TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI

Viện Công nghệ thông tin và Truyền thông

Tài liệu thiết kế phần mềm

(Software Design Document – SDD)

PHẦN MỀM THUÊ XE ĐẠP THEO GIỜ

Môn: Thiết kế và xây dựng phần mềm

Nhóm 6

Phạm Trung Kiên 20170088

Phạm Minh Khiêm 20170084

Lê Vũ Lợi 20173240

*Hà Nội, ngày 8 tháng 10 năm 2020*

Table of Contents

Table of Contents 1

1 Introduction 3

1.1 Objective 3

1.2 Scope 3

1.3 Glossary 3

1.4 References 3

2 Overall Description 4

2.1 General Overview 4

2.2 Assumptions/Constraints/Risks 4

2.2.1 Assumptions 4

2.2.2 Constraints 4

2.2.3 Risks 5

3 System Architecture and Architecture Design 6

3.1 Architectural Patterns 6

3.2 Interaction Diagrams 6

3.3 Analysis Class Diagrams 6

3.4 Unified Analysis Class Diagram 6

3.5 Security Software Architecture 6

4 Detailed Design 7

4.1 User Interface Design 7

4.1.1 Screen Configuration Standardization 7

4.1.2 Screen Transition Diagrams 7

4.1.3 Screen Specifications 7

4.2 Data Modeling 7

4.2.1 Conceptual Data Modeling 7

4.2.2 Database Design 7

4.3 Non-Database Management System Files 8

4.4 Class Design 8

4.4.1 General Class Diagram 8

4.4.2 Class Diagrams 8

4.4.3 Class Design 8

5 Design Considerations 10

5.1 Goals and Guidelines 10

5.2 Architectural Strategies 10

5.3 Coupling and Cohesion 11

5.4 Design Principles 11

5.5 Design Patterns 11

**List of Figures**

No table of figures entries found.

**List of Tables**

No table of figures entries found.

# Giới thiệu

## Mục đích

* Mục tiêu của tài liệu này là đưa ra thiết kế phần mềm cho ứng dụng EcoBikeRental cho thuê xe đạp theo giờ với nhiều bãi để xe để thuê/trả xe tự động trong khu đô thị Ecopark.
* Tài liệu này phục vụ cho designers, developers và testers của hệ thống có thể dễ dàng đi tới các bước tiếp theo của quy trình phát triển ứng dụng này.

## Phạm vi

Thuê xe đạp dạo quanh là một trong những dịch vụ được thu hút nhất tại khu đô thị sinh thái Ecopark. Hiện nay có 2 điểm cho thuê và đỗ xe đạp tại khu đô thị. Để dịch vụ này tiếp tục phát triển mở rộng ra, cần giải quyết khâu hạ tầng, bao gồm hệ thống làn đường cho xe đạp, điểm dừng, trông giữ, bảo quản, và đặc biệt là hệ thống thông tin thuê xe và trả xe tự động có thể hoạt động 24/7

## Từ điển thuật ngữ

|  |  |  |
| --- | --- | --- |
| STT | Thuật ngữ, từ viêt tắt | Giải thích |
| 1 | API | viết tắt của Application Programming Interface, là phần mềm trung gian cho phép kết nối 2 ứng dụng với nhau |
|  |  |  |
|  |  |  |

## Tài liệu tham khảo

*.*

|  |  |
| --- | --- |
| STT | Tên tài liệu |
| 1 | D. Budgen. Software Design, 2nd Edition. Addison-Wesley. 2004 |
|  |  |
|  |  |

# Mô tả tổng quan

## Tổng quan

*<Briefly introduce the system context and the basic design approach or organization. Provide a brief overview of the system and software architectures and the design goals. Include the high-level context diagram(s) for the system and subsystems provided in previous documents like SRS (e.g., general use case diagram, lower-level use case diagrams, activity diagrams), updated as necessary to reflect any changes that have been made based on more current information or understanding. If the high-level context diagram has been updated, identify the changes that were made and why>*

## Giả thiết, Điều Kiện, Rủi ro

### Giả thiết

*<Describe any assumptions or dependencies regarding the system, software and its use. These may concern such issues as: related software or hardware, operating systems, end-user characteristics, and possible and/or probable changes in functionality>*

### Điều kiện

*<Describe any global limitations or constraints that have a significant impact on the design of the system’s hardware, software and/or communications, and describe the associated impact. Such constraints may be imposed by any of the following (the list is not exhaustive):*

* *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*

*>*

### Rủi ro

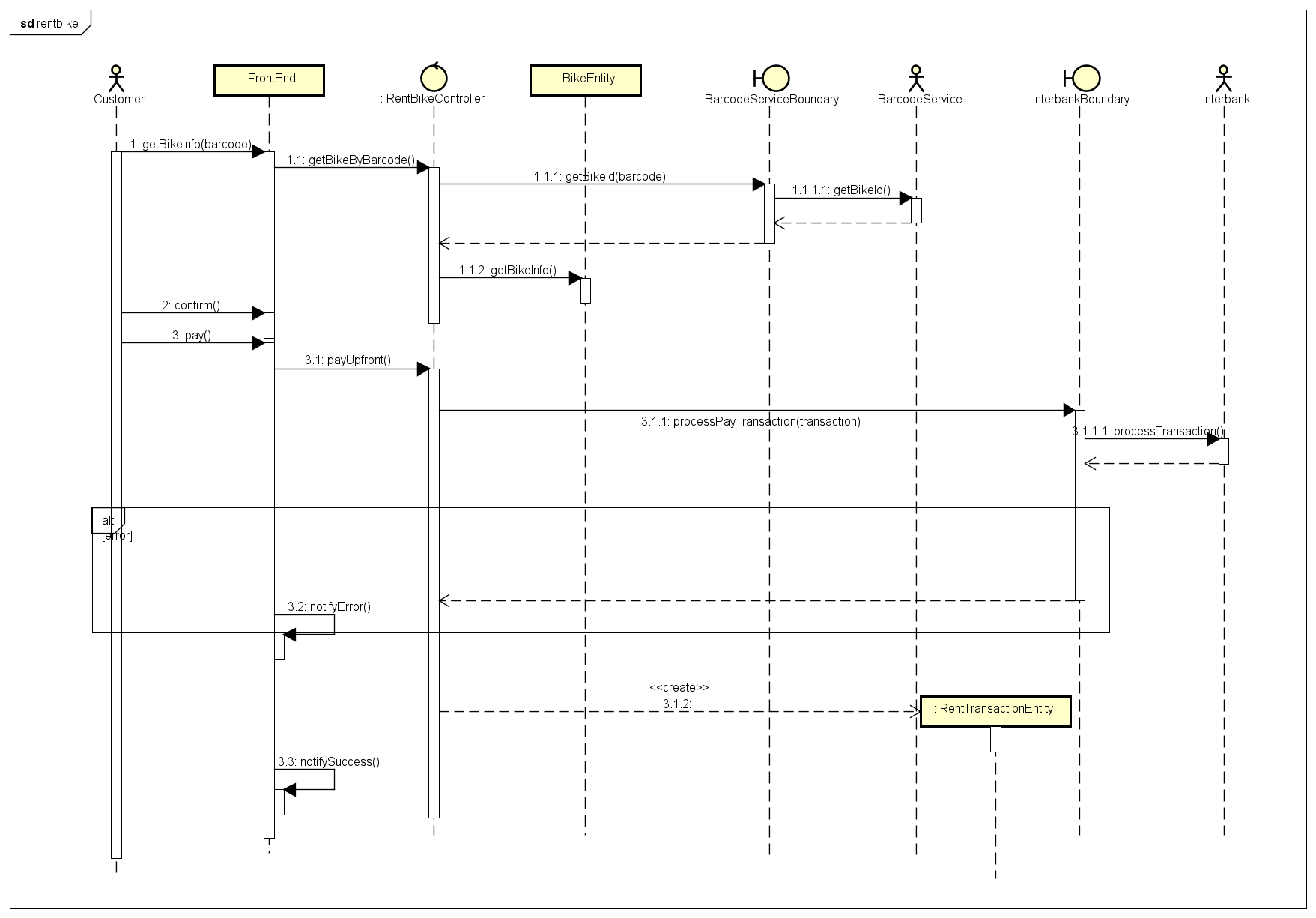
*<Describe any risks associated with the system design and proposed mitigation strategies.>*

# System Architecture and Architecture Design

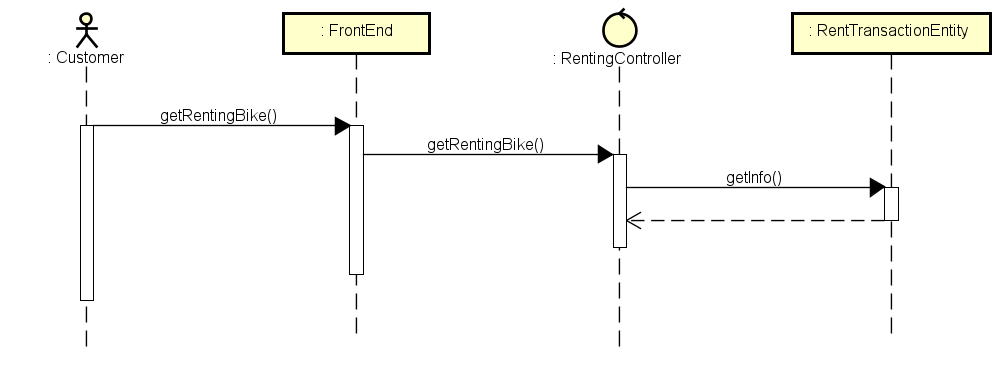
## Architectural Patterns

*Nhóm chọn thiết kế theo kiến trúc Frontend – Backend*

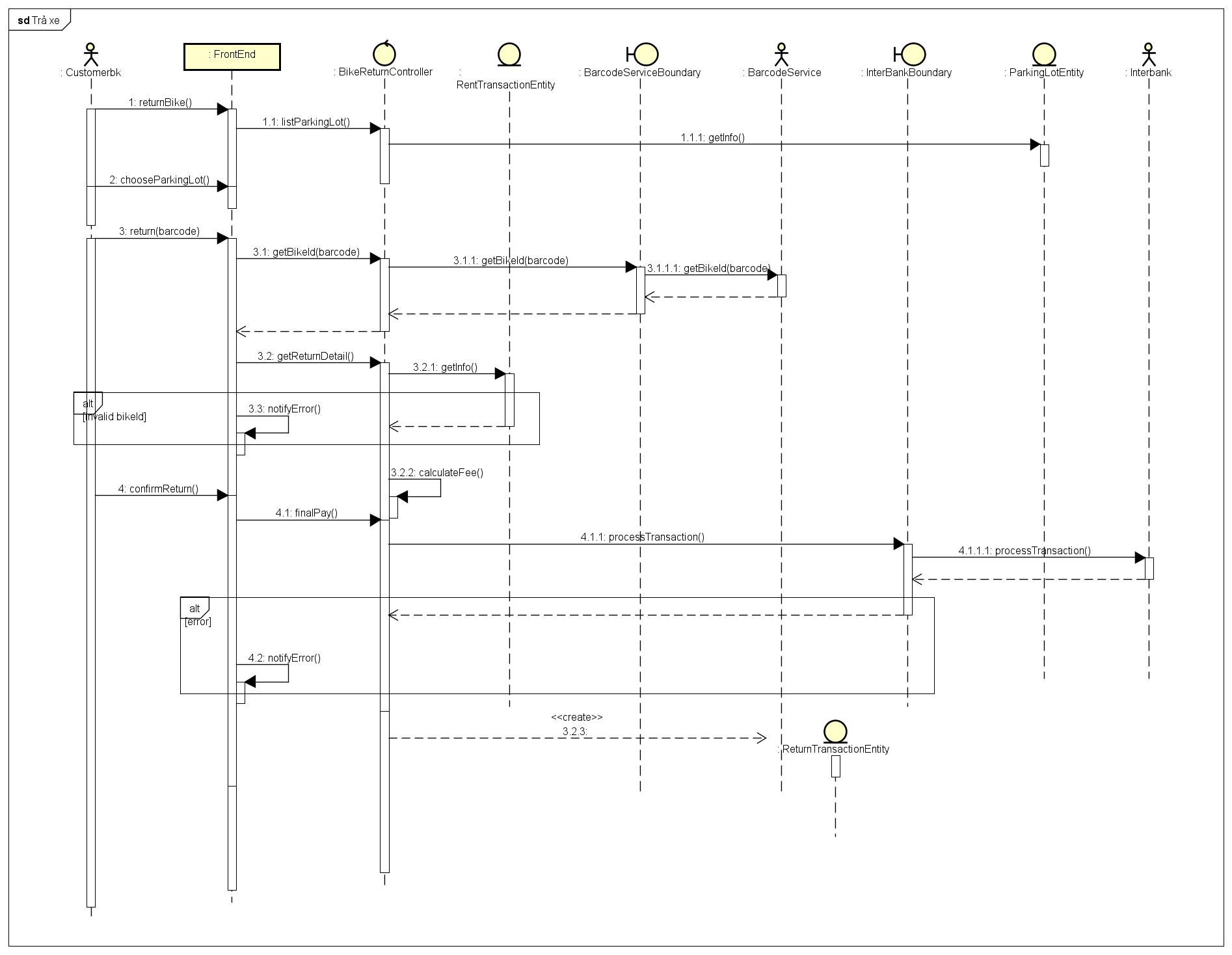
## Interaction Diagrams



Hình : Biểu đồ trình tự cho thuê xe

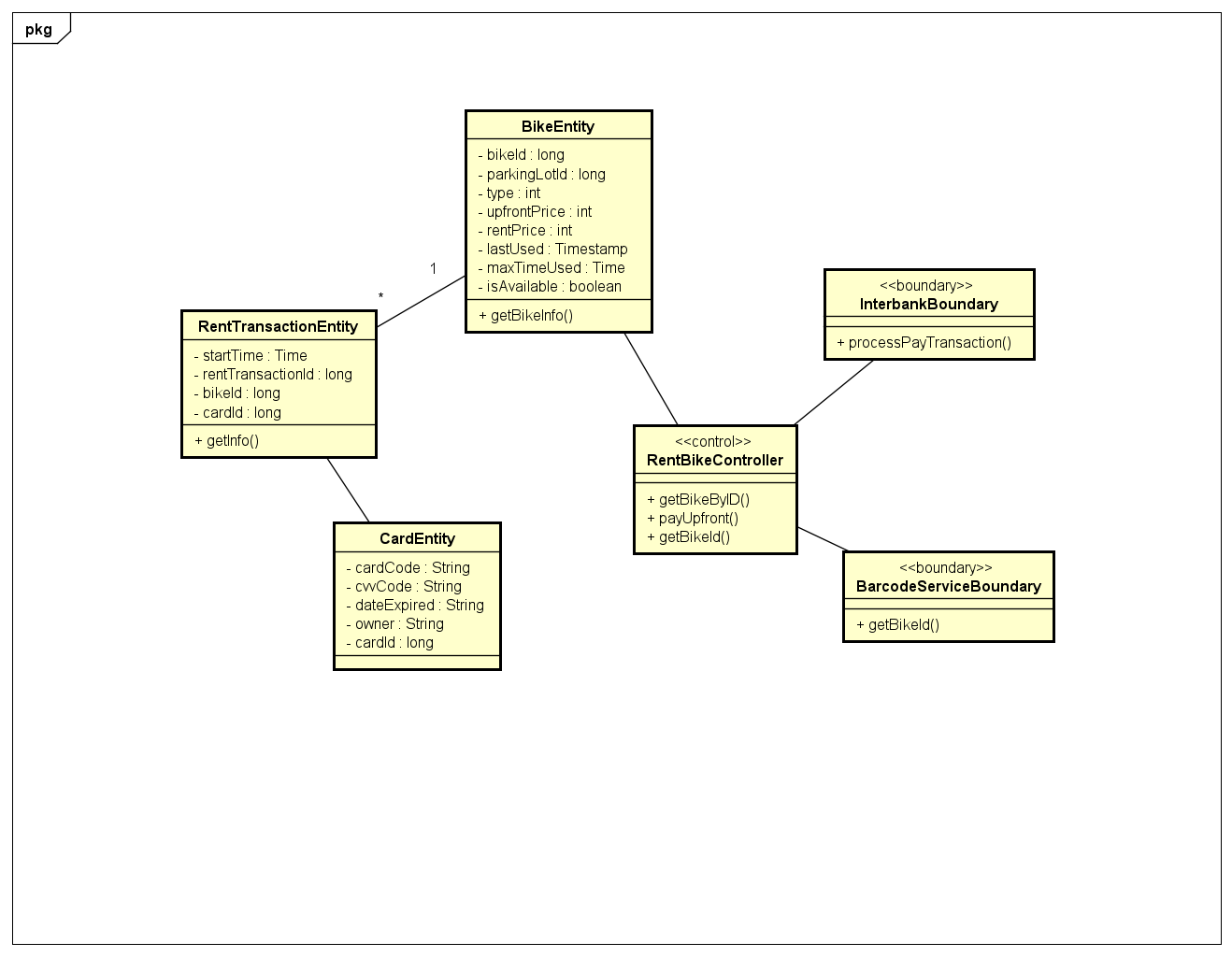


Hình : Biểu đồ trình tự cho xem xe đang thuê

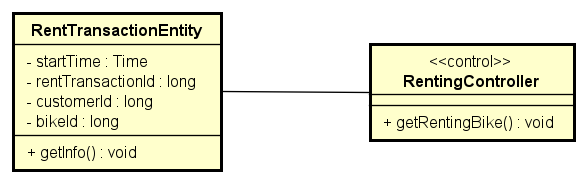


Hình : Biểu đồ trình tự cho trả xe

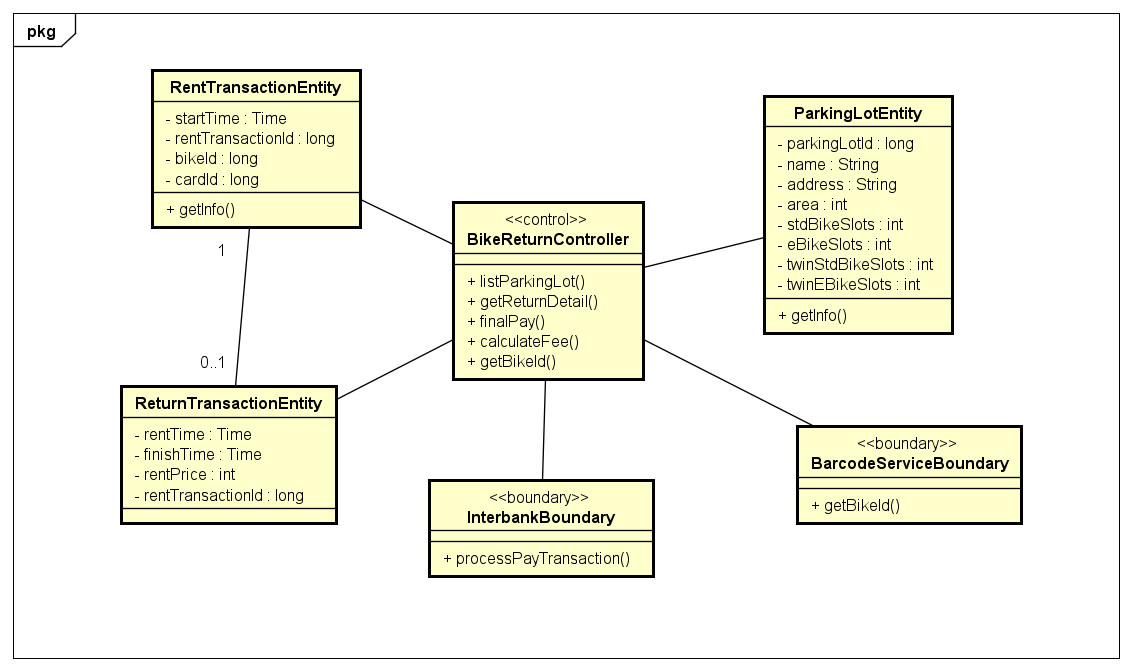
## Analysis Class Diagrams



Hình : Biểu đồ cho thuê xe

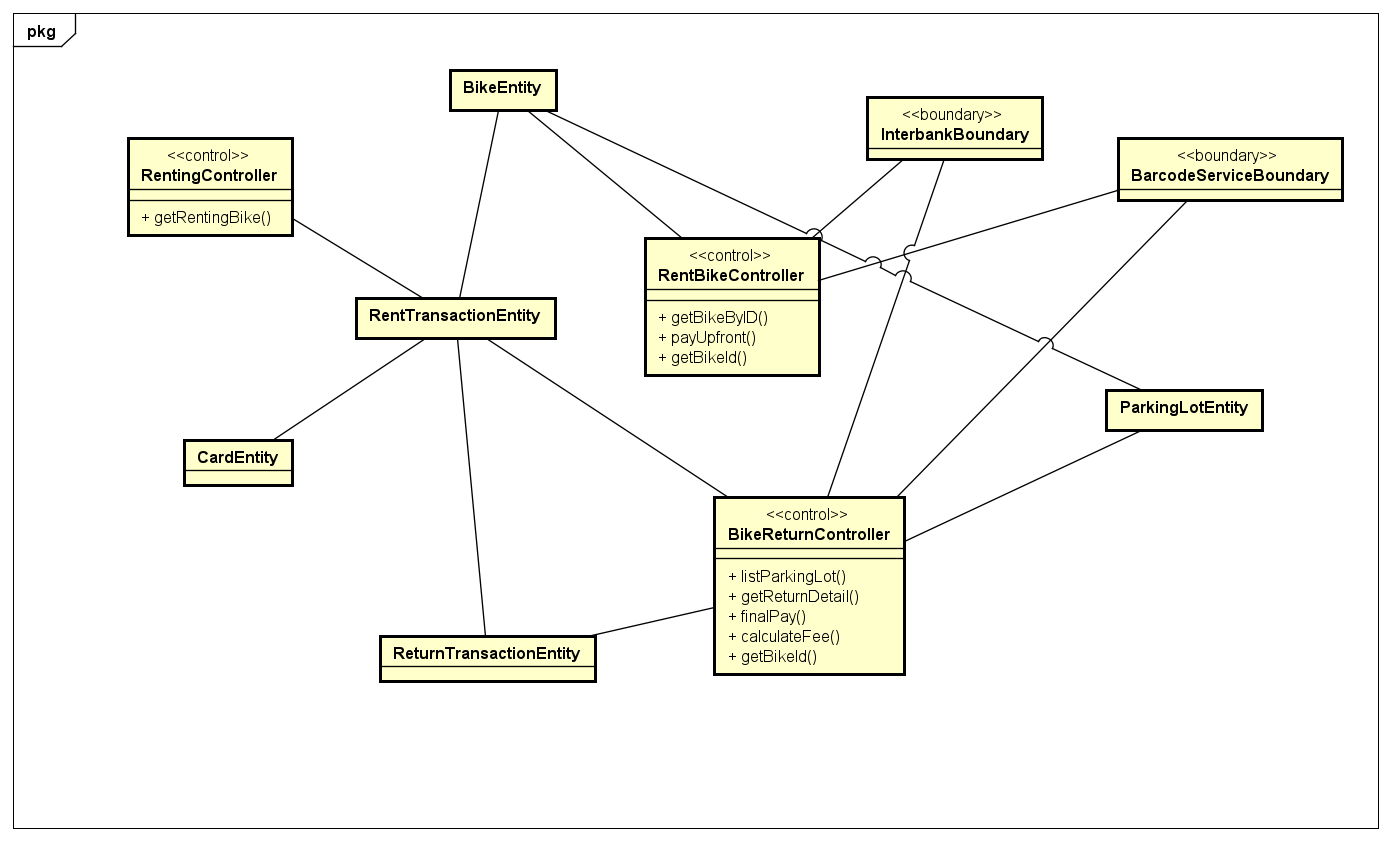


Hình : Biểu đồ cho xem xe đang thuê



Hình : Biểu đồ cho trả xe

## Unified Analysis Class Diagram



## Security Software Architecture

*<Describe the software components and configuration supporting the security and privacy of the system. Specify the architecture for (1) authentication to validate user identity before allowing access to the system;(2) authorization of users to perform functional activity once logged into the system, (3) encryption protocol to support the business risks and the nature of information, and (4) logging and auditing design, if required.>*

# Detailed Design

## User Interface Design

*<Suppose that you design a Graphical User Interface (GUI)>*

|  |  |  |  |
| --- | --- | --- | --- |
| Home screen | Barcode renting screen | Rent detail screen | Payment info screen |
|  |  |  |  |
| Confirm payment screen | Home rented screen | Renting detail screen | Return parkinglot screen |
|  |  |  |  |
| Barcode return screen | Return detailed screen | Invoice screen | Bike not found alert |
|  |  |  |  |
| Success renting alert | Rent payment failed alert | Barcode return error | Payment error alert |
|  |  |  |  |

### Screen Configuration Standardization

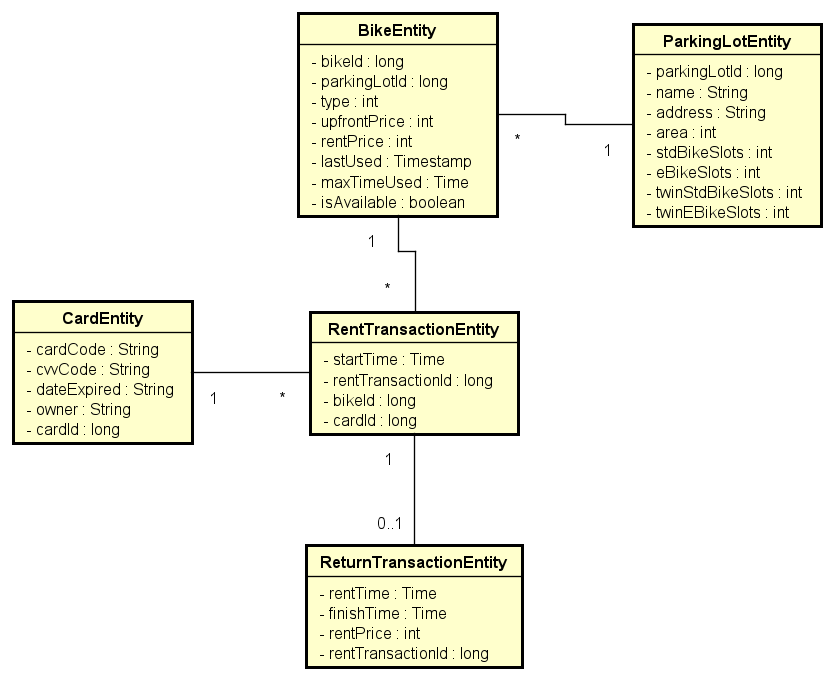
### Screen Transition Diagrams

### Screen Specifications

*<Screen images should be included in the screen specifications>*

## Data Modeling

### Conceptual Data Modeling

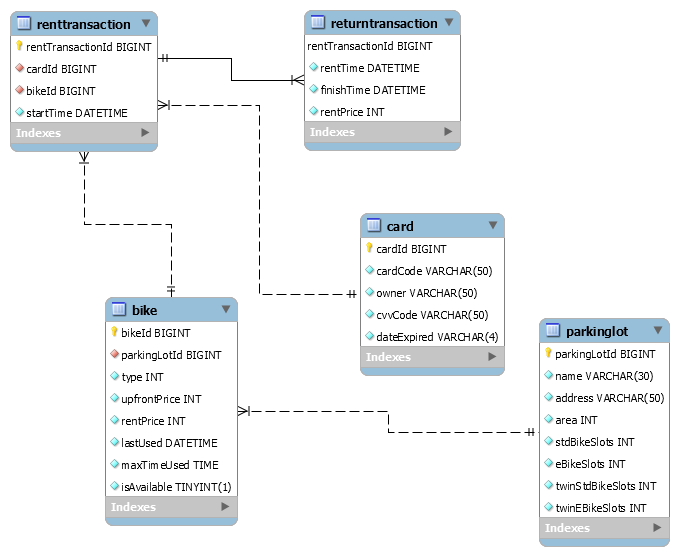


### Database Design

#### Database Management Systems

*MySQL*

#### Logical Data Model



#### Physical Data Model

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Field | Type | Null | Key | Default | Extra |
| parkingLotId | bigint | NO | PRI | NULL | auto\_increment |
| name | varchar(30) | NO |  | NULL |  |
| address | varchar(50) | NO |  | NULL |  |
| area | int | NO |  | NULL |  |
| stdBikeSlots | int | NO |  | NULL |  |
| eBikeSlots | int | NO |  | NULL |  |
| twinStdBikeSlots | int | NO |  | NULL |  |
| twinEBikeSlots | int | NO |  | NULL |  |

Bang:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Field | Type | Null | Key | Default | Extra |
| cardId | bigint | NO | PRI | NULL | auto\_increment |
| cardCode | varchar(50) | NO |  | NULL |  |
| owner | varchar(50) | NO |  | NULL |  |
| cvvCode | varchar(50) | NO |  | NULL |  |
| dateExpired | varchar(4) | NO |  | NULL |  |

Bang:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Field | Type | Null | Key | Default | Extra |
| bikeId | bigint | NO | PRI | NULL | auto\_increment |
| parkingLotId | bigint | NO | MUL | NULL |  |
| type | int | NO |  | NULL |  |
| upfrontPrice | int | NO |  | NULL |  |
| rentPrice | int | NO |  | NULL |  |
| lastUsed | datetime | NO |  | NULL |  |
| maxTimeUsed | time | NO |  | NULL |  |
| isAvailable | tinyint(1) | NO |  | NULL |  |

Bang:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Field | Type | Null | Key | Default | Extra |
| rentTransactionId | bigint | NO | PRI | NULL | auto\_increment |
| cardId | bigint | NO | MUL | NULL |  |
| bikeId | bigint | NO | MUL | NULL |  |
| startTime | datetime | NO |  | NULL |  |

Bang:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Field | Type | Null | Key | Default | Extra |
| rentTransactionId | bigint | NO | PRI | NULL |  |
| rentTime | datetime | NO |  | NULL |  |
| finishTime | datetime | NO |  | NULL |  |
| rentPrice | int | NO |  | NULL |  |

## Non-Database Management System Files

*<Provide the detailed description of all non-DBMS files if any and include a narrative description of the usage of each file that identifies if the file is used for input, output, or both, and if the file is a temporary file. Also provide an indication of which modules read and write the file and include file structures (refer to the data dictionary). As appropriate, the file structure information should include the following:*

*• Record structures, record keys or indexes, and data elements referenced within the records*

*• Record length (fixed or maximum variable length) and blocking factors*

*• Access method (e.g., index sequential, virtual sequential, random access, etc.)*

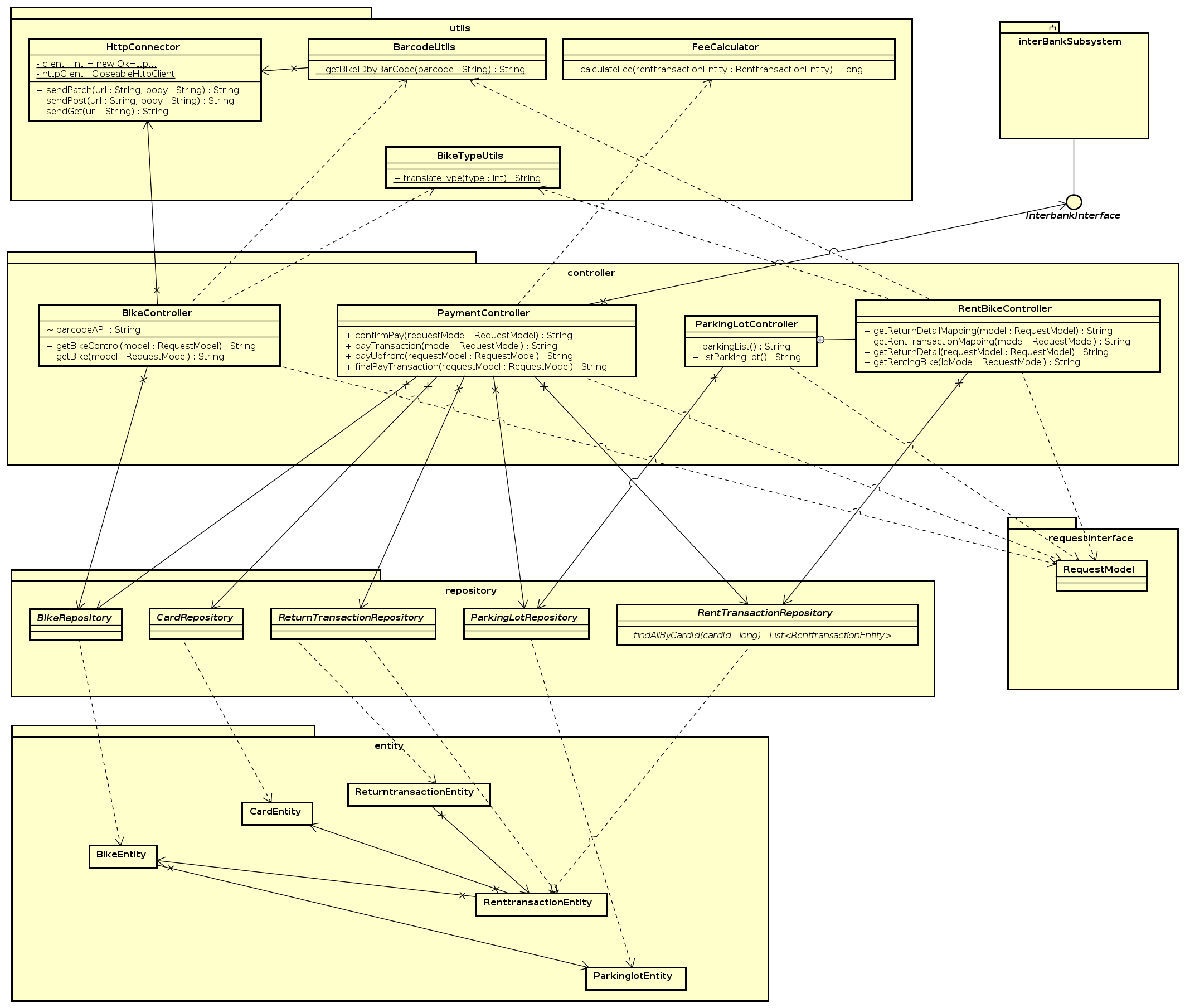
*• Estimate of the file size or volume of data within the file, including overhead resulting from file access methods*

*• Definition of the update frequency of the file (If the file is part of an online transaction-based system, provide the estimated number of transactions per unit of time, and the statistical mean, mode, and distribution of those transactions.)*

*• Backup and recovery specifications>*

## Class Design

### General Class Diagram



### Class Diagrams

#### Class Diagram for Package A

#### Class Diagram for Subsystem B

…

### Class Design

#### Class “SampleClass1”

<SampleClass1 class image in UML>

**Attribute**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| *#* | *Name* | *Data type* | *Default value* | *Description* |
| 1 |  |  |  |  |
| 2 |  |  |  |  |

**Operation**

|  |  |  |  |
| --- | --- | --- | --- |
| *#* | *Name* | *Return type* | *Description (purpose)* |
| 1 |  |  |  |
| 2 |  |  |  |

*Parameter*:

* x: Default value, description
* y: Default value, description

*Exception*:

* AException if …
* BException if …

**Method**

How to use parameters / attributes

Flowchart / Sequence diagram if the method has a complex/special algorithm

**State**

State diagram if any

#### Class “SampleClass2”

…

# Design Considerations

***<Describe issues which need to be addressed or resolved before attempting to devise a complete design solution>***

## Goals and Guidelines

*<Describe any goals, guidelines, principles, or priorities which dominate or embody the design of the system and its software.*

*Examples of such goals might be: an emphasis on speed versus memory use; or working, looking, or “feeling” like an existing product.*

*Guidelines include coding guidelines and conventions.*

*For each such goal or guideline, describe the reason for its desirability unless it is implicitly obvious.*

*Describe any design policies and/or tactics that do not have sweeping architectural implications (meaning they would not significantly affect the overall organization of the system and its high-level structures), but which nonetheless affect the details of the interface and/or implementation of various aspects of the system (e.g., choice of which specific product to use)*>

## Architectural Strategies

*<Describe any design decisions and/or strategies that affect the overall organization of the system and its higher-level structures. These strategies should provide insight into the key abstractions and mechanisms used in the system architecture. Describe the reasoning employed for each decision and/or strategy (possibly referring to previously stated design goals and principles) and how any design goals or priorities were balanced or traded-off.*

*Examples of design decisions might concern (but are not limited to) things like the following:*

*• Use of a particular type of product (programming language, database, library, commercial off-the-shelf (COTS) product, etc.)*

*• Reuse of existing software components to implement various parts/features of the system*

*• Future plans for extending or enhancing the software*

*• User interface paradigms (or system input and output models)*

*• Hardware and/or software interface paradigms*

*• Error detection and recovery*

*• Memory management policies*

*• External databases and/or data storage management and persistence*

*• Distributed data or control over a network*

*• Generalized approaches to control*

*• Concurrency and synchronization*

*• Communication mechanisms*

*• Management of other resources*

>

## Coupling and Cohesion

*<Evaluate your design and describe which levels of coupling and cohesion that your design is at. Give proofs for your assumptions. Explain if there is any special design or exceptions>*

## Design Principles

*<Does your design follow the SOLID principles for the new requirements/changing requirements? Give proofs for your assumptions. Explain if there is any special design or exceptions>*

## Design Patterns

*<Do you use any design patterns for your design? If yes, describe detailly why you use those design patterns? Describe in detail on the solutions and how to implement each design pattern>*