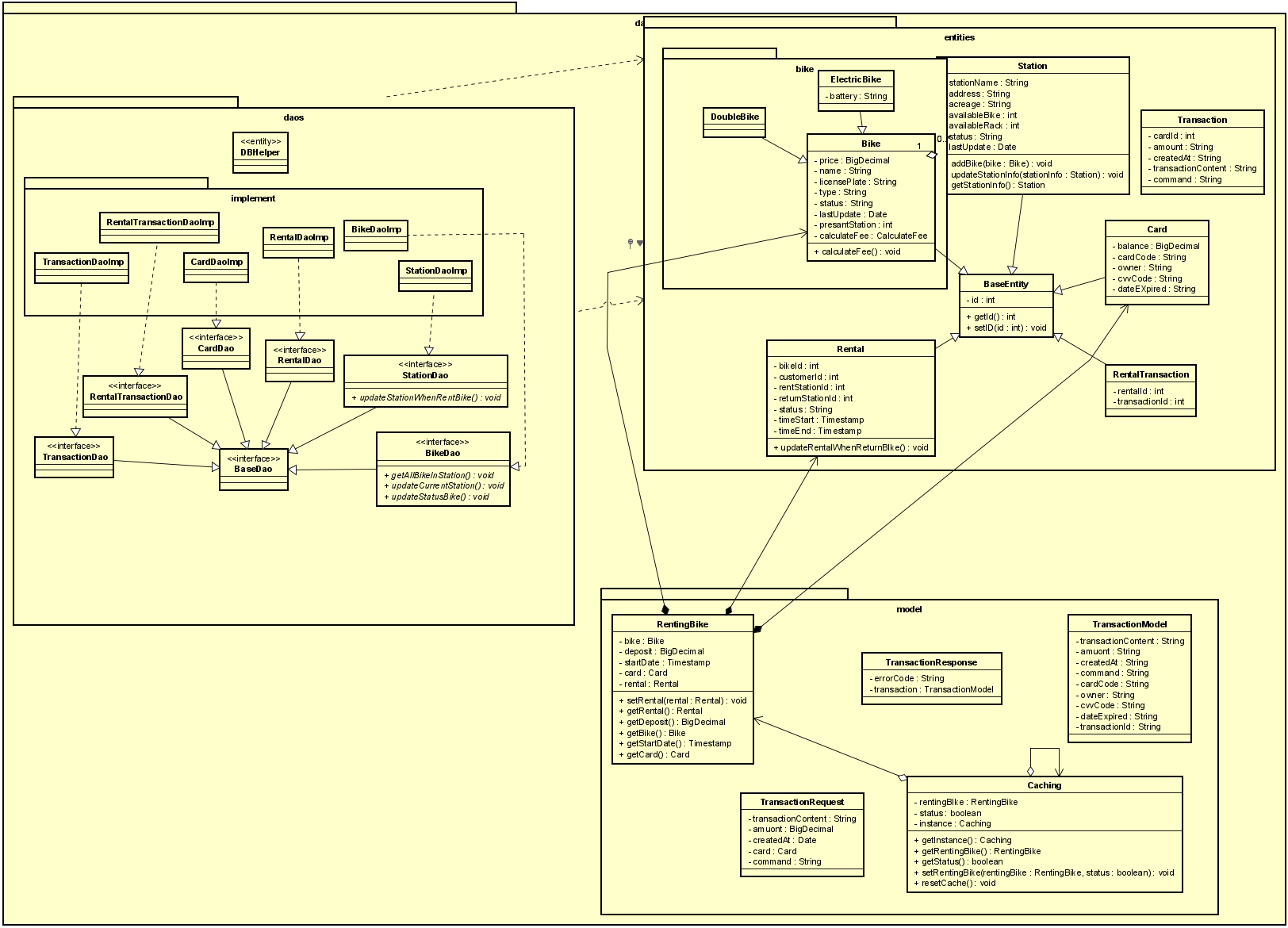
### General Class Diagram



Hình . *Biểu đồ mô tả phụ thuộc chung giữa các package trong hệ thống*

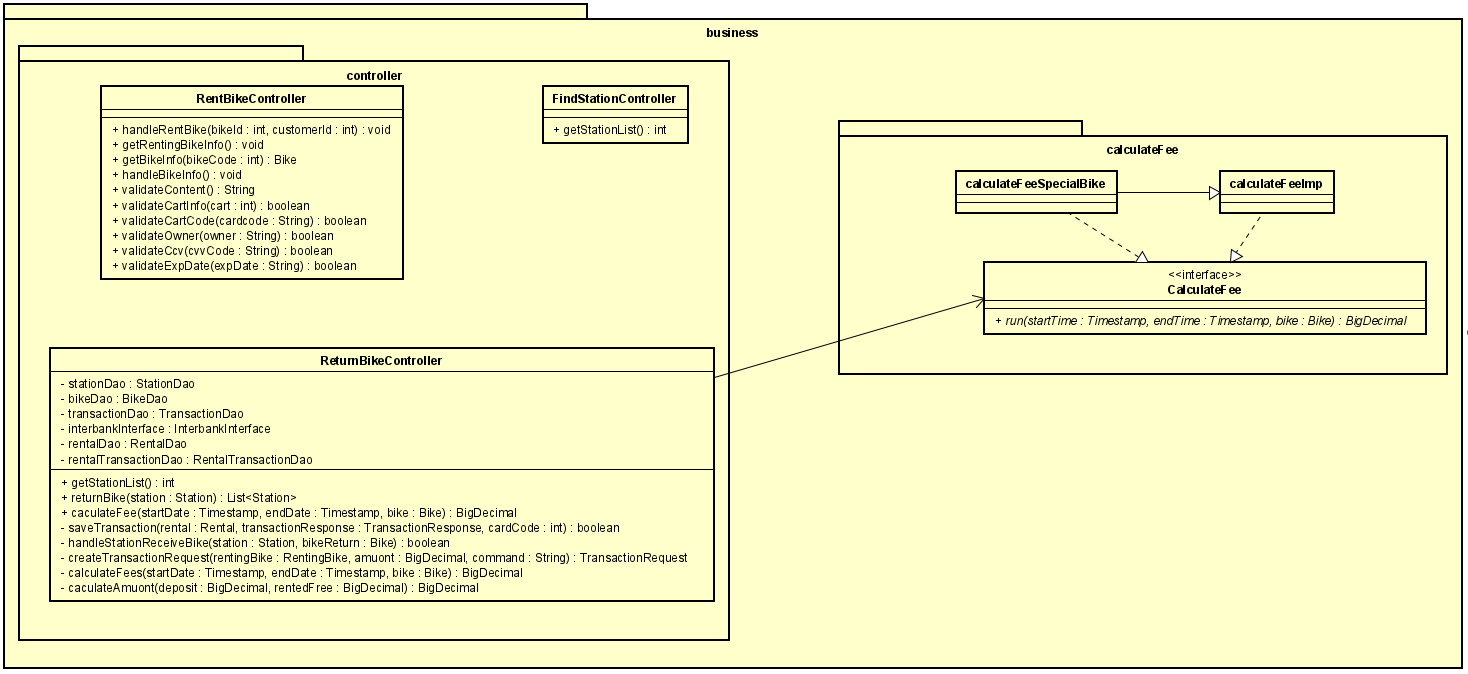
### Biểu đồ lớp

#### Biểu đồ lớp cho package Data



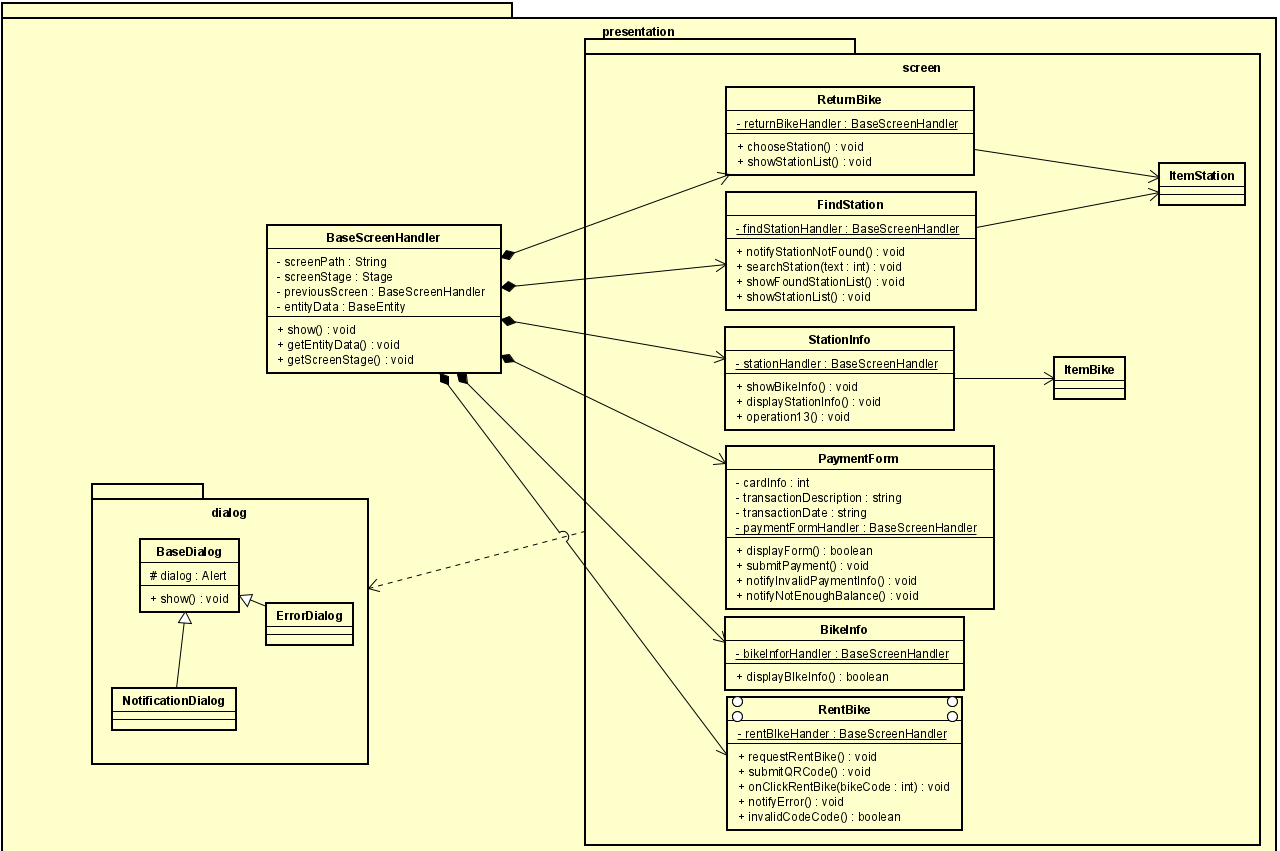
Hình . *Biểu đồ lớp của package Data*

#### Biểu đồ lớp cho package Business



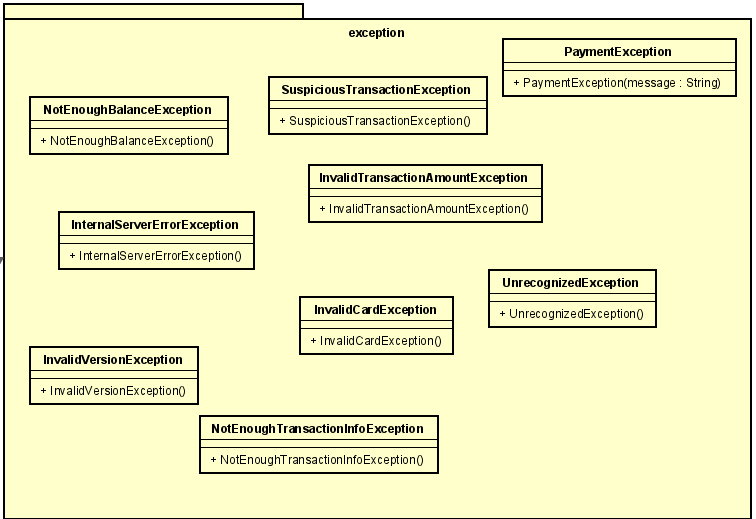
Hình . *Biểu đồ lớp cho package Business*

#### Class Diagram for Package Presentation



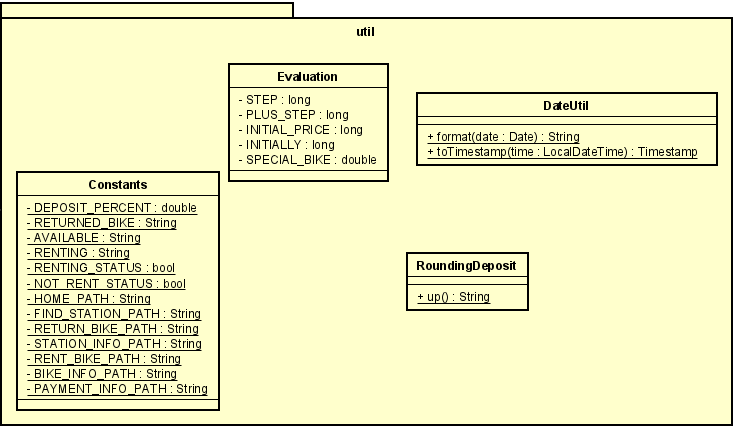
Hình . *Biểu đồ lớp cho package Presentation*

#### Class Diagram for Package Exception



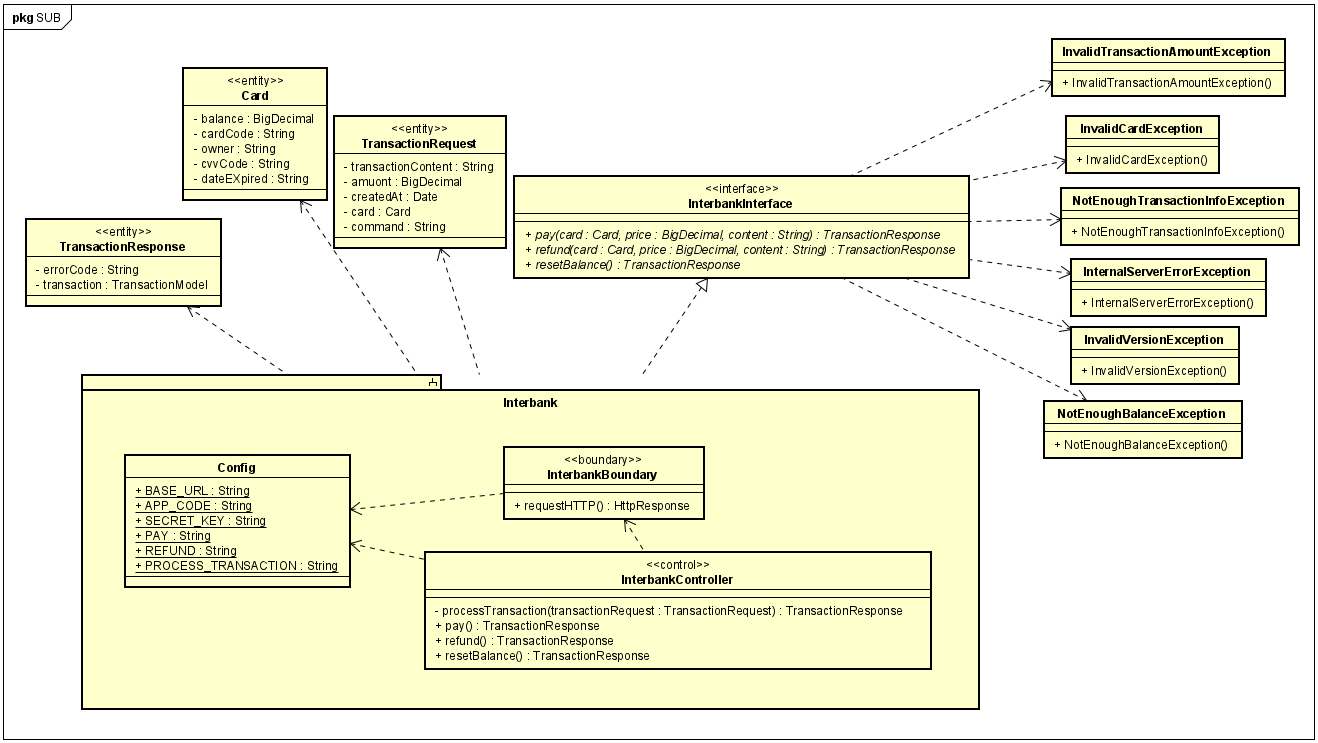
Hình . *Biểu đồ lớp cho package Exception*

#### Class Diagram for Package Util



Hình . *Biểu đồ lớp cho package Util*

#### Class Diagram for Subsystem InterbankAPI



Hình . *Biểu đồ lớp cho subsystem Interbank*

### Class Design

#### Class Rentbike Controller

**Attribute**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| *#* | *Name* | *Kiểu dữ liệu* | *Giá trị mặc định* | *Mô tả* |
| 1 | stationDao | StationDao | null | Đối tượng dùng để truy vấn dữ liệu bảng station |
| 2 | bikeDao | BikeDao | null | Đối tượng dùng để truy vấn dữ liệu bảng bike |
| 3 | transactionDao | TransactionDao | null | Đối tượng dùng để truy vấn dữ liệu bảng transaction |
| 4 | interBankApiSystem | InterBankApiSystem | null | Đối tượng dùng để truy vấn api interbank |
| 5 | rentalDao | RentalDao | null | Đối tượng dùng để truy vấn dữ liệu bảng rental |
| 6 | rentalTransactionDao | RentalTransactionDao | null | Đối tượng dùng để truy vấn dữ liệu bảng rental transaction |

**Operation**

|  |  |  |  |
| --- | --- | --- | --- |
| *#* | *Name* | *Return type* | *Description (purpose)* |
| 1 | getBikeInfo | bikeCode | Lấy thông tin xe hiện tại dựa vào mã xe |
| 2 | handleRentBike | void | Thực hiện các nghiệp vụ thuê xe (cho tầng trên gọi) |
| 3 | saveRentingData | RentingBike | Lưu các thông tin liên quan đến nghiệp vụ thuê xe vào cơ sở dự liệu |
| 4 | validateCardInfo | boolean | Validate thông tin thẻ |
| 5 | validateCardCode |
| 6 | validateExpDate |
| 7 | validateOwner |
| 8 | validateCcv |

*Exception*:

* Ecobike Exception: Khi nhập mã xe, thông tin không hợp lệ
* Payment Exception: Khi api trả về không thành công
* SQL Exception: Khi truy vấn cơ sở dữ liệu không thành công

#### Class ReturnbikeController

**Attribute**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| *#* | *Name* | *Kiểu dữ liệu* | *Giá trị mặc định* | *Mô tả* |
| 1 | stationDao | StationDao | Null | Đối tượng dùng để truy vấn dữ liệu bảng station |
| 2 | bikeDao | BikeDao | Null | Đối tượng dùng để truy vấn dữ liệu bảng bike |
| 3 | transactionDao | TransactionDao | Null | Đối tượng dùng để truy vấn dữ liệu bảng bike |
| 4 | rentalDao | RentalDao | Null | Đối tượng dùng để truy vấn dữ liệu bảng bike |
| 5 | interbankInterface | InterbankInterface | Null | Đối tượng dùng để truy vấn dữ liệu bảng bike |
| 6 | calculateFee | CalculateFee | Null | Đối tượng dùng để truy vấn dữ liệu bảng bike |
| 7 | rentalTransactionFee | RentalTransactionFee | Null | Đối tượng dùng để truy vấn dữ liệu bảng bike |

**Operation**

|  |  |  |  |
| --- | --- | --- | --- |
| *#* | *Name* | *Return type* | *Description (purpose)* |
| 1 | returnBike(station) | void | Thực hiện các nghiệp vụ trả xe |
| 2 | saveTransaction | boolean | Thực hiện lưu thông tin giao dịch |
| 3 | handleStationRecieveBike | boolean | Thực hiện update thông tin station khi trả xe |
| 4 | calculateAmount | BigDecimal | Tính toán phí giao dịch |

*Exception*:

# Design Considerations

## Mục tiêu và phương châm

Mục tiêu:

* Hướng đến khả năng tái sử dụng mã nguồn một cách tối ưu nhất.
* Các module trong hệ thống kết nối với nhau và đạt được hai tính chất quan trọng trong thiết kế là Low coupling và High cohension.
* Áp dụng tối đa các nguyên tắc SOLID.
* Các module dễ dàng giao tiếp.
* Dễ dàng mở rộng trong tương lai.
* Dễ dàng bảo trì.
* Dễ đọc, dễ đọc.

Phương châm:

* Mã nguồn gọn gàng, sạch sẽ, tên hàm phải nói lên được chức năng của hàm, tên biến rõ ràng.

## Chiến lược

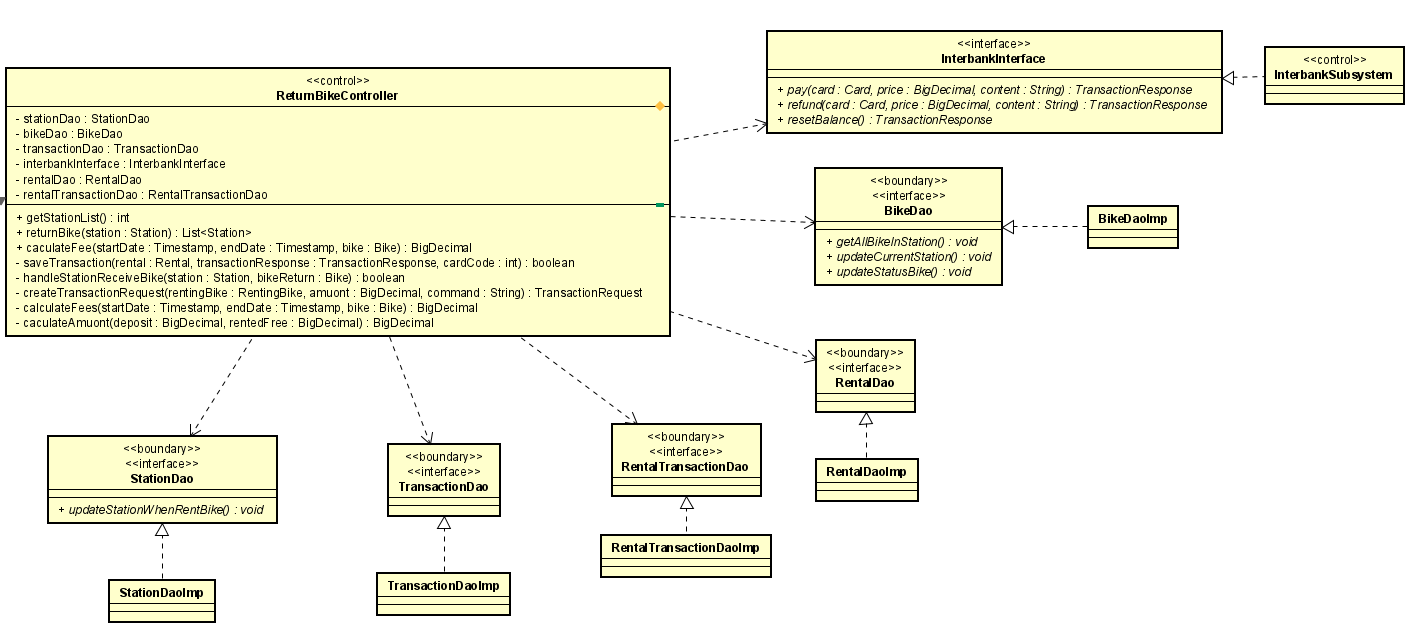
Chiến lược thiết kế:

* Ngôn ngữ sử dụng: java
* Trình quản lý và thiết lập tự động: Maven
* Sử dụng các thư viện open source để tối giản chi phí và giảm thời gian phát triển sản phẩm như: javafx open, apacheHttpClient.
* Cơ sở dữ liệu sử dụng: postgreSQL
* Công cụ: IDE Intellij, DataGrip connect DB, SceneBuilder sinh code giao diện fxml tự động khi kéo thả.
* Phần cứng: yêu cầu máy hỗ trợ cấu hình core i3 trở lên và tối thiểu ram 2GB, bộ nhớ 500GB.
* Xử lý lỗi: khi có lỗi xảy ra trong hệ thống thì sẽ được throw ra ngoài giao diện, giao diện bắt lấy lỗi và hiểu thị Alert lên cho người dùng.
* Quản lý bộ nhớ: thông tin xe đang thuê sẽ được lưu trữ trực tiếp trên ram của máy khách để tối ưu thời gian truy xuất thông tin
* Kế hoạch trong tương lai: Mở rộng thêm nhiều loại xe, nhiều kiểu station, có chức năng thay đổi cách tính giá thuê xe, có chức năng cho phía administrator quản trị, người dùng có thể nhập vào tìm kiếm bất cứ thứ gì mà mình muốn.

## Coupling và Cohension

|  |  |  |  |
| --- | --- | --- | --- |
| ***Coupling*** | *Mô tả* | *Minh chứng* | *Giải thích* |
| 1. *Uncoupled* | Cấp độ tốt nhất của Coupling, liên kết lỏng lẻo giữa các module. Ở đây, các lớp và phương thức không chia sẻ bất kỳ dữ liệu nào, độc lập hoàn toàn với nhau, chỉ đơn giản là lớp này sử dụng chức năng của lớp kia mà không phụ thuộc vào nhau. | Tiêu biểu là phần tương tác với Database. Trong Controller chỉ cần gọi ra các Interface tương ứng cần giao tiếp để thực hiện chức năng cần thiết.  (Xem hình 17 bên dưới) | Áp dụng nguyên lý Dependency Inversion vào để có được loosely coupling tốt nhất, Controller chỉ giao tiếp với các interface như BikeDao, StationDao,… trong khi việc thực thi các chức năng của các class này là một class khác mà ta có thể dễ dàng thay đổi chỉnh sửa mà không cần quan tâm đến class trong controller. |
| 1. *Data* | Khi A gọi B và truyền một giá trị hay một tập dữ liệu cần thiết cho B thì lúc này, A và B đang được ràng buộc về dữ liệu. Phần mềm đặc biệt được thiết kế nhiều bởi ràng buộc này. |  | theo như hình minh chứng, class ReturnBikeController sẽ gọi phương thức **run()** và truyền vào các giá trị cần thiết để thực hiện chức năng tính phí thuê xe. Như vậy, class returnBikeController và phương thức run() ràng buộc nhau về dữ liệu. |

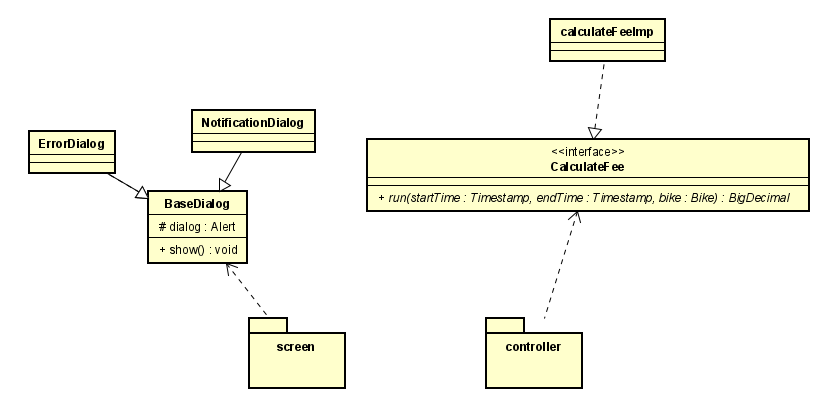
Bảng . *Mô tả concept of Coupling*

**

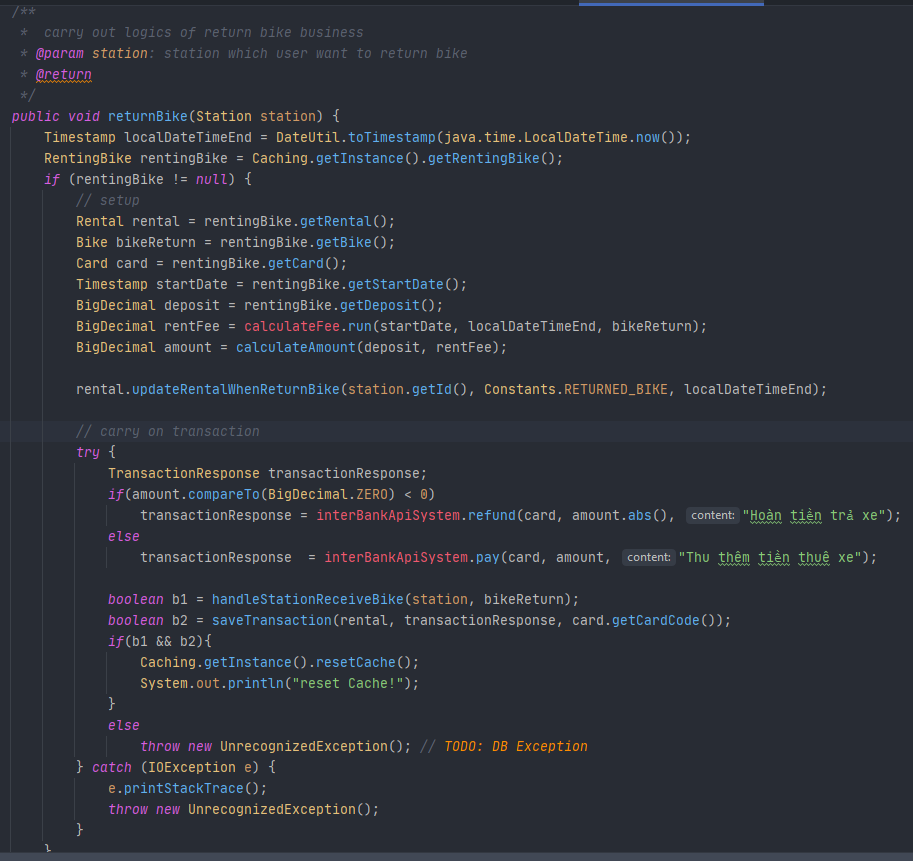
Hình . *Minh chứng Nocoupled level*

|  |  |  |  |
| --- | --- | --- | --- |
| ***Cohension*** | *Mô tả* | *Minh chứng* | *Giải thích* |
| 1. *Functional* | Nhiệm vụ của từng module, từng lớp được tách ra rõ ràng, và cụ thể. | Chức năng tính phí thuê xe, hiểu thị Alert thông báo cho người dùng. (xem hình 18) | Các chức năng con này được tách riêng ra thành một class để có thể dễ dàng nhận biết, thay đổi chức năng. |
| 1. *Sequential* | Các phần tử con được nhóm lại khi mà đầu ra của phần tử này là đầu vào của phần tử khác. | Bên trong phương thức returnBike(station) có 1 loạt các hàm con được gọi và kết quả của các hàm trước được sử dụng như đầu vào của hàm tiếp theo.  (xem hình 19) | Một loạt các hành động xảy ra theo trình tư được gọi ra bên trong module returnBike, các hành động này là các phương thức có đầu vào và đầu ra cụ thể, có tên nói lên rõ ràng chức năng, nhiệm vụ của từng hành động giúp chương trình dễ đọc, dễ bảo trì. |

Bảng . *Mô tả concept of cohension*



Hình . *Minh chứng cho Functional cohension*



Hình . *Minh chứng cho Sequential cohension*

## Design Principles

*Áp dụng các nguyên tắc SOLID:*

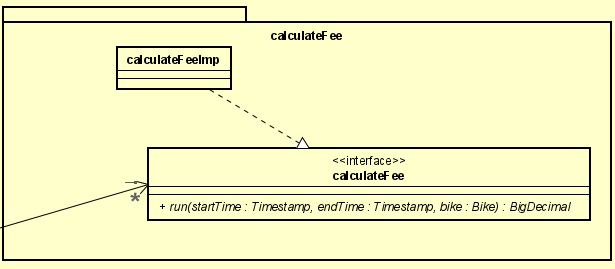
* *Single responsibility principle*

*Source code ngay từ đầu đã được triển khai để thực hiện theo nguyên tắc này, khi mà các chức năng nằm hoàn toàn về mặt logic được đặt ở Controller, chức năng lưu trữ thông tin được lưu trữ trong model và entity, chức năng tương tác với cơ sở dữ liệu thì sử dụng Data access object, chức năng tính phí thuê xe hay hiển thị thông báo cũng được tách ra rõ ràng và tường minh nhất để sau này có thể bảo trì dễ dàng.*

* *Open/closed principle*

*Trong source code, có rất nhiều vị trí áp dụng nguyên tắc này:*

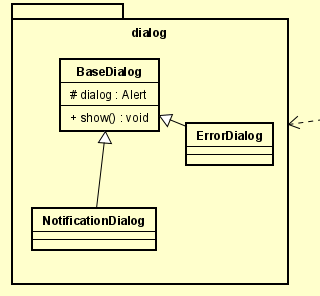
*Chẳng hạn như chức năng tính phí thuê xe được tách ra thành class riêng biệt. Sử dụng interface để giao tiếp. Khi cần thay đổi chức năng tính phí thì chỉ cần tạo 1 class khác và implement interface là xong (hình 17)*



Hình . *Ví dụ 1 áp dụng nguyên tắc Open/closed*

*Hoặc là khi có thêm loại xe (bike) mới, chỉ cần thêm lớp mới biểu thị loại bike cần thêm kế thừa Bike base có sẵn và override lại những thứ cần thiết.*

*Hiển thị Dialog thông báo cho người dùng cũng được thiết kế theo nguyên lý này (xem hình dưới)*

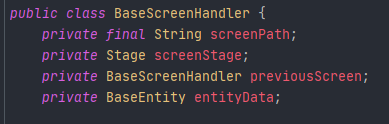


Hình . *Ví dụ 2 áp dụng nguyên tắc Open/Closed*

* *Liskov substitution principle*

*Trong source code, các class đều không vi phạm tính chất này. Khi mà các class con đều kế thừa class cha và sử dụng xuyên suốt không xảy ra lỗi.*

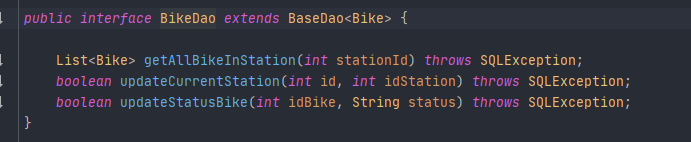
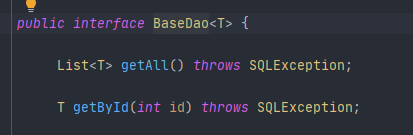
*Điển hình là các lớp entity kế thừa lớp BaseEntity, vì thế, bên trong lớp BaseScreenHandler chỉ cần khai báo 1 thuộc tính có kiểu là BaseEntity là có thể sử dụng được như là các lớp con của nó.*



Hình . *Ví dụ áp dụng nguyên tắc Liskov*

* *Interface segregation principle*

*Các interface đều được chia nhỏ ra theo các thực thể đảm nhiệm tương ứng. Cụ thể trong package Daos, các chức năng chung đều được đặt trong 1 class chung là BaseDao, còn lại nếu entity nào có chức năng cần truy vấn riêng, khác với những thực thể khác thì chỉ cần tạo một phương thức ở interface đó (xem hình 23).*



Hình . *Ví dụ áp dụng nguyên tắc Interface segregation*

* *Dependency inversion principle*

*Nguyên lý này đã được áp dụng khi xử lý subsytem trong hệ thống, và giao tiếp với database. Để kết nối với database hoặc với subsystem ta chỉ cần quan tâm tới các interface như BikeDao, StationDao, hay InterbankInterface mà không cần quan tâm đến những cái implement nó.*

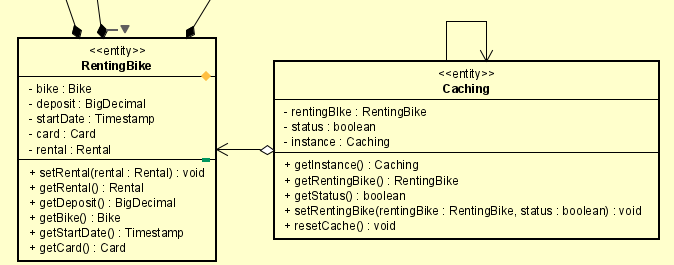
## Design Patterns

### Singleton

*Project lưu trữ thông tin thuê xe tồn tại xuyên suốt khi chương trình chạy, và chỉ được biểu diễn duy nhất trong một instance. Giống như lưu cache trên server thì singleton được sử dụng để tạo ra 1 thể hiện duy nhất lưu trữ thông tin (trạng thái của người dùng và thông tin về xe đang thuê)*

*Cách triển khai: lớp Caching được tạo ra và chỉ có 1 thể hiện duy nhất (dùng private Constructor) có 2 thuộc tính:*

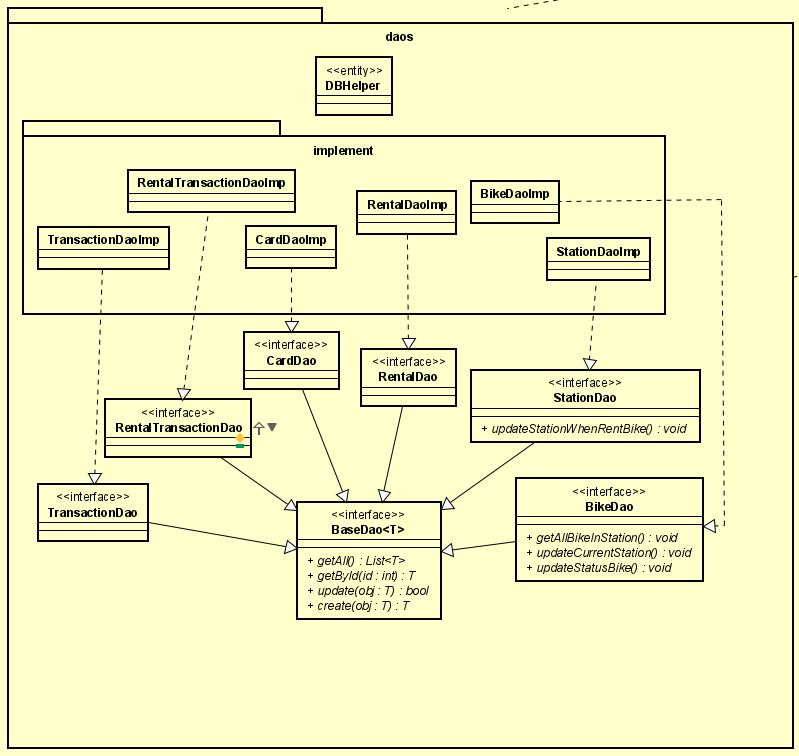
* *RentingBike lưu các thông tin về Bike, sự thuê xe, card*
* *Status: trạng thái hiện tại có đang thuê xe hay không*



Hình . *Ví dụ áp dụng Design Pattern Singleton*

### Data access object pattern

Mục đích sử dụng của pattern này trong project là để tách riêng logic của phần lưu trữ dữ liệu (entity) và phần truy cập vào cơ sở dữ liệu (data access object). Bằng cách này, 2 thành phần lưu trữ và truy cập dữ liệu được tách ra rõ ràng, các thành phần của tầng business sẽ chỉ cần tương tác lấy dữ liệu từ database thông qua các interface xxxDao tương ứng. Đồng thời, pattern này giúp đóng gói code lại để thực hiện các thao tác CRUD chung của các thực thể, tránh được việc thực thi riêng lẻ trong từng thành phần khác nhau của ứng dụng và sự không đồng nhất của cách gọi phương thức. Sau này, nếu có cần thay đổi database thì chỉ cần tạo các class khác implement các xxxDao là được.



Hình . *Áp dụng Dao pattern trong project*