HANOI UNIVERSITY OF SCIENCE AND TECHNOLOGY

School of Information and communications technology

Software Requirement Specification

Version 1.2

**EcoBike Rental**

Subject: ITSS Software Development

Group 6

Vũ Hồng Hải - 20205150

Bùi Duy Hiển - 20200208

Nguyễn Tuấn Hiệp - 20205151

Hà Huy Hoàng - 20205152

*Hanoi,* *August 2023*

Table of contents

Table of contents 1

1 Introduction 2

1.1 Objective 2

1.2 Scope 2

1.3 Glossary 2

1.4 References 3

2 Overall Description 4

2.1 Survey 4

2.2 Overall requirements 4

2.3 Business process 4

3 Detailed Requirements 5

3.1 Use case 1 6

3.2 Use case 2 7

4 Supplementary specification 8

4.1 Functionality 8

4.2 Usability 8

4.3 Reliability 8

4.4 Performance 8

4.5 Supportability 8

4.6 Other requirements 8

# Introduction

## Objective

The document gives a detailed description of the functionality related to renting and returning bikes. Document describing the purpose and features of the system, the interfaces and constraints the system needs to implement in response to external stimuli.

Documentation for stakeholders and software developers.

## Scope

Our application will provide managers with automatic bike rental and return services in dock lots. But in this problem, we will skip the login part and focus on the function related to renting and returning the bike. The goal is to create a fully automated management application that provides customers with the most basic functionalit A simple way the user after accessing, will see a list of dock lots. After that, users can view information about dock lots and perform bike rental functions through bike codes. After the initial payment process, customers can use the registered bike. Before or during use, customers can also view information about their rented bike such as bike status, battery status. At the end of the usage period, the user will be able to perform the function of returning the bike and making the final payment.

## Glossary

| ***No*** | ***Term*** | ***Explanation*** | ***Example*** | ***Note*** |
| --- | --- | --- | --- | --- |
| 1 | token | A piece of data created by server, and contains the user's information, as well as a special token code that user can pass to the server with every method that supports authentication, instead of passing a username and password directly. | JSON Web Token (JWT) | Compact, URL-safe and usable especially in web browser single sign-on (SSO) context. |
| 2 | Use case | Show the interaction between the user and the system, thereby expressing the user's requirements | Return Bike | Use case performs the bike return function |
| 3 | Actor | Only users or external objects interact with the system | Customer | As a system user |
| 4 | Use case diagrams | Demonstrate the different ways that a user might interact with a system. |  |  |
| 5 | Activity Diagram | Activity diagram is basically a flowchart to represent the flow from one activity to another activity. The activity can be described as an operation of the system |  |  |
|  |  |  |  |  |

## References

# Overall Description

## Survey

The software has 2 actors: Customer, Interbank. Customer will be the main user of the system, interacting with basic functions such as renting and returning bikes. The Interbank actor will represent the interaction between the system and the bank to perform payment-related work.

## Overall requirements

Customers when entering the system can see a list of dock lots, to decide to choose which dock lot is appropriate. When booking a bike, customers will choose a bike, view bike information, and finally make a payment (like a deposit when renting a bike). In return bike, the system will simply receive the request, simplify the calculation, and send the goods budget to complete the payment to the customer.

A diagram of a company

Description automatically generated

## Business process

*<Sequence of use cases, e.g. Use case 1 then use case 2, if a condition matched, do the use case 3… You should visualize the process with activity diagrams>*

# Detailed Requirements

## Use case 1

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Use Case “View docks”**   1. **Use case code**   UC001   1. **Brief Description**   This use case describes the interaction between Customer and EcoBike Rental System when Customer wishes to view docks information.   1. **Actors**    1. **Customer** 2. **Preconditions** 3. **Basic Flow of Events** 4. The software displays home screen. 5. Customer press view get started button. 6. The software displays a list of docks (see Table B). 7. **Alternative flows**   - None   1. **Input data**  * **None**  1. **Output data**   **Table B-Output data list of dock**   |  |  |  |  |  | | --- | --- | --- | --- | --- | | **No** | **Data fields** | **Description** | **Display format** | **Example** | | **1** | Dock ID | ID of dock | String | 1 | | **2** | Dock name | Dock name | String | DCV | | **3** | Address | Address of dock | String | 1 Dai Co Viet | | **4** | Image | Image of dock | Image |  |  1. **Postconditions**  * None |

**Use Case “Search Dock”**

**1.Use case code**

UC002

**2.Brief Description**

This use case describes the interaction between Customer and EcoBike when Customer wishes to search dock

**3.Actors**

**-** Customer

**4.Preconditions**

**5.Basic Flow of Events**

1. The customer input name of dock in search bar
2. The software displays matched dock

**6.Alternative flows**

- None

**7.Input data**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Data fields** | **Description** | **Display format** | **Example** |
|  | Dock name | Name of dock | String | DCV |

**8. Output data**

Table B-Output data of dock information

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Data fields** | **Description** | **Display format** | **Example** |
| **1** | Dock ID | ID of dock | String | 1 |
| **2** | Dock name | Dock name | String | DCV |
| **3** | Address | Address of dock | String | 1 Dai Co Viet |
| **4** | Image | Image of dock | Image |  |

## Use case 3

**Use Case “View bike list in dock”**

**1.Use case code**

UC003

**2.Brief Description**

This use case describes the interaction between Customer and EcoBike when Customer wishes to view bike information

**3.Actors**

**-** Customer

**4.Preconditions**

**5.Basic Flow of Events**

1. The customer chooses a dock to display more information.
2. The software displays list of bikes in that dock (see Table B).

**6.Alternative flows**

- None

**7.Input data**

**-** None

**8. Output data**

Table B-Output data of bike information

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Data fields** | **Description** | **Display format** | **Example** |
|  | Bike ID | ID of bike | String | 1 |
| 2. | Bike name | Name of the bike | String | Bike 1 |
| 3. | Image | Image of the bike | Image |  |
| 4. | Category | Type of the bike | String | Ebike |
| 5. | Dock name | Name of the Dock | String | Eco1 |
| 6. | Barcode | Barcode of this bike | String | aac26076-7c60-40a5-a2eb-dfd808dc2574 |
| 7. | Battery | Current status of the battery; only for the ebike type | integer | 80% |

**9.Postconditions**

## Use case 5

**Use Case “Return Bike”**

1. **Use case code**

UC004

1. **Brief Description**

This use case describes the interaction between Customer and EcoBike when Customer wishes to view dock information.

1. **Actors**

* Customer

1. **Preconditions**

Customer must log in to the system and rent bike.

1. **Basic Flow of Events**

1. The software displays a list of nearby docks.

2. The user selects a dock and pushes the bike into an empty docking point of a dock (usually the nearest dock from the user's current location) and closes the lock.

3. The system automatically returns the deposit and deducts the amount of money corresponding to the rental period.

**Alternative flows**

Table N-Alternative flows of events for UC Place order

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Location** | **Condition** | **Action** | **Resume location** |
| 1 | At Step S | If … | * Action 1 | Resumes at Step Q |
| 2 | At Step O | If … | * Action 2 | Use case ends |

1. **Input data**

Table A-Input data of …

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **No** | **Data fields** | **Description** | **Mandatory** | **Valid condition** | **Example** |
|  |  |  |  |  |  |

1. **Output data**

Table B-Output data of …

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Data fields** | **Description** | **Display format** | **Example** |
|  |  |  |  |  |

1. **Postconditions**

## Manage active transaction

A diagram of a customer

Description automatically generated

**Use Case “View active transaction.”**

**1.Use case code**

UC003

**2.Brief Description**

This use case describes the interaction between Customer and EcoBike Rental System when Customer wishes to view active transaction.

**3. Actors**

**- Customer**

**4. Preconditions**

Customer must log in to the system.

**5. Basic Flow of Events**

1. Customer click the “Get started” button to view dock list
2. System display a list of available docks.
3. Customer (searches and) selects a dock to view the bike list.
4. System displays a list of bikes of the corresponding dock.
5. Customer enters the barcode of a bike and clicks the “Deposit” button.
6. System switch to deposit screen and show information of the bike.
7. Customer chooses the type of the transaction and enters credit card information (Table A).
8. System displays active transaction information (Table B).

**6. Alternative flows**

Table N-Alternative flows of events for UC Place order

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Location** | **Condition** | **Action** | **Resume location** |
| 1 | At step 5 | If customer already have an active transaction | * System displays a notification “Customer already rent a bike” | Resumes at Step 5 |
| 2 | At Step 7 | If credit card input information is invalid or the money in the card is not enough | * System notifies the customer with the corresponding notification | Resumes at Step 7 |

**7.Input data**

Table A-Input data

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **No** | **Data fields** | **Description** | **Mandatory** | **Data type** | **Example** |
| 1 | Bike barcode | Barcode of a bike | YES | UUID | 123e4567-e89b-12d3-a456-426614174000 |
| 2 | Transaction Type | Transaction Type | YES | STRING | NORMAL/ 24 HOURS |
| 3 | Card Number | ID of the credit card (16 digits) | YES | NUMBER | 55266263262626 |
| 4 | Cardholder Name | Name of cardholder | YES | STRING | John Doe |
| 5 | Issue Bank | Name of the Interbank | YES | STRING | Bank A |
| 6 | CVV Number | Card Security Code (3 digits) | YES | Number | 123 |
| 7 | Month | Expired month | YES | Number | 2 |
| 8 | Year | Expired year (2 digits) | YES | Number | 26 |

**8.Output data**

Table B-Output data of rental bike

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Data fields** | **Description** | **Display format** | **Example** |
| 1 | Transaction ID | ID of the transaction | Number | 9 |
| 2 | Bike image | Image of the bike | Image |  |
| 3 | Customer ID | ID of the customer | Number | 1 |
| 4 | Customer Name | Name of the customer | String | John Doe |
| 5 | Transaction Type | Type of the transaction | String | Normal |
| 6 | Status | Status of the transaction | String | Active |
| 7 | Deposit | The price to deposit (VND) | Number | 220000 |
| 8 | Last Pause | The exact time of last pause | String | 8/22/2023, 4:52:51 PM |
| 9 | Current Pay | The current price (VND) | Number | 447000 |
| 10 | Time Rent | The total time rent | String (hh:mm) | 24h25m |

**9.Postconditions**

**- None**

**Use Case “Pause active transaction.”**

**1.Use case code**

UC004

**2.Brief Description**

This use case describes the interaction between Customer and EcoBike Rental System when Customer wishes to pause his active transaction.

**3. Actors**

**- Customer**

**4. Preconditions**

Customer must have an active transaction and the status of the transaction is active.

**5. Basic Flow of Events**

1. Customer click the “Active Transaction” button on the navigation bar
2. System displays active transaction information (Table B).
3. Customer clicks the “Pause” button to pause the active transaction.
4. System changes the status of transaction from “active” to “paused” and update the last pause time.

**6. Alternative flows**

**- None**

**7.Input data**

**- None**

**8.Output data**

Table B-Output data of rental bike

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Data fields** | **Description** | **Display format** | **Example** |
| 1 | Transaction ID | ID of the transaction | Number | 9 |
| 2 | Bike image | Image of the bike | Image |  |
| 3 | Customer ID | ID of the customer | Number | 1 |
| 4 | Customer Name | Name of the customer | String | John Doe |
| 5 | Transaction Type | Type of the transaction | String | Normal |
| 6 | Status | Status of the transaction | String | Active |
| 7 | Deposit | The price to deposit (VND) | Number | 220000 |
| 8 | Last Pause | The exact time of last pause | String | 8/22/2023, 4:52:51 PM |
| 9 | Current Pay | The current price (VND) | Number | 447000 |
| 10 | Time Rent | The total time rent | String (hh:mm) | 24h25m |

**9.Postconditions**

**- None**

**Use Case “Resume active transaction.”**

**1.Use case code**

UC005

**2.Brief Description**

This use case describes the interaction between Customer and EcoBike Rental System when Customer wishes to resume his active transaction.

**3. Actors**

**- Customer**

**4. Preconditions**

Customer must have an active transaction and the status of the transaction is paused.

**5. Basic Flow of Events**

1. Customer click the “Active Transaction” button on the navigation bar
2. System displays active transaction information (Table B).
3. Customer clicks the “Pause” button to pause the active transaction
4. System changes the status of transaction from “paused” to “active” and update the last pause time.

**6. Alternative flows**

**- None**

**7.Input data**

**- None**

**8.Output data**

Table B-Output data of rental bike

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Data fields** | **Description** | **Display format** | **Example** |
| 1 | Transaction ID | ID of the transaction | Number | 9 |
| 2 | Bike image | Image of the bike | Image |  |
| 3 | Customer ID | ID of the customer | Number | 1 |
| 4 | Customer Name | Name of the customer | String | John Doe |
| 5 | Transaction Type | Type of the transaction | String | Normal |
| 6 | Status | Status of the transaction | String | Active |
| 7 | Deposit | The price to deposit (VND) | Number | 220000 |
| 8 | Last Pause | The exact time of last pause | String | 8/22/2023, 4:52:51 PM |
| 9 | Current Pay | The current price (VND) | Number | 447000 |
| 10 | Time Rent | The total time rent | String (hh:mm) | 24h25m |

**9.Postconditions**

**- None**

## Use case Rent Bike

**Use Case “Rent Bike”**

**1. Use case code**

UC005

**2. Brief Description**

This use case describes the interaction between Customer and EcoBike when Customer wishes to rent a bike.

**3. Actors**

**-** Customer

**4. Preconditions**

- Customer must log in to the system.

- Customer has no Active Transaction

- Bike is not rented

**5. Basic Flow of Events**

1. Customer input barcode, then choose Rent Button

2. Customer select Rent type

3. Customer input card information, then choose Deposit Button

4. The system call UC Create Transaction

5. The system notifies Rent Bike successfull, then return to Active Transaction Page

Table 5-Alternative flows of events for UC Rent Bike

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Location** | **Condition** | **Action** | **Resume location** |
| **1** | At Step 1 | If barcode is invalid | Notify Barcode invalid error | Resumes at Step 1 |
| **2** | At Step 1 | If customer already rented | Notify Customer rented error | Resumes at Step 1 |
| **3** | At Step 1 | If the bike is rented | Notify Bike is rented error | Resumes at Step 1 |
| **4** | At Step 4 | If the message from UC Create Transaction is not successful | Notify the error message of UC Create Transaction | Resumes at Step 3 |

**6. Input Data**

Table A-Input data of UC Rent Bike

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **No** | **Data fields** | **Description** | **Mandatory** | **Valid condition** | **Example** |
| 1 | Customer ID | ID of customer | Yes | Integer | 1 |
| 2 | Bike ID | ID of bike | Yes | Integer | 2 |
| 3 | Transaction Type | Type of Transaction | Yes | “Normal” or “24h” | Normal |
| 4 | Card Number | Card number | Yes | 16 digits number | 123412341234 |
| 5 | Cardholder Name | Cardholder name | Yes | First Character not blank. No special characters or number | Ha Huy Hoang |
| 6 | Issue Bank | Bank of Card | Yes | First Character not blank. No special characters or number | Bank A |
| 7 | CVV Number | Card CVV | Yes | 3 digits number | 901 |
| 8 | Month | Experation Month | Yes | From 1 to 12 | 2 |
| 9 | Year | Experation Month | Yes | From 00 to 99 | 24 |

**7. Output Data**

Table B-Output data of UC Rent Bike

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Data fields** | **Description** | **Display format** | **Example** |
| 1 | Message | response message | Show message on response | “Successful”,  “Fail to rent bike” |

## Use case Create Transaction

**1. Use case code**

UC006

**2. Brief Description**

This use case describes the interaction between Ecobike system and Interbank when the system wishes to create a new transaction.

**3. Actors**

**-** No

**4. Preconditions**

- The system has input data

**5. Basic Flow of Events**

1. The system check if input data related to transaction is valid

2. The system call UC Pay Money

3. The system create Transaction

4. The system notifies Successful and return to next step of UC which called it

Table 6-Alternative flows of events for UC Create Transaction

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Location** | **Condition** | **Action** | **Resume location** |
| **1** | At step 1 | If input data is invalid | - Notify Data invalid error to the UC which called it | Resumes at Step of UC which called it |
| **2** | At Step 2 | If the message from UC Pay Money is not successful | - Notify the error message of UC Pay Money to the UC which called it | Resumes at Step of UC which called it |
| **4** | At Step 3 | If Transaction can not be created | - Call UC Add Money to return deposit  - Notify Transaction can not be created error to the UC which called it | Resumes at Step of UC which called it |

Table A-Input data of UC Rent Bike

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **No** | **Data fields** | **Description** | **Mandatory** | **Valid condition** | **Example** |
| 1 | Transaction Type | Type of Transaction | Yes | “Normal” or “24h” | Normal |
| 2 | Card Number | Card number | Yes | 16 digits number | 123412341234 |
| 3 | Cardholder Name | Cardholder name | Yes | First Character not blank. No special characters or number | Ha Huy Hoang |
| 4 | Issue Bank | Bank of Card | Yes | First Character not blank. No special characters or number | Bank A |
| 5 | CVV Number | Card CVV | Yes | 3 digits number | 901 |
| 6 | Month | Experation Month | Yes | From 1 to 12 | 2 |
| 7 | Year | Experation Month | Yes | From 00 to 99 | 24 |

**7. Output Data**

Table B-Output data of UC Rent Bike

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Data fields** | **Description** | **Display format** | **Example** |
| 1 | Message | response message | Show message on response | “Successful”,  “Invalid Card Number” |
| 2 | Transaction | New Transaction | No |  |

## Use case Return Bike

**1. Use case code**

UC007

**2. Brief Description**

This use case describes the interaction between Customer and EcoBike system when the customer wishes to return the bike.

**3. Actors**

**-** Customer

**4. Preconditions**

- Customer has an active Transaction

**5. Basic Flow of Events**

1. Customer choose Pay Button

2. Customer input card information and dock return ID, then choose Pay Button

3. The system calls UC Create Invoice

4. The system notifies Invoice create successfull, then return to Main Page

Table 7-Alternative flows of events for UC Return Bike

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Location** | **Condition** | **Action** | **Resume location** |
| **1** | At Step 2 | If dock ID is not existed | - Notify Dock not existed error | Resumes at Step 2 |
| **2** | At Step 3 | If the message from UC Create Invoice is not successful | - Notify the error message of UC Create Invoice | Resumes at Step 2 |

Table A-Input data of UC ReturnBike

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **No** | **Data fields** | **Description** | **Mandatory** | **Valid condition** | **Example** |
| 1 | Transaction ID | ID of Transaction | Yes | Integer | 1 |
| 2 | Dock ID | ID of returned Dock | Yes | Integer | 2 |
| 3 | Transaction Type | Type of Transaction | Yes | “Normal” or “24h” | Normal |
| 4 | Card Number | Card number | Yes | 16 digits number | 123412341234 |
| 5 | Cardholder Name | Cardholder name | Yes | First Character not blank. No special characters or number | Ha Huy Hoang |
| 6 | Issue Bank | Bank of Card | Yes | First Character not blank. No special characters or number | Bank A |
| 7 | CVV Number | Card CVV | Yes | 3 digits number | 901 |
| 8 | Month | Experation Month | Yes | From 1 to 12 | 2 |
| 9 | Year | Experation Month | Yes | From 00 to 99 | 24 |

**7. Output Data**

Table B-Output data of UC Return Bike

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Data fields** | **Description** | **Display format** | **Example** |
| 1 | Message | response message | Show message on response | “Successful”,  “Fail to Create Invoice” |

## Use case Create Invoice

**1. Use case code**

UC005

**2. Brief Description**

This use case describes the interaction between Ecobike system and Interbank when the system wishes to create a new invoice.

**3. Actors**

**-** No

**4. Preconditions**

- Input Customer has an active Transaction

- The system has input value

**5. Basic Flow of Events**

1. The system check if input data related to invoice is valid

2. The system calls UC Pay Money

3. The system creates Invoice

4. The system calls UC Add Money to return deposit money

5. The system notifies Successful and return to next step of UC which called it

Table 8-Alternative flows of events for UC Create Invoice

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Location** | **Condition** | **Action** | **Resume location** |
| **1** | At step 1 | If input data is invalid | - Notify Data invalid error to the UC which called it | Resumes at Step of UC which called it |
| **2** | At Step 2 | If the message from UC Pay Money is not successful | - Notify the error message of UC Pay Money to the UC which called it | Resumes at Step of UC which called it |
| **4** | At Step 3 | If Invoice can not be created | - Call UC Add Money to return paying money  - Notify Invoice can not be created error to the UC which called it | Resumes at Step of UC which called it |

Table A-Input data of UC Rent Bike

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **No** | **Data fields** | **Description** | **Mandatory** | **Valid condition** | **Example** |
| 1 | Transaction Type | Type of Transaction | Yes | “Normal” or “24h” | Normal |
| 2 | Card Number | Card number | Yes | 16 digits number | 123412341234 |
| 3 | Cardholder Name | Cardholder name | Yes | First Character not blank. No special characters or number | Ha Huy Hoang |
| 4 | Issue Bank | Bank of Card | Yes | First Character not blank. No special characters or number | Bank A |
| 5 | CVV Number | Card CVV | Yes | 3 digits number | 901 |
| 6 | Month | Experation Month | Yes | From 1 to 12 | 2 |
| 7 | Year | Experation Month | Yes | From 00 to 99 | 24 |

**7. Output Data**

Table B-Output data of UC Rent Bike

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Data fields** | **Description** | **Display format** | **Example** |
| 1 | Message | response message | Show message on response | “Successful”,  “Invalid Card Number” |
| 2 | Invoice | New Invoice | No |  |

## Use case Get Balance

**Use Case “Get Balance”**

**1.Use case code**

UC00

**2.Brief Description**

This use case describes the interaction between Interbank and EcoBike when want to check if there are enough money in the card or not.

**3.Actors**

**-** Interbank

**4.Preconditions**

Do a transaction.

**5.Basic Flow of Events**

1. The system requests Interbank to get balance
2. Interbank gets balance data in the database.
3. Interbank considers if there are enough money or not.
4. Interbank returns result to the system.

**6.Alternative flows**

- None

**7.Input data**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Data fields** | **Description** | **Display format** | **Example** |
| 1 | cardNumber | Number of the card | String | 1234567890123456 |

**8. Output data**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Data fields** | **Description** | **Display format** | **Example** |
| 1 | balance | Balance of the card | Double | 900000.000 |

**9.Postconditions**

## Use case Pay Money

**Use Case “Pay Money”**

**1.Use case code**

UC00

**2.Brief Description**

This use case describes the interaction between Interbank and EcoBike when pay money.

**3.Actors**

**-** Interbank

**4.Preconditions**

Customer rent bike or return bike.

**5.Basic Flow of Events**

1. The system requests Interbank to pay money.
2. Interbank check all the validate conditions.
3. Interbank do the transaction.
4. Interbank returns result.

**6.Alternative flows**

**7.Input data**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Data fields** | **Description** | **Display format** | **Example** |
| 1 | cardNumber | Number of the card | String | 1234567890123456 |
| 2 | cardholderName | Name of the card holder | String | John Doe |
| 3 | issueBank | Name of the bank | String | Bank A |
| 4 | month | Expired month of the card | int | 12 |
| 5 | year | Expired year of the card | int | 23 |
| 6 | cvvCode | CVV code | String | 123 |
| 7 | amount | Amount customer has to pay | Double | 50000 |

**8. Output data**

**9.Postconditions**

## Use case Add Money

**Use Case “Add Money”**

**1.Use case code**

UC00

**2.Brief Description**

This use case describes the interaction between Interbank and EcoBike when Customer return Bike.

**3.Actors**

**-** Interbank

**4.Preconditions**

Customer returns the bike.

**5.Basic Flow of Events**

1. The system requests Interbank to return money.
2. Interbank does the transaction.
3. Interbank returns result.

**6.Alternative flows**

**7.Input data**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Data fields** | **Description** | **Display format** | **Example** |
| 1 | cardNumber | Number of the card | String | 1234567890123456 |
| 2 | amount | Amount system need to return | Double | 50000 |

**8. Output data**

**9.Postconditions**

# Supplementary specification

*<Presenting other requirements if necessary, including non-functional requirements such as performance, reliability, usability, and supportability; or other technical requirements such as database system, used technology…>*

## Functionality

* **User Authentication:** Users must be able to create accounts and log in securely. Passwords should be stored securely, preferably using hashing algorithms.
* **Bike Rental:** Users should be able to view available bikes, rent bikes, and return bikes. The system should calculate rental fees accurately based on the selected bike category, rental duration, and any additional charges.
* **Reservation:** Users should have the option to reserve bikes for a specified future time. Reserved bikes should be guaranteed to be available when the reservation time comes.
* **Admin Panel:** Administrative users should have access to an admin panel where they can manage bike inventory, view transaction history, and generate reports.
* **Transaction History:** Users and admins should be able to view their transaction history, including rental details, invoices, and payment records.

## Usability

* **User-Friendly Interface:** The user interface should be intuitive and easy to navigate, ensuring that users can rent bikes with minimal training or assistance.
* **Accessibility:** The system should adhere to accessibility standards (e.g., WCAG) to accommodate users with disabilities.
* **Mobile Responsiveness:** The system should be responsive and accessible on various devices, including smartphones and tablets.

## Reliability

* **System Availability:** The system should aim for high availability, with minimal downtime. Specify a target uptime percentage (e.g., 99.9%).
* **Data Integrity**: Implement measures to ensure data integrity, including regular backups and data recovery procedures.

## Performance

* **Response Times:** Define specific response time requirements for critical operations, such as bike rental and transaction processing. For example, bike rental should be completed in under 5 seconds.
* **Scalability:** The system should be designed to scale horizontally to accommodate increased usage and data growth.

## Supportability

* **Maintenance and Updates**: Specify that the system should be designed with ease of maintenance in mind, including clear code documentation and version control practices.
* **Technical Support:** Describe the availability of technical support for users and administrators, including contact methods and response times.