HANOI UNIVERSITY OF SCIENCES AND TECNOLOGY
School of Information and communications technology

Software Requirement Specification

Version 1.0

EcoBike Rental
Subject: ITSS Software Development

Group Number 6

Đặng Việt Anh: 20176686 – 20%
Trần Quý Dương: 20160869 - 22%
Lại Tiến Đức: 20176722 - 35%
Phạm Vũ Minh: 20176818 - 23%

Hanoi, 12/2020

1 Table of contents

1 Introduction

a. Objective

This document provides developers an description of software's flow of events, its components and operations.

b. Scope

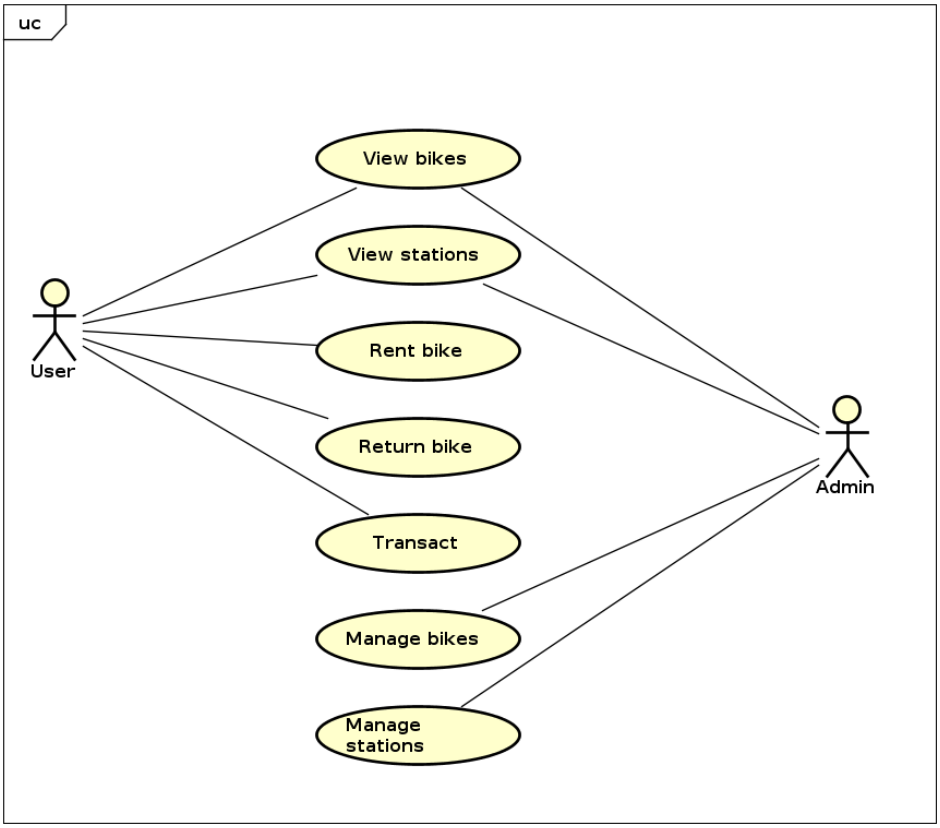
- Name: ECOBIKE Rental

- Function:
  - CRUD system information (Admin only)
- Show docks individually and as a list
- Show bike information before and during renting
- Disable bike renting when another bike is currently in use
- Rent and return bike
- Prevent user from closing the app before returning current bike

- Application: Stimulate an actual bike renting service in a park. The user can also pay with a credit card stimulation.

2 Use case diagram

2.1 General use case diagram



Description of actors:

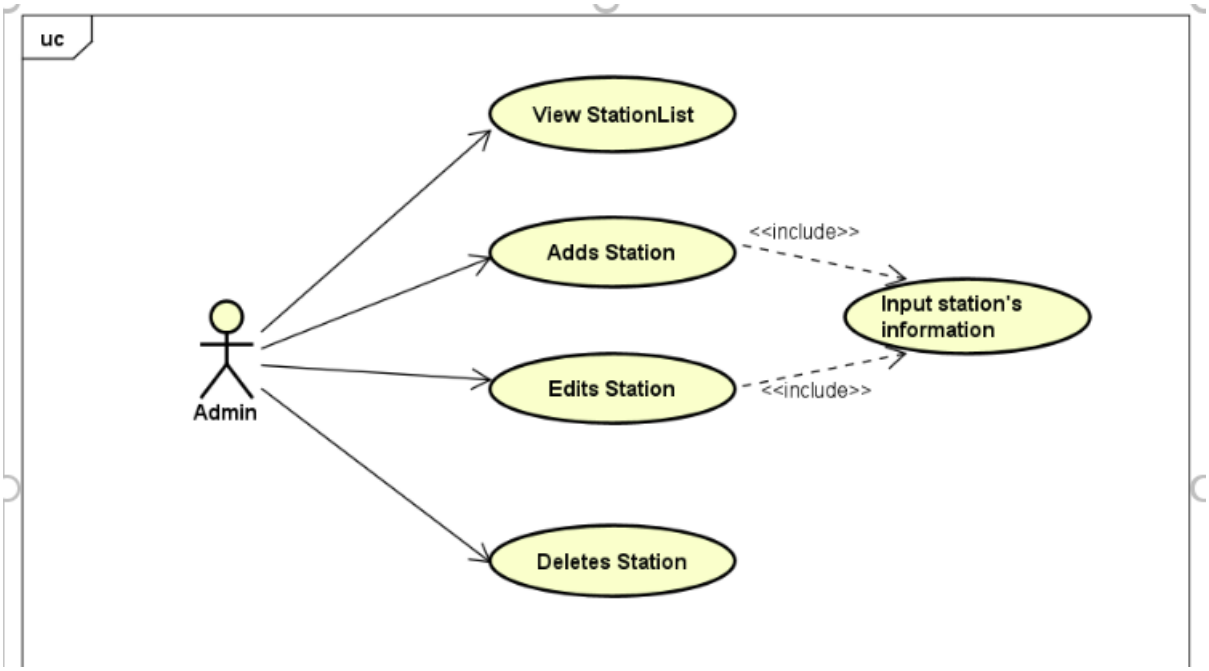
- User:
- Admin:

Explanation:

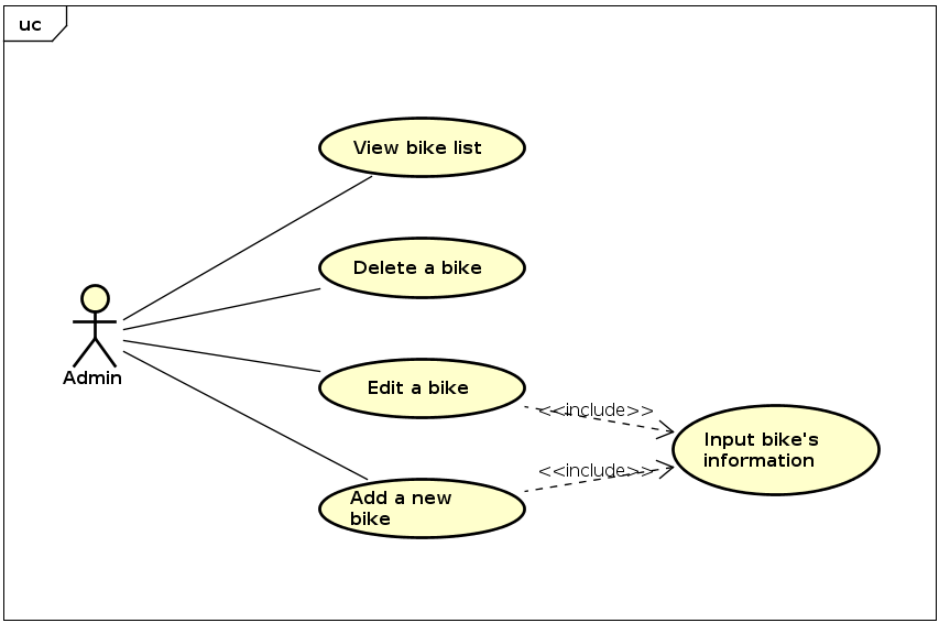
Composite use case:

- Manage bikes
- Manage stations

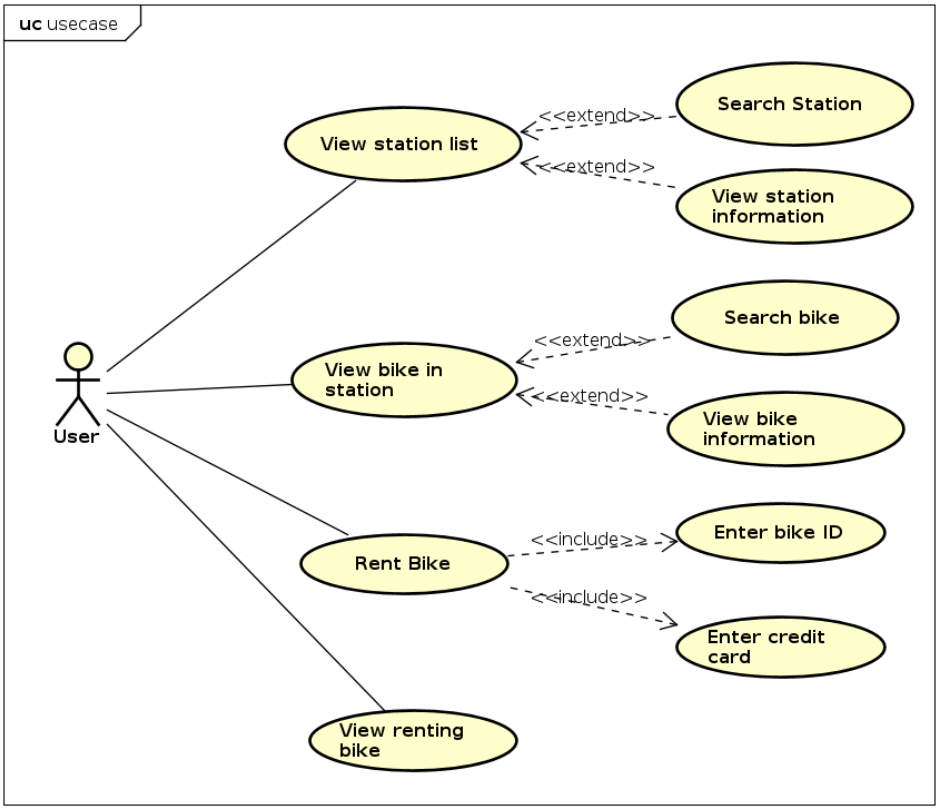
2.2 Use case diagram for “Manage Stations”



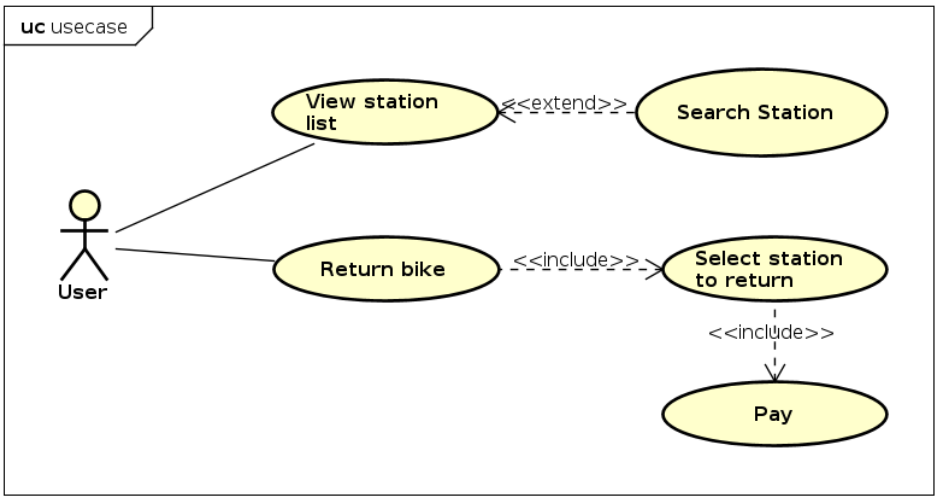
2.3 Use case diagram for “Manage Bikes”



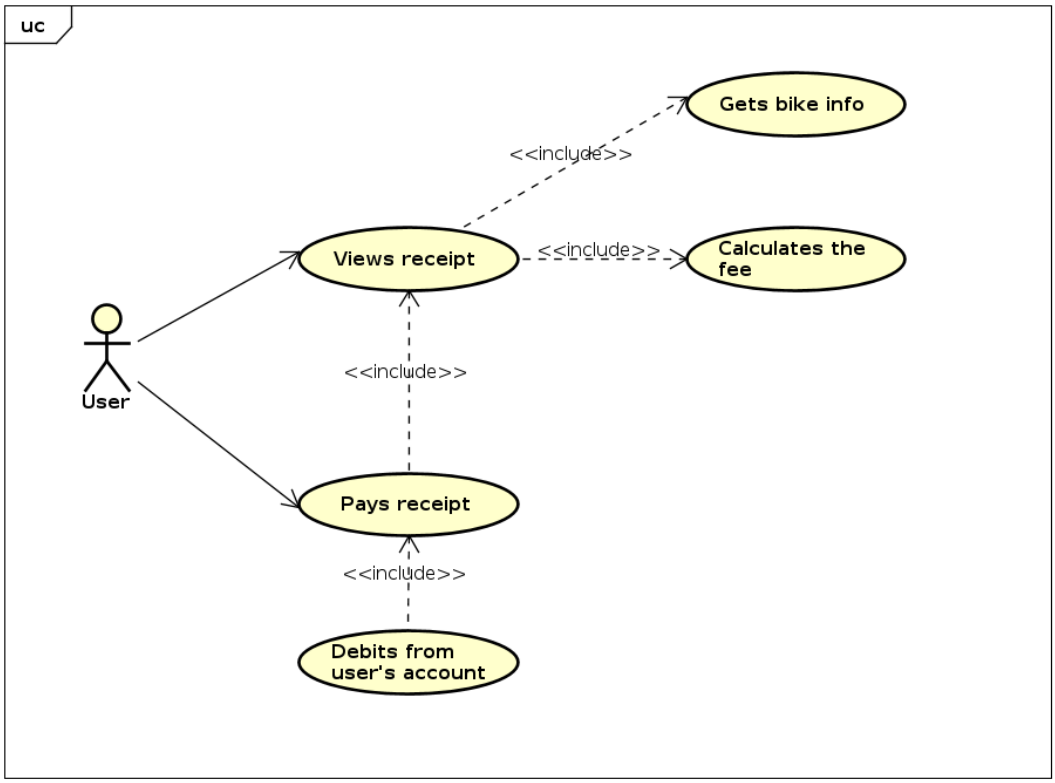
2.4 Use case diagram for “Rent Bike”



2.5 Use case diagram for “Return Bike”



2.6 Use case diagram for “Payment”



3 Use case specification

2.1. Use case "AdminDockStation"

Use case ID	UC001	**Use case name**	AdminDockStation
Actors	Admin, System		
Pre-Condition(s)	- Admin account must be on the system - Admin has modified access right.		

Basic Path (Success)

No	Proceeded by	Actions
1	Admin	chooses 'Add'
2	System	displays 'AddDockStation' interface
3	Admin	enters inputs for adding new station to DB
4	System	checks input for adding(name and address can't be both same one's)
5	System	updates new station to DB
6	System	prompts "Success to add!" and displays admin panel
7	Admin	clicks row and chooses 'Edit'
8	System	displays 'EditDockStation' interface

	9	Admin	enters inputs for editting station in DB
	10	System	checks input for editting(name and address can't be both same one's)
	11	System	updates station to DB
	12	System	prompts "Success to update!" and displays admin panel
	13	Admin	Clicks row and chooses 'Delete'
	14	System	checks input for deleting(check whether have bike or not)
	15	System	updates DB with deleting
	16	System	prompts "Success to delete!" and displays admin panel

Alternative Paths	**No**	**Proceeded by**	**Actions**
	5a	System	Can't add new station
	6a	System	Error prompt: "Fail to add!"
	11a	System	Can't update station in DB
	12a	System	Error prompt: "Fail to update!"
	15a	System	Can't delete station in DB
	16a	System	Error prompt: "Cannot delete, this station has bikes!"

Post-Condition(s)	- Admin proceeds modifying successfully. - System save the current modification and update to DB

2.2 Use case "Add New Bike"

Use case ID	UC002	**Use case name**	Add New Bike
Actors	Admin, System		
Pre-Condition(s)	- Admin successfully logged in. - Admin has modify rights.		

Basic Path (Success)	**No**	**Proceeded by**	**Actions**
	1.	Admin	Click "Add New" button
	2.	System	Display bike information screen
	3.	System	Generate and display new bike's ID

	4.	Admin	Input new bike's information (input description below)
	5.	Admin	Click "Add" button
	6.	System	Validate if the new bike name is unique
	7.	Admin	Validate if the new bike license plate is unique
	8.	System	Add new bike to database
	9.	System	Update bike list
	10.	System	Show updated bike list

Alternative Paths

No	Proceeded by	Actions
4a.	System	IF any input field is empty, display error message "Invalid Input"
4b.	System	IF input for weight is not a float, display error message "Invalid Input"
6a.	System	IF new bike name is not unique, display error message
7a.	System	IF new license plate is not unique, display error message..

Post-Condition(s)	None

* Input data of added bike information includes these following fields:

ID	Data field	Description	Mandatory	Valid condition	Example
1.	Name	Bike name	Yes	Must not exist in database	Argon 18
1.	Type	Bike type	Yes	Must be "NormalBike", "EcoBike" or "TwinBike"	EcoBike
3.	License Plate	Bike's license plate	Yes	Must not exist in database	068581324-0
4.	Weight	Bike's weight	Yes	Must be a float	13.2
5.	Manufacturing date	Bike's date of manufacturing	Yes	Must be a valid date	2020-12-01
6.	Producer	Bike's producer	Yes	None	Trek
7.	StationID	Station ID of which bike is added to	Yes	Selected station's dock must not be full	1

2.3 Use case "Rent Bike"

Use case ID	UC003	**Use case name**	Rent Bike

<table>
<tr><td>Actors</td><td>User, System</td></tr>
<tr><td>Pre-Condition(s)</td><td>- User login the app
- User has bank account</td></tr>
<tr><td>Basic Path
(Success)</td><td>
<table>
<tr><th>No</th><th>Proceeded by</th><th>Actions</th></tr>
<tr><td>1.</td><td>User</td><td>Login in the app</td></tr>
<tr><td>2.</td><td>System</td><td>Display station list screen</td></tr>
<tr><td>3.</td><td>User</td><td>Click rent button</td></tr>
<tr><td>4.</td><td>System</td><td>Check whether bike is renting or not</td></tr>
<tr><td>5.</td><td>System</td><td>Display rent bike screen</td></tr>
<tr><td>6.</td><td>User</td><td>Input bike id</td></tr>
<tr><td>7.</td><td>System</td><td>Chekc bike id</td></tr>
<tr><td>8.</td><td>User</td><td>Input credit card</td></tr>
<tr><td>9.</td><td>System</td><td>Check credit card</td></tr>
<tr><td>10.</td><td>User</td><td>Click rent button</td></tr>
<tr><td>11.</td><td>System</td><td>Record information of user, bike, account</td></tr>
</table>
</td></tr>
<tr><td>Alternative Paths</td><td>
<table>
<tr><th>No</th><th>Proceeded by</th><th>Actions</th></tr>
<tr><td>3a</td><td>User</td><td>Click view station detatil</td></tr>
<tr><td>3a1</td><td>System</td><td>Display station detail screen</td></tr>
<tr><td>3a2</td><td>User</td><td>Click view bike list</td></tr>
<tr><td>3a3</td><td>System</td><td>Display list bike in station screen</td></tr>
<tr><td>3a4</td><td>User</td><td>Click view bike detatil</td></tr>
<tr><td>3a5</td><td>System</td><td>Display bike detail screen</td></tr>
<tr><td>3a6</td><td>User</td><td>Click rent bike</td></tr>
<tr><td>7a</td><td>System</td><td>Display error : invalid bike id</td></tr>
<tr><td>9a</td><td>System</td><td>- IF card_number is not exist , display error
- IF balance is not enough, display error</td></tr>
</table>
</td></tr>
<tr><td>Post-Condition(s)</td><td>- User rented bike successfully
- System records the information of user, bike, bank account</td></tr>
</table>

* Input data of renting information includes these following fields:

ID	Data field	Description	Mandatory	Valid condition	Example
2.	bike id		Yes	bike id exist	12
3.	credit card		Yes	card exists, has enough money	

2.4 Use case “Return Bike”

Use case ID	UC004	Use case name	Return Bike

<table>
<tr><td>Actors</td><td>User, System</td></tr>
<tr><td>Pre-Condition(s)</td><td>- User rented a bike</td></tr>
<tr><td>Basic Path
(Success)</td><td>
<table>
<tr><th>No</th><th>Proceeded by</th><th>Actions</th></tr>
<tr><td>1.</td><td>User</td><td>Click return button</td></tr>
<tr><td>2.</td><td>System</td><td>Display station list screen</td></tr>
<tr><td>3.</td><td>User</td><td>Select station to return</td></tr>
<tr><td>4.</td><td>System</td><td>Check empty dock in station</td></tr>
<tr><td>5.</td><td>User</td><td>Click payment button</td></tr>
<tr><td>6.</td><td>System</td><td>Display payment screen</td></tr>
<tr><td>7.</td><td>User</td><td>Click pay button</td></tr>
<tr><td>8.</td><td>System</td><td>Update record</td></tr>
<tr><td>9.</td><td>System</td><td>Refund money</td></tr>
<tr><td>10.</td><td>System</td><td>Show success payment</td></tr>
</table>
</td></tr>
<tr><td>Alternative Paths</td><td>
<table>
<tr><th>No</th><th>Proceeded by</th><th>Actions</th></tr>
<tr><td>4a</td><td>System</td><td>IF station doesn's have empty dock, display error: can not returnn to this station</td></tr>
</table>
</td></tr>
<tr><td>Post-Condition(s)</td><td>- User return bike successfully
- System records the information of user, bike, bank account
- User's account bank debited</td></tr>
</table>

2.5 Use case “Payment”

<table>
<tr><td>Use case ID</td><td>UC005</td><td>Use case name</td><td>Payment</td></tr>
<tr><td>Actors</td><td colspan="3">User, System</td></tr>
<tr><td>Pre-Condition(s)</td><td colspan="3">- User account must be on the system
- User has used bike-renting service.</td></tr>
<tr><td>Basic Path
(Success)</td><td colspan="3">
<table>
<tr><th>No</th><th>Proceeded by</th><th>Actions</th></tr>
<tr><td>1.</td><td>User</td><td>choose ‘Payment’</td></tr>
<tr><td>2.</td><td>System</td><td>calculate the fee user needs to pay</td></tr>
<tr><td>3.</td><td>System</td><td>displays ‘Payment’ interface</td></tr>
<tr><td>4.</td><td>User</td><td>checks payment information</td></tr>
<tr><td>5.</td><td>User</td><td>proceeds payment</td></tr>
<tr><td>6.</td><td>System</td><td>debits and refund deposit to user’s bank account</td></tr>
<tr><td>7.</td><td>System</td><td>Saves transaction information and sends to user’s registered email</td></tr>
</table>
</td></tr>
</table>

Alternative Paths	No	Proceeded by	Actions
	6a.	System	Error prompt: Account does not have enough balance to proceed payment.
	7.a	System	Error prompt: Error when connecting to database..
Post-Condition(s)	- User proceeds payment successfully. - System save the current transaction and sends to user's registered email.		

* Input data of payment information includes these following fields:

ID	Data field	Description	Mandatory	Valid condition	Example
4.	Credit card information	User's credit card information	Yes	Must be present in database.	h.anh@gmail.com
5.	Total use time	Time from when the user rents to when the user returns bike.	Yes	Must be calculated correctly.	1 hour 14 minutes
3.	Money to pay	The fee user must to pay	Yes	Must be correct based on use time.	500.000 (VNĐ)

4 Glossary

This document is used to define terminology specific using in this project EcoBike Rental

4.1 Station

- A class contains information of a station

4.2 Bike

- A class contains information of a bike

4.3 Record

- A class store information of reting bike action: user, bike, time.

5 Supplementary specification

5.1 Functionality

- If having any error when manipulation with database, inform that error related to DBMS not to user

5.2 Performance

5.3 Reliability