HANOI UNIVERSITY OF SCIENCE AND TECHNOLOGY

School of Information and communications technology

Software Design Document

Version 1.3

ECOBIKE RENTAL

Subject: Software Design and Construction

Group 13

List of participants

Hoàng Quốc Trung 20173419

Lại Đắc Kiên 20173212

Lê Thị Huế 20173149

Phan Duy Bảo 20172967

*Hanoi, 01/2022*

*<All notations inside the angle bracket are not part of this document, for its purpose is for extra instruction. When using this document, please erase all these notations and/or replace them with corresponding content as instructed>*

*<This document, written by Prof. NGUYEN Thi Thu Trang, is used as a case study for student with related courses. Any modifications and/or utilization without the consent of the author is strictly forbidden>*

**Table of Contents**

Table of Contents 1

1 Introduction 5

1.1 Objective 5

1.2 Scope 5

1.3 Glossary 5

1.4 References 5

2 Overall Description 7

2.1 General Overview 7

2.2 Assumptions/Constraints/Risks 7

2.2.1 Assumptions 7

2.2.2 Constraints 7

2.2.3 Risks 8

3 System Architecture and Architecture Design 9

3.1 Architectural Patterns 9

3.2 Interaction Diagrams 9

3.3 Analysis Class Diagrams 9

3.4 Unified Analysis Class Diagram 9

3.5 Security Software Architecture 9

4 Detailed Design 10

4.1 User Interface Design 10

4.1.1 Screen Configuration Standardization 10

4.1.2 Screen Transition Diagrams 10

4.1.3 Screen Specifications 10

4.2 Data Modeling 10

4.2.1 Conceptual Data Modeling 10

4.2.2 Database Design 10

4.3 Non-Database Management System Files 11

4.4 Class Design 11

4.4.1 General Class Diagram 11

4.4.2 Class Diagrams 11

4.4.3 Class Design 11

5 Design Considerations 13

5.1 Goals and Guidelines 13

5.2 Architectural Strategies 13

5.3 Coupling and Cohesion 14

5.4 Design Principles 14

5.5 Design Patterns 14

# Introduction

## Objective

Tài liệu này nhằm mục đích xây dựng và thiết kế hệ thống cho thuê xe EcoBike Rental, dựa trên tài liệu đặc tả SRS

Tài liệu này dành cho nhà phát triển phần mềm

## Scope

Hệ thống cho thuê xe : Ecobike Rental

Các chức năng chính của hệ thống

* Xem danh sách bãi đỗ xe
* Xem chi tiết bãi đỗ xe
* Xem chi tiết xe
* Xem danh sách xe đang thuê
* Thuê xe
* Trả xe
* Tìm kiếm xe/ bãi đỗ xe

Mục đích chính của hệ thống là giúp người dùng đơn giản hóa công việc thuê xe và trả xe, thay thế cách làm thông thường là ghi sổ

## References

| [1] | Tài liệu đặc tả yêu cầu phần mềm SRS |
| --- | --- |
| [2] | Tài liệu mẫu SDD-Template-v1.3 - cô Nguyễn Thị Thu Trang |

# Overall Description

## General Overview

Khu đô thị Ecopark có dịch vụ cho thuê xe đạp theo giờ với nhiều bãi để xe để thuê/trả xe tự động trong khu đô thị. Mỗi khi trả xe, khách đưa xe vào ổ khoá tại mỗi vị trí để xe

Trong hệ thống này, sẽ cung cấp các phương thức để truy cập vào hệ thống và thực hiện các chức năng thuê xe, trả xe thay vì làm các hành động như thực tế

## Assumptions/Constraints/Risks

### Assumptions

Hệ thống dùng trên nền tảng Web, kết nối TCP/IP với người dùng

Người dùng cần có thiết bị máy tính có thể kết nối mạng wifi

Khách hàng cần phải tạo tài khoản trên ứng dụng EcoBike Rental, xác thực thông tin, thiết lập quyền truy cập của ứng dụng, và thiết lập phương thức thanh toán để trả phí thuê xe (bằng cách liên kết với liên ngân hàng hoặc ví điện tử).

EcoBike Rental là một hệ thống đa nền tảng hoạt động 24/7, cho phép người dùng mới có thể làm quen dễ dàng.

Hệ thống này có thể phục vụ 100 người dùng cùng lúc mà hiệu suất không thay đổi đáng kể, đồng thời có thể hoạt động 200 giờ liên tục không lỗi. Ngoài ra, hệ thống có thể hoạt động trở lại bình thường trong vòng 2 giờ sau khi xảy ra lỗi.

Thời gian đáp ứng tối đa của hệ thống (nếu 2 không được nêu rõ) là 1 giây lúc bình thường hoặc 2 giây lúc cao điểm

### Risks

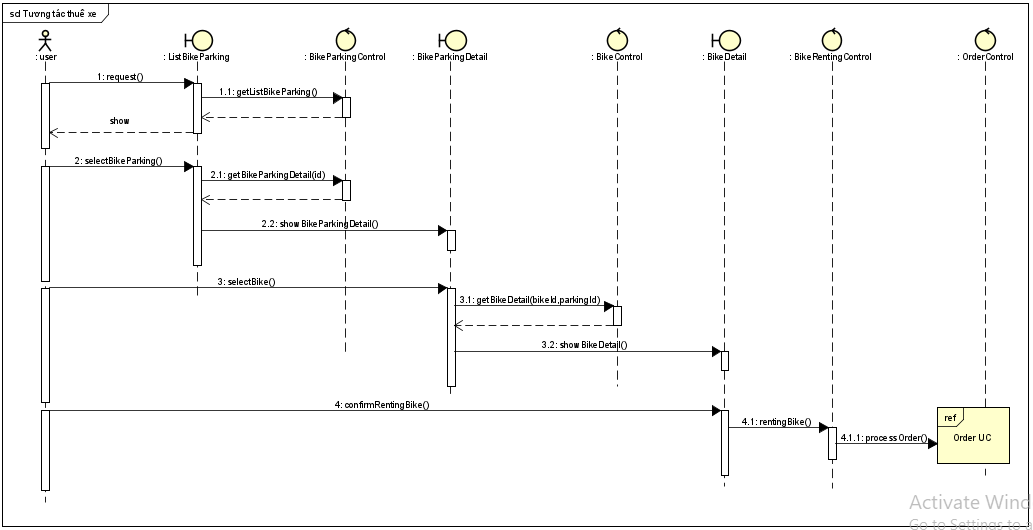
Mọi thao tác trên phần mềm nếu bị ngắt kết nối Internet sẽ bị mất hết dữ liệu

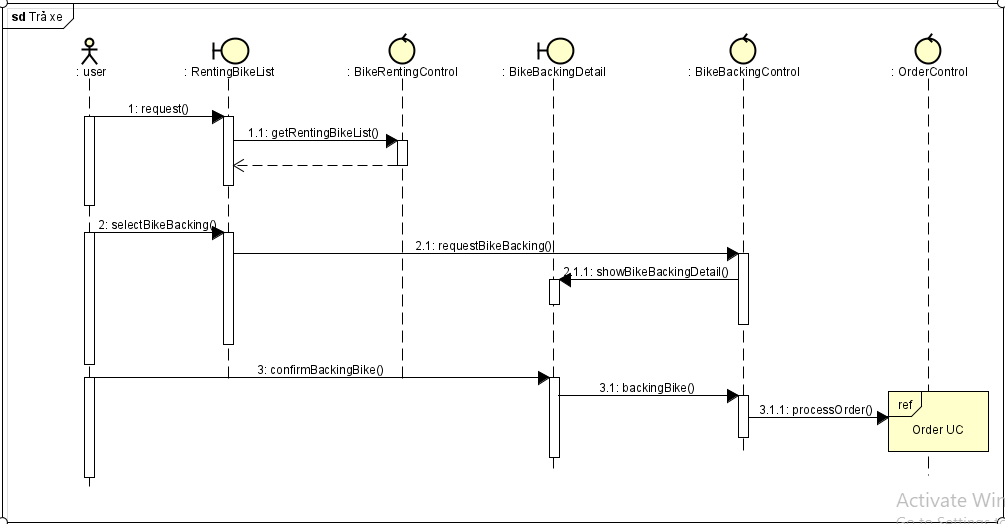
# System Architecture and Architecture Design

## Architectural Patterns

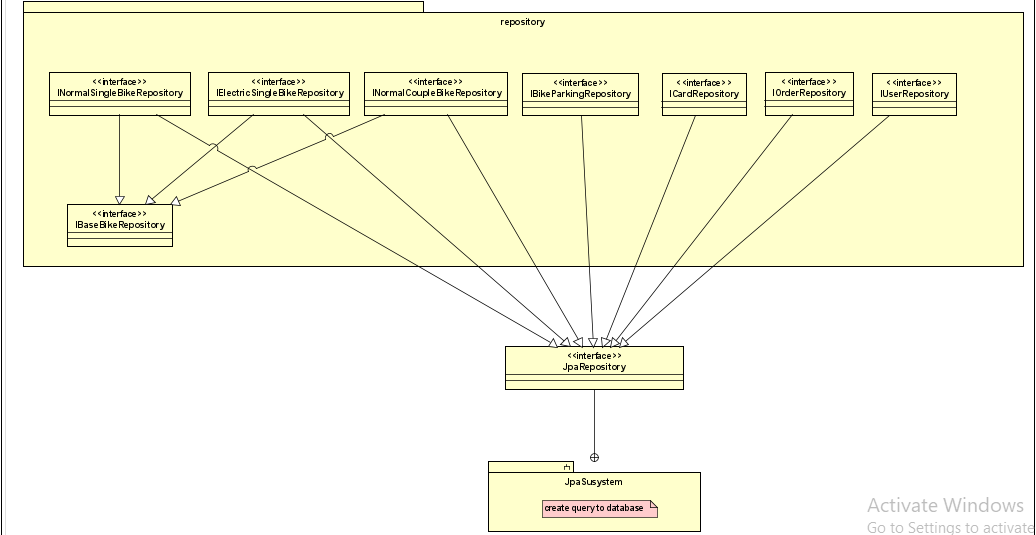
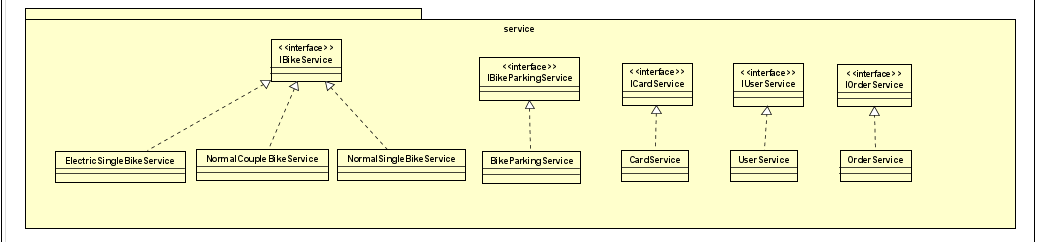
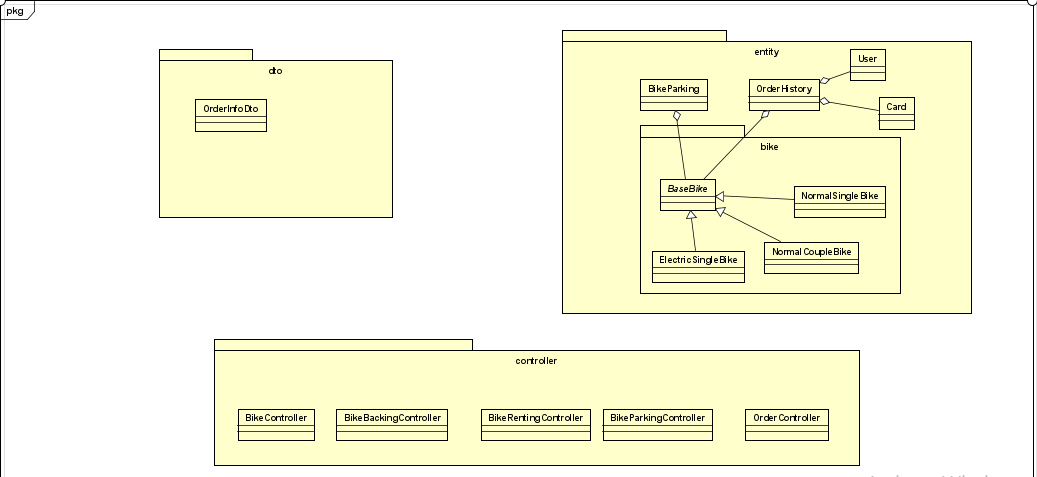
Thiết kế hệ thống theo mô hình MVC đây là cách thiết kế phổ biến cho các hệ thống web

## Interaction Diagrams





## Unified Analysis Class Diagram



# Detailed Design

## User Interface Design

### Screen Configuration Standardization

***Display***

Số lượng màu được hỗ trợ: 16,777,216 màu

Độ phân giải: 1366 x 768 pixels

Màu nền : #FFFF

***Screen***

Vị trí của của button: Ở dưới cùng (theo chiều dọc) và ở giữa (theo chiều ngang) của khung.

Vị trí của message: Ở giữa trung tâm khung màn hình

Vị trí của screen title: phía trên chính giữa phần mục

***Control***

Xử lý check input: Nên kiểm tra xem input có empty hay không. Màu xanh #FF68 , chữ màu trắng

Dịch chuyển màn hình: Không có các khung chồng lên nhau. Các màn hình được tách biệt.

Thứ tự các màn hình trong hệ thống:

1. Xem danh sách bãi xe

2. Xem chi tiết bãi xe

3. Xe chi tiết xe

4. Thuê xe

5. Thanh toán thuê xe

6. Xem danh sách xe đang thuê

7. Trả xe

8. Thanh toán trả xe

9. Kết quả

***Nhập input từ bàn phím***

Sẽ không có phím tắt. Người dùng bấm vào button back trên trình duyệt

***Error***

Một thông điệp sẽ được hiện lên để thông báo cho người dùng biết vấn đề đang gặp phải là gì.

***Định dạng chữ***

* + Số căn phải
  + Chữ căn trái
  + Font: Arial medium size 14px, màu đen #0000
  + Nền trắng #FFF

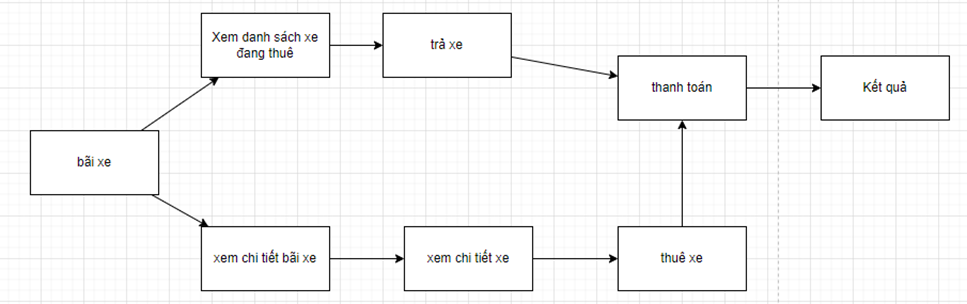
***Định dạng Button***

* + Màu #0a53be
  + Chữ màu trắng

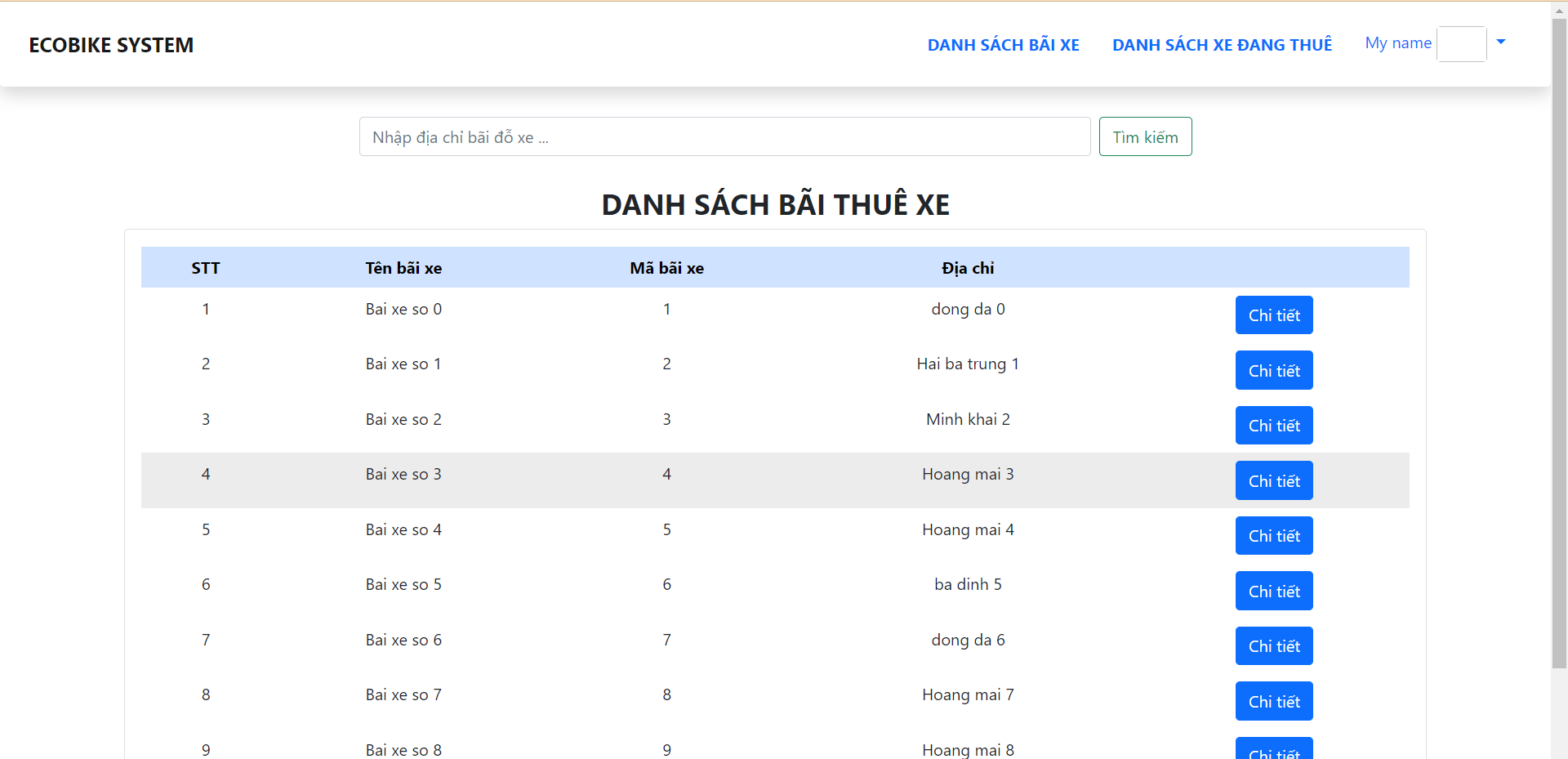
***Định dạng logo***

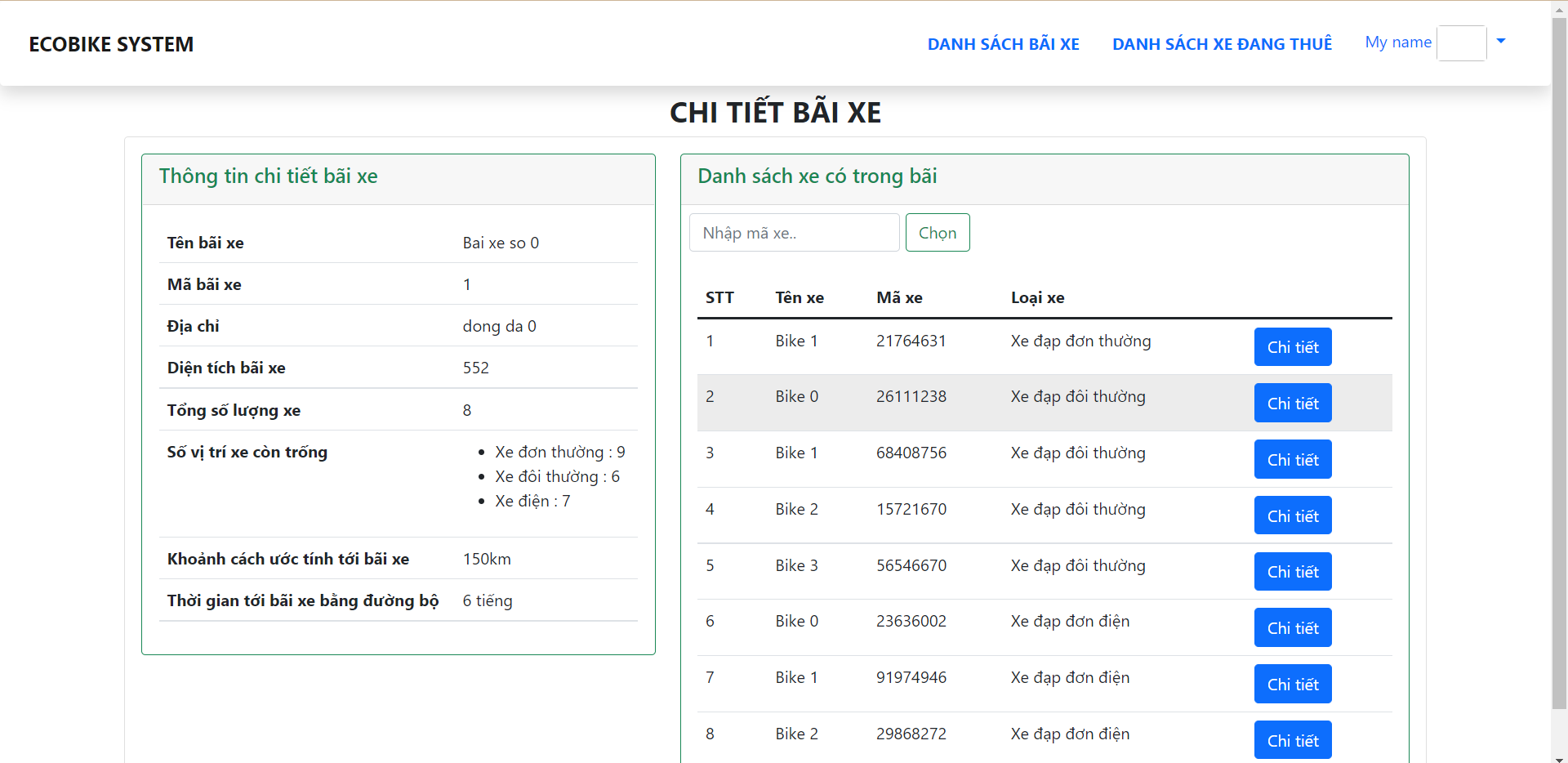
* + Chữ viết hoa in đậm : “ECOBIKE RENTAL”

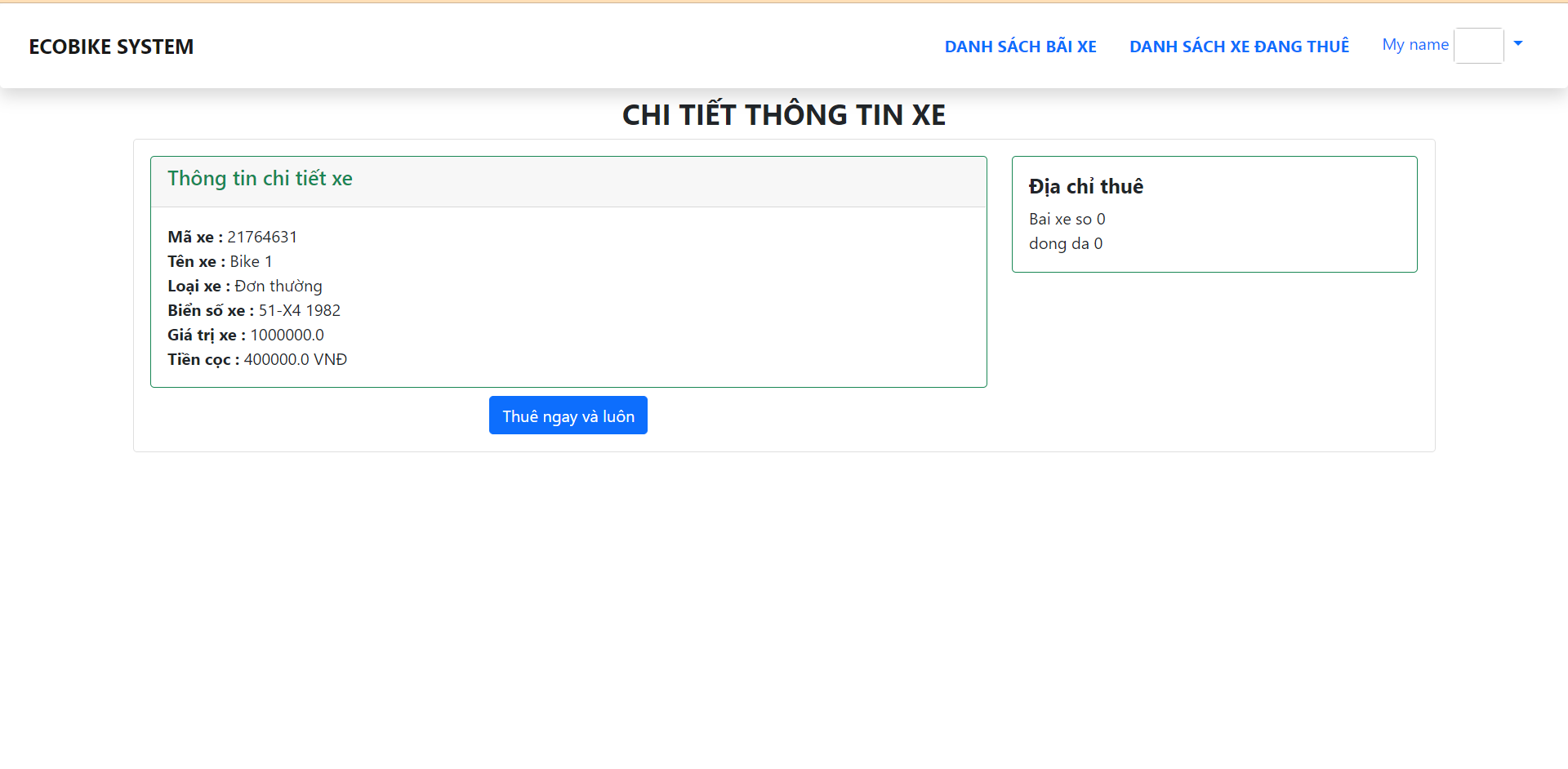
### Screen Transition Diagrams

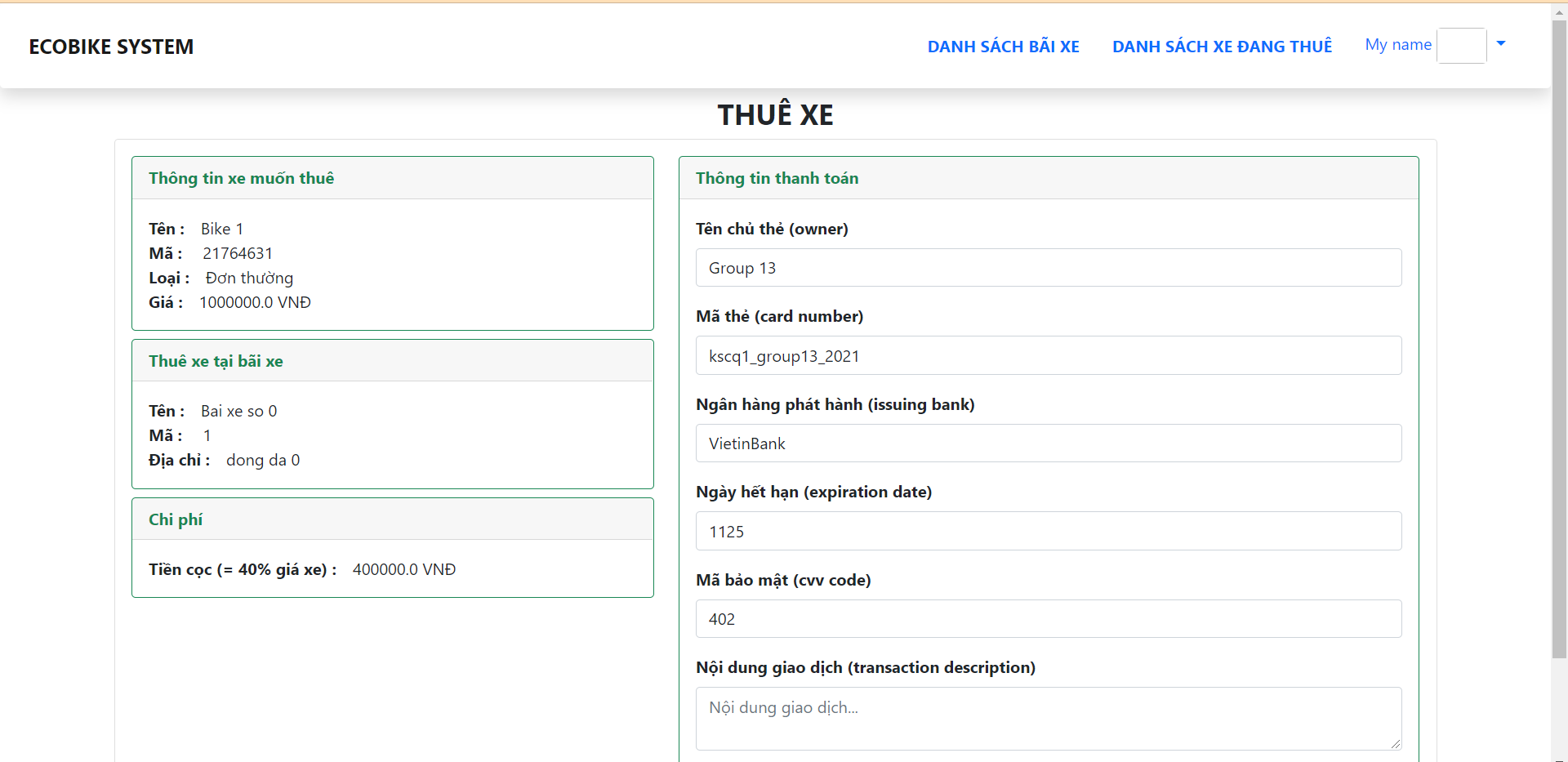


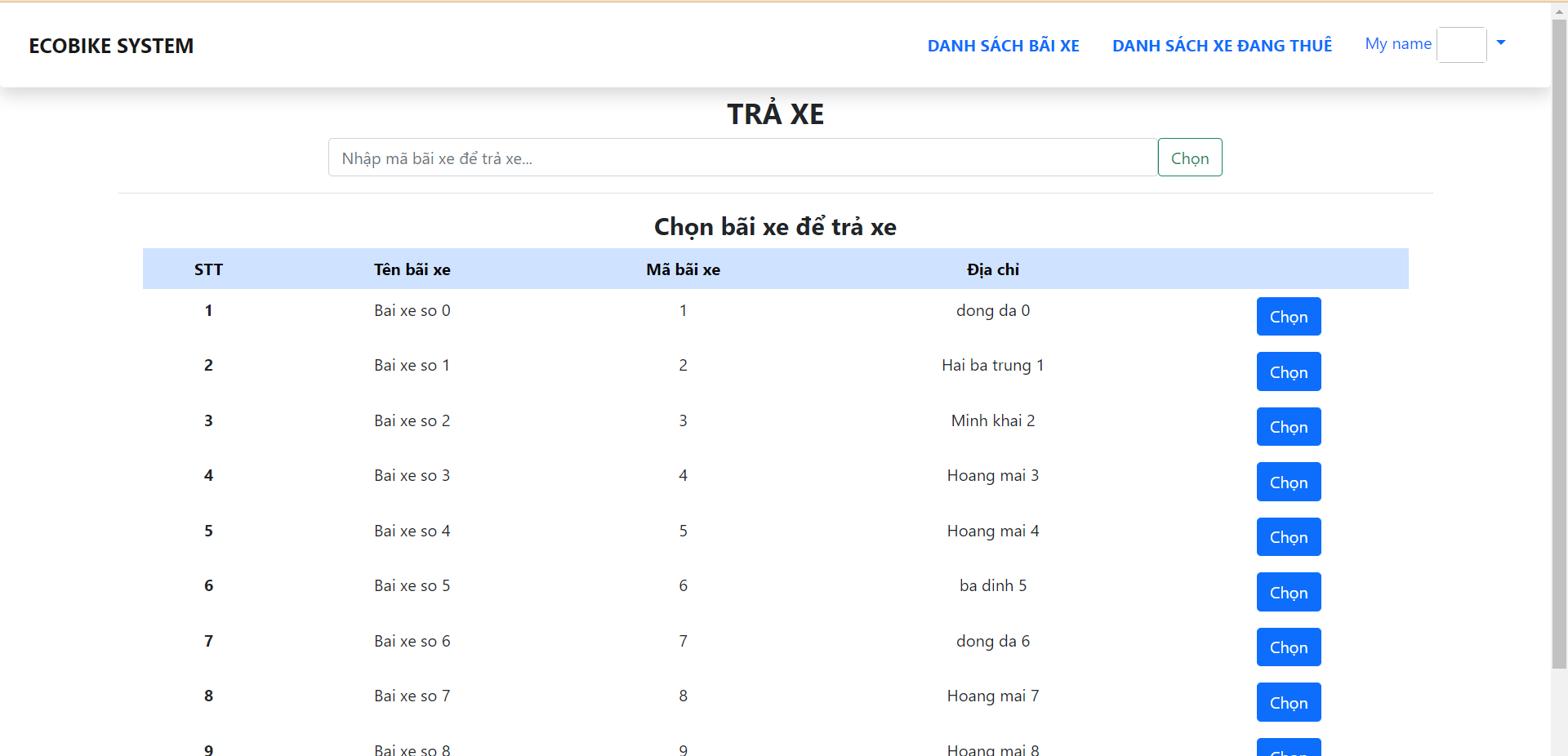
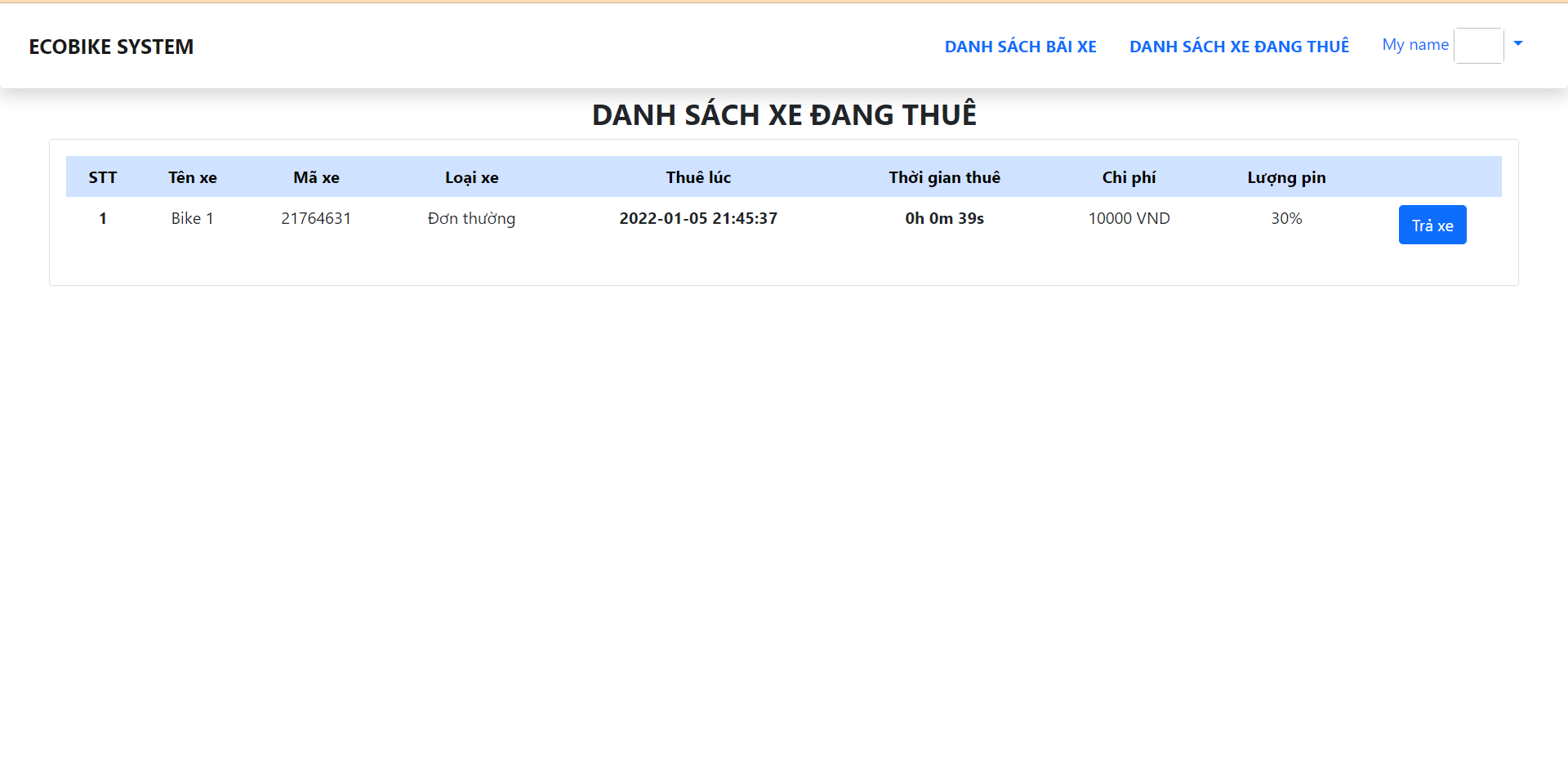
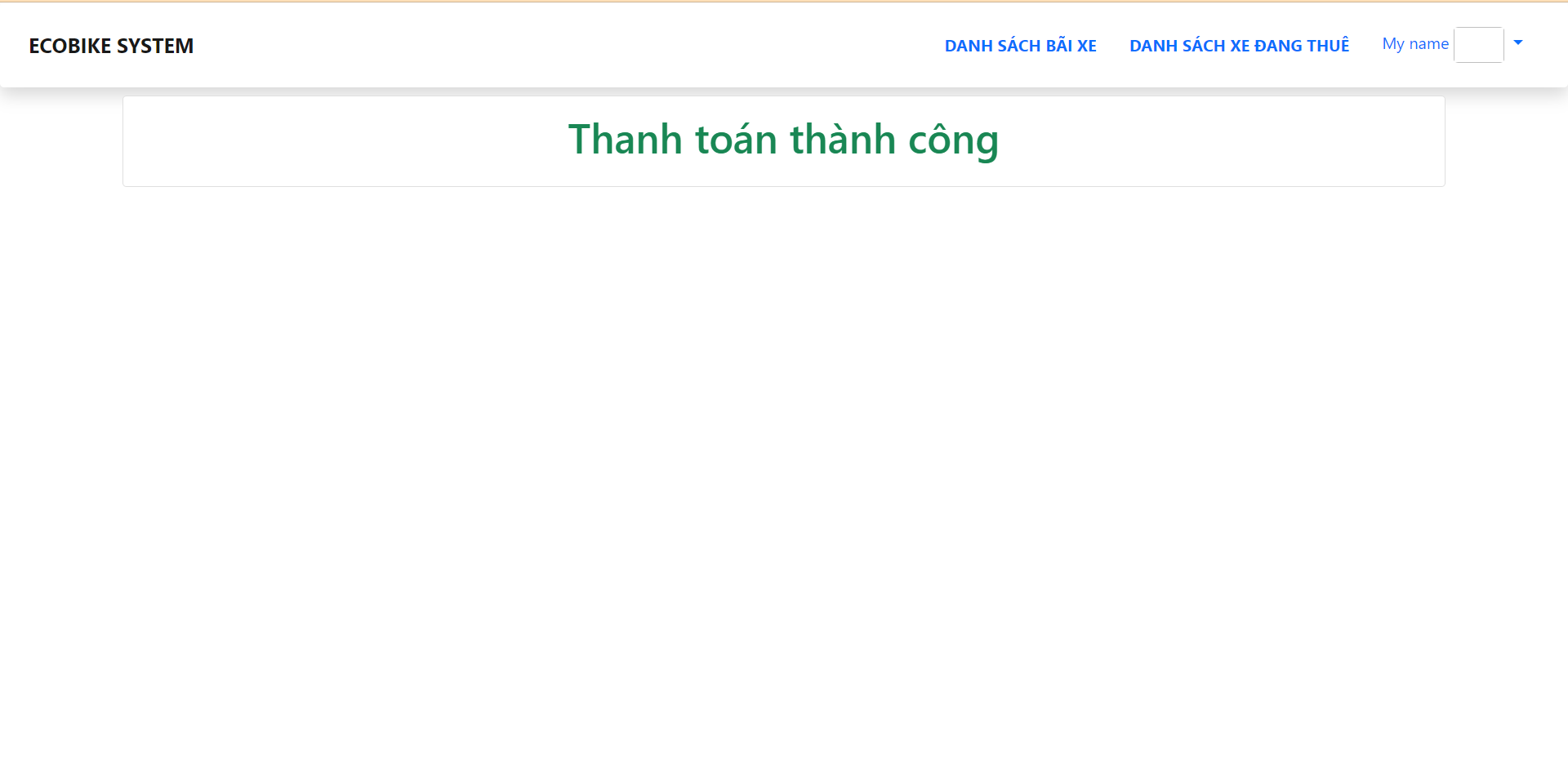
### Screen Specifications

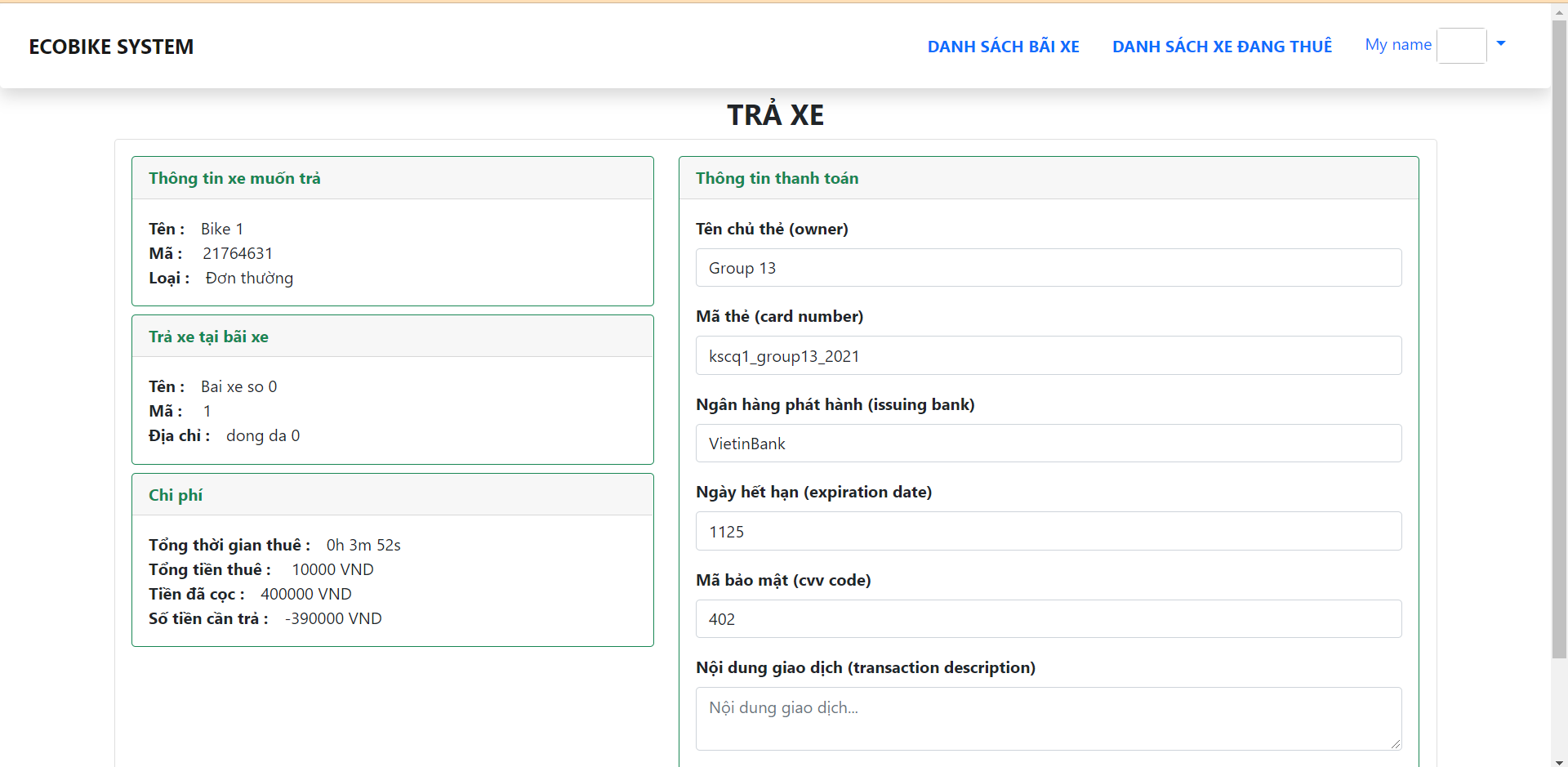
**

**

**

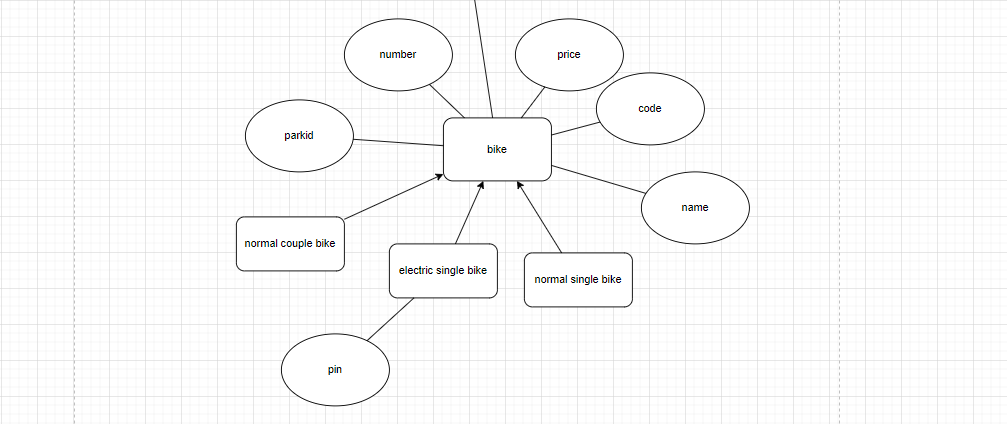
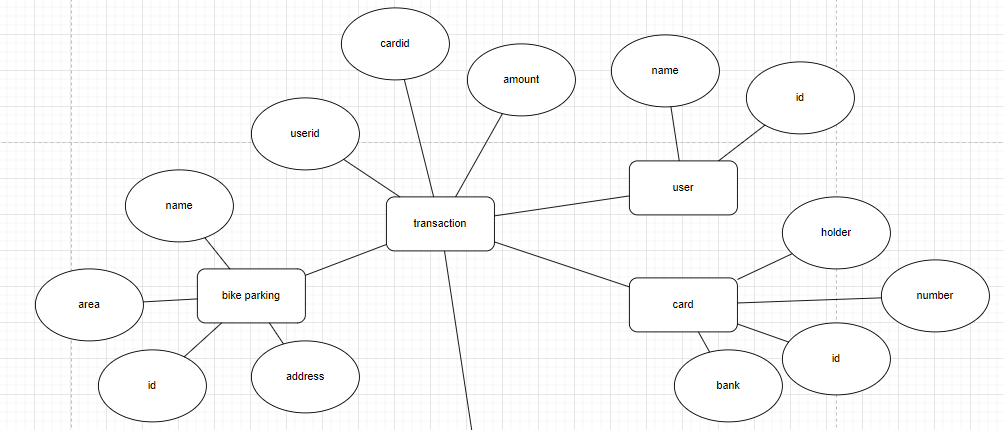
**

**

**

## Data Modeling

### Conceptual Data Modeling

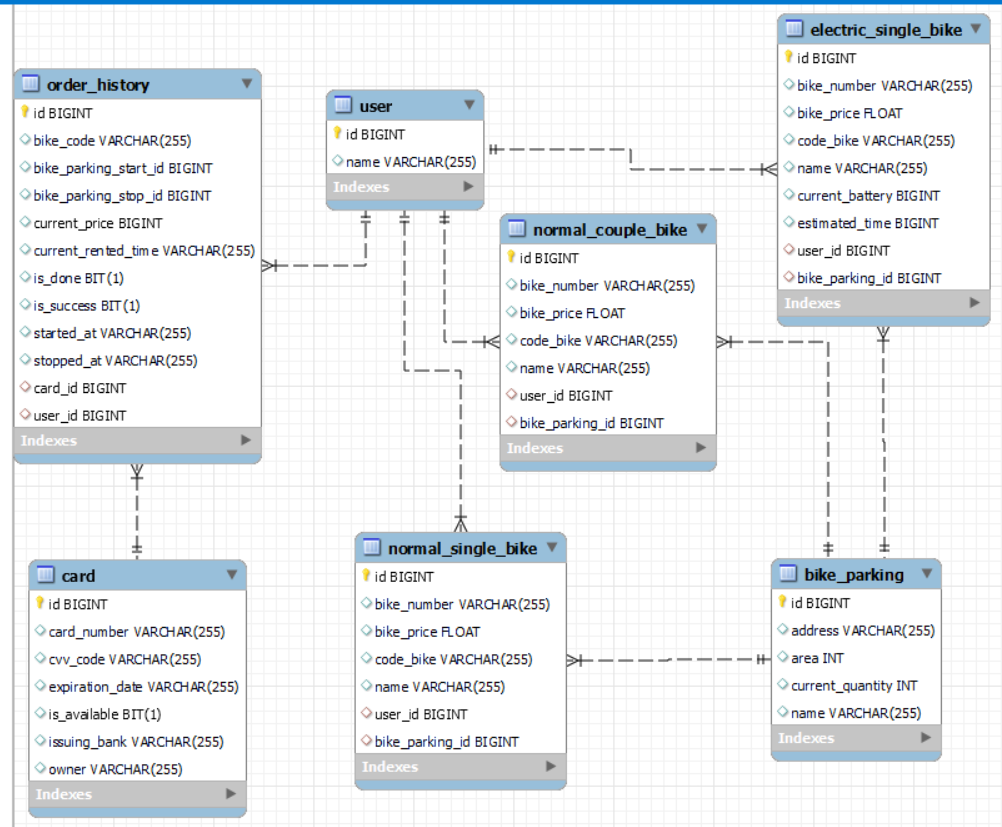
**

### Database Design

#### Database Management System

Hệ quản trị CSDL MySQL. là một hệ quản trị cơ sở dữ liệu quan hệ

#### Database Diagram



#### Database Detail Design

**Table 1. Example of table bike parking**

| *#* | *PK* | *FK* | *Column* | *Data type* | *Default value* | *Mandatory* | *Description* |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | x |  | id | BIGINT | auto generic | yes |  |
| 2 |  |  | address | VARCHAR(255) |  | yes | địa chỉ bãi xe |
| 3 |  |  | area | INT |  | yes | diện tích bãi xe |
| 4 |  |  | current\_quantity | INT |  | yes | số vị trí còn trống |
| 5 |  |  | name | VARCHAR(255) |  | yes | tên bãi xe |

**Table 1. Example of table order history**

| *#* | *PK* | *FK* | *Column* | *Data type* | *Default value* | *Mandatory* | *Description* |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | x |  | id | BIGINT | auto generic | yes |  |
| 2 |  |  | bike\_parking\_start\_id | VARCHAR(255) |  | yes | địa chỉ bãi xe |
| 3 |  |  | bike\_parking\_stop\_id | INT |  | yes | diện tích bãi xe |
| 4 |  |  | current\_price | INT |  | yes | số vị trí còn trống |
| 5 |  |  | current\_rented\_time | VARCHAR(255) |  | yes | tên bãi xe |
| 6 |  |  | cardid | INT |  | yes | id người thuê |
| 7 |  |  | userid | INT |  | yes | id thẻ thanh toán |

**Table 1. Example of table card**

| *#* | *PK* | *FK* | *Column* | *Data type* | *Default value* | *Mandatory* | *Description* |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | x |  | id | BIGINT | auto generic | yes |  |
| 2 |  |  | number | VARCHAR(255) |  | yes | địa chỉ bãi xe |
| 3 |  |  | expiration\_data | INT |  | yes | diện tích bãi xe |
| 4 |  |  | is\_available | BIT |  | yes | số vị trí còn trống |
| 5 |  |  | bank | VARCHAR(255) |  | yes | tên bãi xe |
| 6 |  |  | owner | VARCHAR(255) |  | yes | chủ sở hữu |

**Table 1. Example of table user**

| *#* | *PK* | *FK* | *Column* | *Data type* | *Default value* | *Mandatory* | *Description* |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | x |  | id | BIGINT | auto generic | yes |  |
| 2 |  |  | name | VARCHAR(255) |  | yes | tên khách hàng |

**Table 1. Example of table normal single bike**

| *#* | *PK* | *FK* | *Column* | *Data type* | *Default value* | *Mandatory* | *Description* |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | x |  | id | BIGINT | auto generic | yes |  |
| 2 |  |  | bike\_number | VARCHAR(255) |  | yes | số xe |
| 3 |  |  | bike\_price | INT |  | yes | giá xe |
| 4 |  |  | code\_price | VARCHAR(255) |  | yes | mã xe |
| 5 |  |  | name | VARCHAR(255) |  | yes | tên xe |
| 6 |  |  | userid | INT |  |  | tên người thuê xe |
| 7 |  |  | bike\_parking\_id | INT |  | yes | tên bãi đỗ xe |

**Table 1. Example of table normal couple bike**

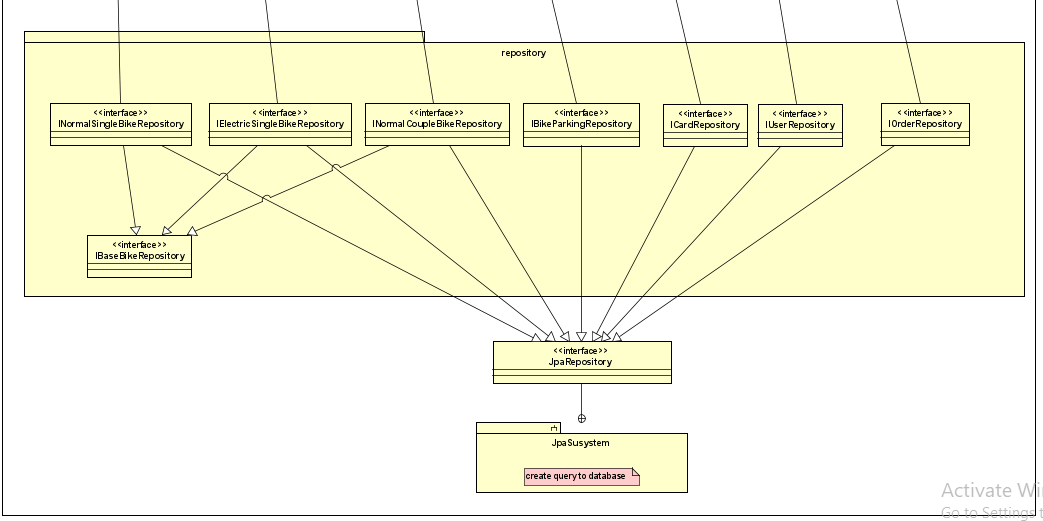
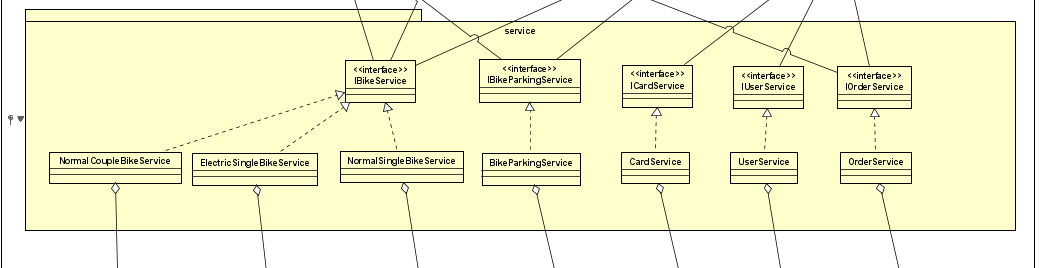
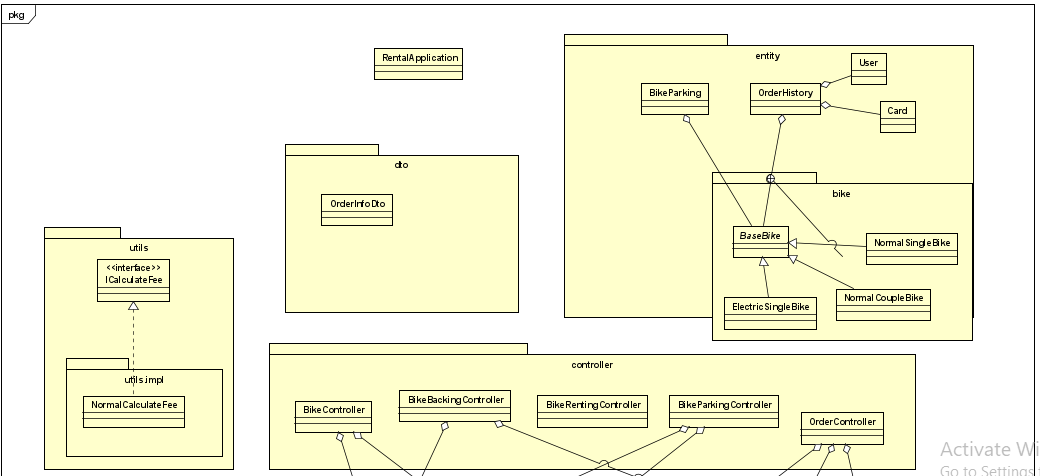
| *#* | *PK* | *FK* | *Column* | *Data type* | *Default value* | *Mandatory* | *Description* |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | x |  | id | BIGINT | auto generic | yes |  |
| 2 |  |  | bike\_number | VARCHAR(255) |  | yes | số xe |
| 3 |  |  | bike\_price | INT |  | yes | giá xe |
| 4 |  |  | code\_price | VARCHAR(255) |  | yes | mã xe |
| 5 |  |  | name | VARCHAR(255) |  | yes | tên xe |
| 6 |  |  | userid | INT |  |  | tên người thuê xe |
| 7 |  |  | bike\_parking\_id | INT |  | yes | tên bãi đỗ xe |

**Table 1. Example of table electric single bike**

| *#* | *PK* | *FK* | *Column* | *Data type* | *Default value* | *Mandatory* | *Description* |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | x |  | id | BIGINT | auto generic | yes |  |
| 2 |  |  | bike\_number | VARCHAR(255) |  | yes | số xe |
| 3 |  |  | bike\_price | INT |  | yes | giá xe |
| 4 |  |  | code\_price | VARCHAR(255) |  | yes | mã xe |
| 5 |  |  | name | VARCHAR(255) |  | yes | tên xe |
| 6 |  |  | userid | INT |  |  | tên người thuê xe |
| 7 |  |  | bike\_parking\_id | INT |  | yes | tên bãi đỗ xe |
| 8 |  |  | current\_batery | BIGINT |  | yes | lượng pin hiện tại |
| 9 |  |  | estimate-time | BIGINT |  | yes | thời gian ước tính |

## Class Design

### General Class Diagram



…

### Class Design

#### Class “SampleClass1”

<SampleClass1 class image in UML>

**Table 1. Example of attribute design**

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

**Table 1. Example of operation design**

| *#* | *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. Remember that, you have to refactor your source code to strictly follow the final design>***

## 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>*

*<You may show the previous design from which you made improvements to get better levels of coupling and cohesion. You should clarify how and why you did these improvements>*

## Design Principles

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

*<You may show the previous design from which you made improvements to get a better design, which follows SOLID principles in spite of additional requirements. You should clarify how and why you did these improvements>*

## 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>*