HANOI UNIVERSITY OF SCIENCE AND TECHNOLOGY
School of Information and communications technology

# Software Requirement Specification

Version 1.0

# Capstone Project
# Subject: ITSS Software Development

Group Number: 7
List of participants:
Nguyễn Vũ Minh 20194801
Lê Huy Hoàng 20194766
Bùi Mạnh Tú 20194870

*Hanoi, December, 2021*

# Table of contents

# 1 Introduction

## *1.1 Objective*

This Software Requirement Specification *(or SRS for short)* purpose is to clarify the main aspects of the Capstone project : The users, the use cases, the processes inside the software as well as declaring the main demands for the software: functionality, usability, reliabity,... With this, the making of the software will become easier.

## *1.2 Scope*

- The product: **EcoBikeRental software system**
- The using: Allowing the user to rent and return bikes from docking stations in EcoPark (or even other places where can satisfy the infrastructure) through the phone app.
- The System will control the renting and returning of the bikes, as well as calculating and charging the user's fee.

## *1.3 Glossary*

| Terms | Explanation |
|---|---|
| **Standard bicycle** | 01 saddle, 01pedal, and 01 rear seat in the back. |
| **Standard e-bike** | a standard bike, but it has an integrated electric motor for assist propulsion |
| **Twin bike** | 02 saddle, 02 pedal, and 01 rear seat with no integrated electric motor |

## *1.4 References*

*"EcoBikeRental-ProblemStatement-EN.pdf"* ***- Dr. Nguyen Thi Thu Trang***

# 2 Overall Description

## *2.1 Actors*

- Customers: the main user of the program
- Interbank: the one who do all the transaction
- Software system: the software itself

## *2.2 Use case diagrams*

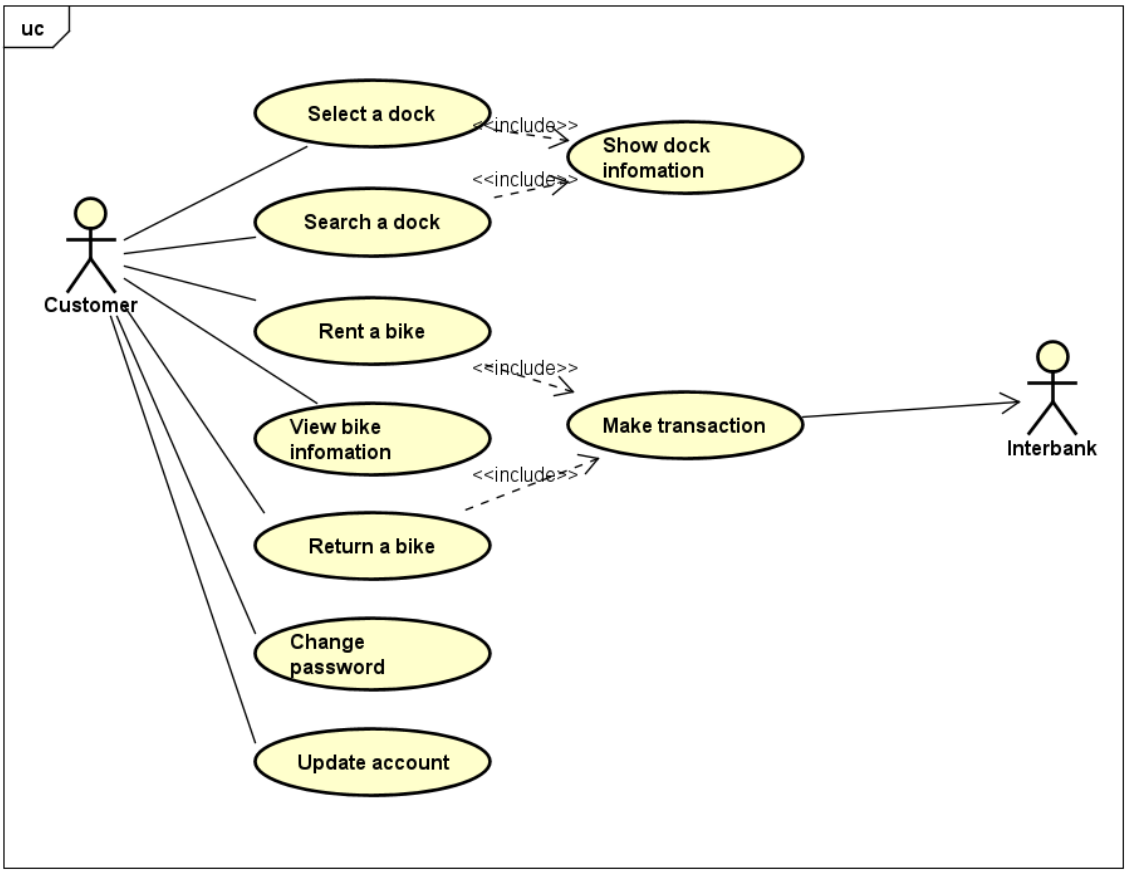


***Figure 1. EcoBikeRental software Use Case diagram***

## 2.3 Business processes

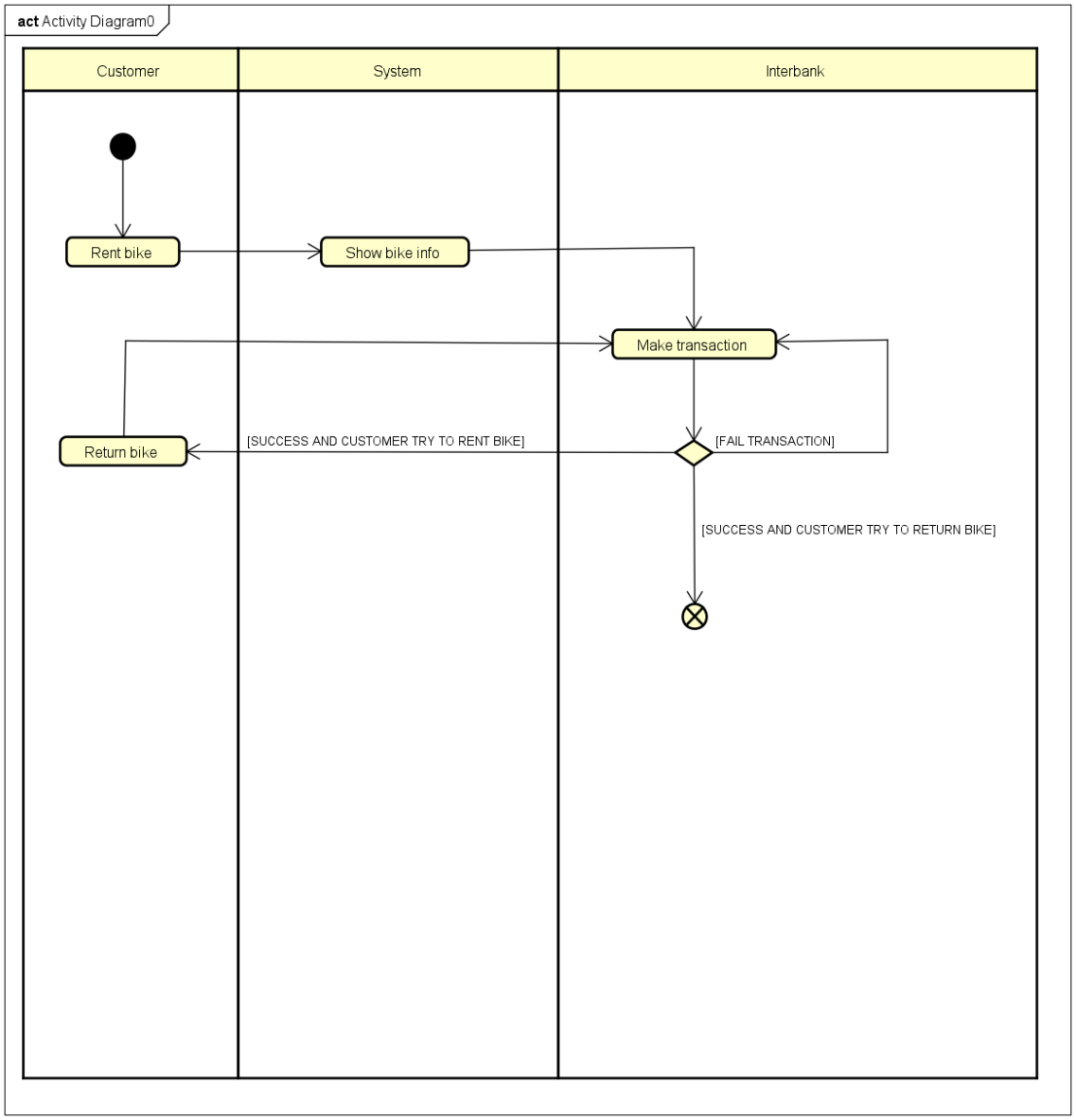


***Figure 2. EcoBikeRental software Business processes***

# 3 Detailed Requirements

## *3.1 Use case specification for "Show Bike Infomation"*

# Use Case "Show bike info"

### 1. Brief description

This use case describes the interaction between the user and the system when the user wants to view the information of the bike which they are renting.

### 2. Actors

#### 2.1. Customer

#### 2.2. Software system

### 3. Preconditions

The user must first had rent a bike.

### 4. Basic flow

Step 1. The user selects to view the information of their current renting bike

Step 2. Check if the user has rent a bike

Step 3. Take the information about the rented bike from the sever

Step 4. Display the bike information

Step 5. End the use case

### 5. Alternative flow

***Table 1. Alternative flows of events for UC "Show bike info"***

| No | Location | Condition | Action | Resume location |
|---|---|---|---|---|
| 1 | At step 2 | If the user hasn't rent a bike yet | End the use case | End the use case |
| 2 | At step 4 | If the bike is and e-bike | Show the extra e-bike information | Step 5 |

### 6. Input Data

*None*

### 7. Output Data

*Table 2. Output data of bike information*

| No | Data fields | Description | Display format | Example |
|---|---|---|---|---|
| 1 | Bike type | | | Giant XP |
| 2 | Renting time | | X h : Y ‘ | 1h15’ |
| 3 | Paid amount | | - Positive number<br>- Use “.” to separate thousands<br>- Currency is VND | 34.000 VND |
| 4 | Bike status | Depends on bike type | | 34% |

**8. Postconditions**

*None*

### *3.2 Use case specification for “Rent Bike”.*

## Use Case “Rent Bike”

**1. Brief Description**

This use case describes the interaction between software and the user when the user wishes to rent a bike by scanning the barcode on the lock.

**2. Actors**

**2.1 Customer**

**3. Preconditions**

*None*

**4. Basic Flow of Events**

Step 1. User scan barcode on the lock

Step 2. The software checks the barcode

Step 3. The software displays the information of the corresponding bike

Step 4. The software call “make transaction” use case

Step 5. The software opens the lock of the bike, allowing user to use the bike

**5. Alternative flows**

*Table 3. Alternative flows of events for UC "Rent bike"*

| No | Location | Condition | Action | Resume location |
|---|---|---|---|---|
| 1 | At step 2 | If the barcode is invalid | ▪ The software notifies the user that it fail to recognize the barcode | At step 1 |

## 6. Activity diagrams

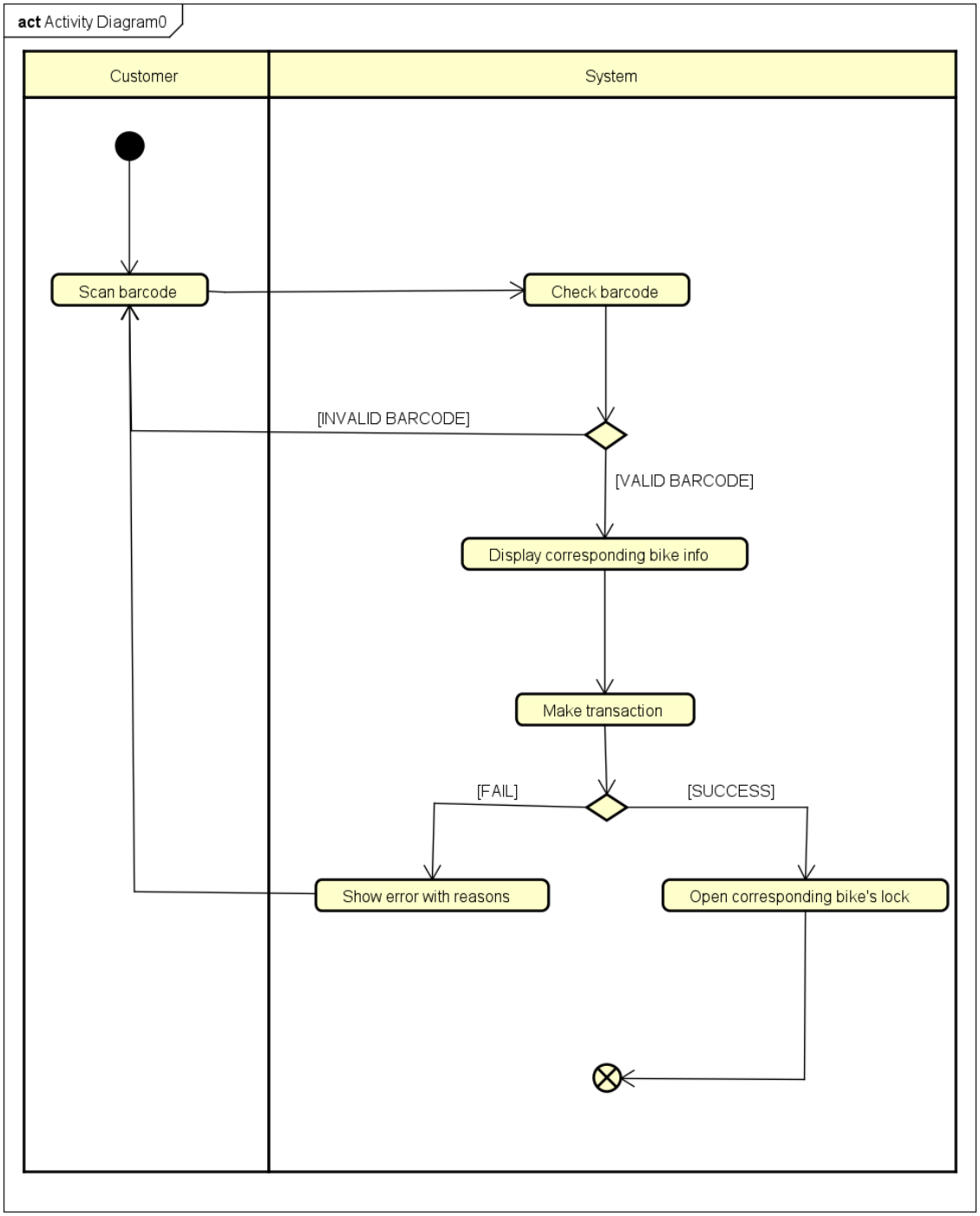


***Figure 3. Use case "Rent bike" Activity diagram***

## 7. Input data

None

## 8. Output data

***Table 4. Output data for Bike information***

| No | Data fields | Description | Display format | Example |
|---|---|---|---|---|
| 1. | License plate | | | 52-N6 1234 |
| 2. | Bike type | Only 3 types: standard bicycle, standard e-bike, twin bike | | standard bicycle |
| 3. | Current battery percentage | Only shown on standard e-bike | • Positive Integer<br>• Less than or equal to 100<br>• Have '%' at the end<br>• Right alignment | 99% |

**9. Postconditions**

If successful, then the lock on the scanned bike will be unlocked and the system will deduct 40% of the value of the bike in user's card or account.

### *3.3 Use case specification for "Make Transaction"*

## Use Case "Make Transaction"

### 1. Brief Description

This use case describes the interaction between software, the user and the interbank when transaction occurred

### 2. Actors

### 2.1 Customer

### 2.2 Interbank

### 3. Preconditions

After user scan a barcode to rent bike or return bike

### 4. Basic Flow of Events

Step 1. The software displays the payment screen

Step 2. The user submits their card information

Step 3. The software checks the format of the information

Step 4. The software asks the Interbank to make transaction

Step 5. The Interbank processes the transaction

Step 6. The software displays and saves transaction information

Step 7. The software sends an email of transaction info to the user

## 5. Alternative flows

***Table 5. Alternative flows of events for UC "Show bike info"***

| No | Location | Condition | Action | Resume location |
|---|---|---|---|---|
| 1 | At step 3 | If the transaction info have invalid format | ▪ The software inform customers that the transaction info have invalid format | At step 1 |
| 2 | At step 4 | If the transaction info is invalid | ▪ The software inform customers that the info is invalid | At step 1 |
| 3 | At step 5 | If the balance is not enough | ▪ The software inform customers that the balance is not enough | At step 1 |

## 6. Activity diagrams

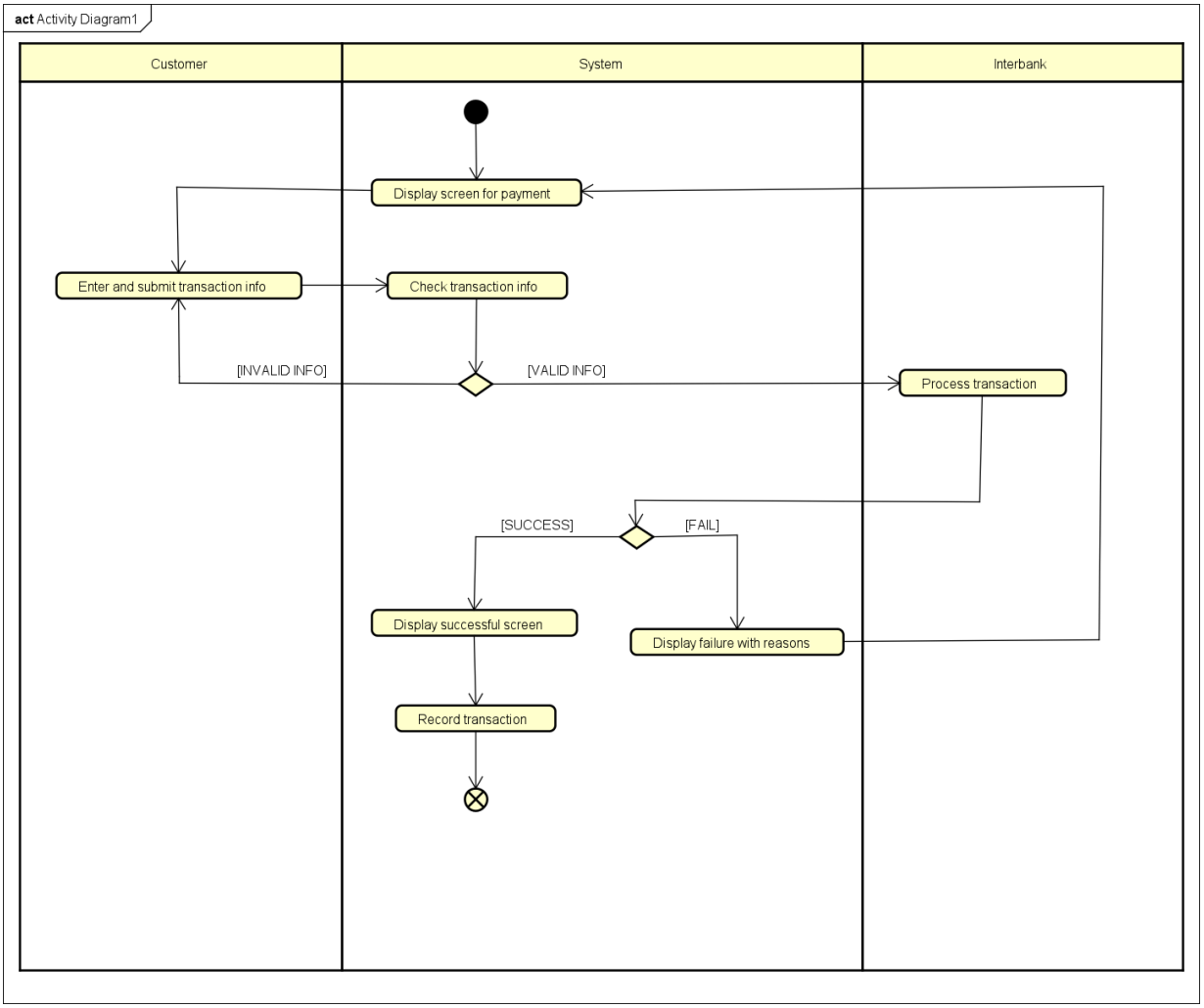


***Figure 4. Use case "Make transaction" Activity diagram***

## 7. Input data

***Table 6. Input data of payment method***

| No | Data fields | Description | Mandatory | Valid condition | Example |
|---|---|---|---|---|---|
| 1. | Card holder name | | Yes | | NGUYEN VU MINH |
| 2. | Card number | | Yes | | 1254 2679 5624 7824 |
| 3. | Issuing bank | | Yes | | VIETINBANK |

| | | | | | |
|---|---|---|---|---|---|
| 4. | Expiration date | | Yes | Consist of month and last 2 digits of year | 10/25 |
| 5. | Security code | | Yes | | 564 |

**8. Output data**

***Table 7. Output data of successful transaction screen***

| No | Data fields | Description | Display format | Example |
|---|---|---|---|---|
| 1. | Transaction ID | Unique ID for the transaction | Unique string of number and uppercase characters | 3CWK98J37FK9 |
| 2. | Card holder name | | | NGUYEN VU MINH |
| 3. | Charged amount | Total money the customer has paid for the order | • Comma for thousands separator<br>• Positive integer<br>• Right alignment | 100,000 |
| 4. | Transaction description | Description of the order | | Rent a standard bike |
| 5. | Balance | Amount of money left in customer's bank account | • Comma for thousands separator<br>• Positive integer<br>Right alignment | 123,456,789 |
| 6. | Transaction data and time | Date and time of the transaction | In the format of hour:minute day/month/year | 18:30 28/02/2021 |

**9. Postconditions**

If success, then the lock on the scanned bike will be unlocked and the system will deduct 40% of the value of the bike in user's card or account.

### *3.4 Use case specification for "Return Bike"*

## Use Case "Return Bike"

## 1. Brief Description

This use case describes the interation between users and the software when returning a bike after renting it.

## 2. Actors

### 2.1 Customer

### 2.2 Software

### 2.3 Interbank

## 3. Preconditions

After customer scan a barcode to rent bike or return bike

## 4. Basic Flow of Events

Step 1. Customer opens Return Bike function

Step 2. Software checks for empty docks

Step 3. Software displays nearest empty dock together with all empty docks marked on map

Step 4. Customer chooses an empty dock

Step 5. Software calculates the deposit, rental fees, and refunds (if any)

Step 6. Software displays the invoice

Step 7. Software calls the "make transaction" use case

## 5. Alternative flows

***Table 8. Alternative flows of events for UC "Return bike"***

| No | Location | Condition | Action | Resume location |
|---|---|---|---|---|
| 1 | At step 3 | If the software cannot find any empty docks | ▪ The software asks user ti wait until there is a vacant dock | At step 2 |
| 2 | At step 6 | If customer rents 24-hour pass | ▪ Software checks for the rented period<br>▪ Software refunds if elapsed time is less than 24 hours or deducts if it exceeds | At step 5 |

## 6. Activity diagrams

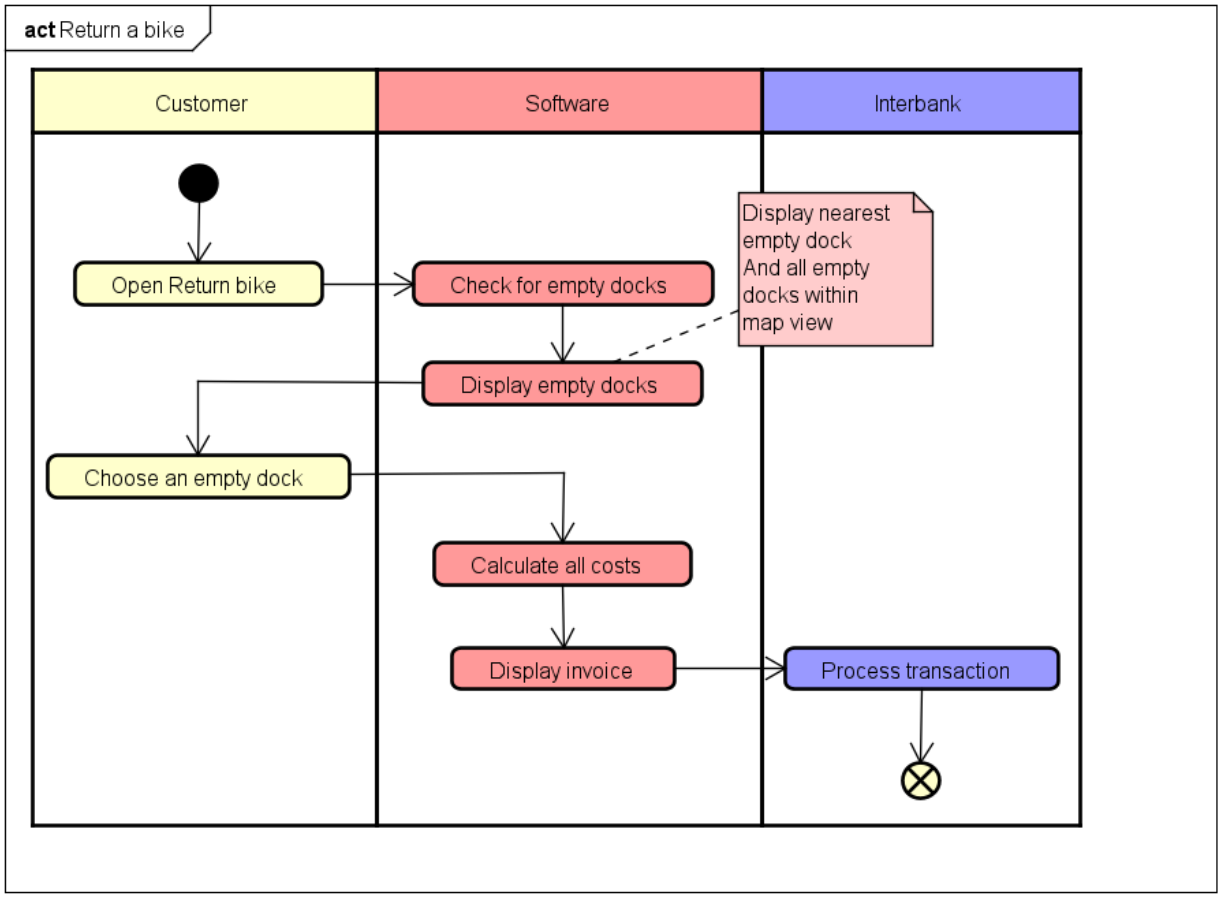


***Figure 5. Return bike Activity diagram***

## 7. Input data

*None*

## 8. Output data

***Table 9. Output data for renting invoice***

| No | Data fields | Description | Display format | Example |
|---|---|---|---|---|
| 1 | Bike type | | | Giant XP |
| 2 | Bike status | Depends on bike type | | 34% |
| 3 | Renting pricing type | 24-hour pass or normal | | |
| 4 | Renting time | | X h : Y ‘ | 1h15’ |

| | | | | |
|---|---|---|---|---|
| 5 | Deposits | | ▪ Positive number<br>▪ Use "." to separate thousands<br>▪ Currency is VND | +30.000 VND |
| 6 | Rental fees | | | -100.000 VND |
| 7 | Refunds | Optional | | +0 VND |
| 8 | Total charged amount | | | -70.000 VND |

## 9. Postconditions

*None*

# 4 Supplementary specification

## *4.1 Functionality*

- In any flow of events, user can choose to navigate back to the previous state/screen.

- Should there be a database, API call error, or any system error rather than user's faults, software must notify user about it.

- General format:

  - Right-algined number
  - Left-aligned text
  - Justify-aligned paragraph

## *4.2 Usability*

- Allow novice users to use without any training

- If user first uses the software, general instructions should be given.

- If there is an error, detailed instructions on the error and how to fix it should be given.

## *4.3 Reliability*

Operate in an average of 200 hours without failure.

## *4.4 Performance*

- Serve 100 users at the same time without noticeable loss of performance.

- The response time for the system is 1 second at normal and 2 seconds during a peak load if it is not explicitly stated.

- The response time for any transaction must not exceed 1 second.

## *4.5 Supportability*

The system can be repaired within 2 hours after any typical failure.

## *4.6 Other requirements*

*None*