|  |  |
| --- | --- |
|  | **MINISTRY OF EDUCATION AND TRAINING** |

**FPT UNIVERSITY**

|  |  |
| --- | --- |
| Capstone Project Document | |
| **Face Detection For Delivery Payment** | |
| **Group 5 - IS** | |
| **Group members** | Nguyễn Văn Hiếu – SE61961 (Leader)  Đinh Phú Thắng – SE62528  Trần Trọng Nghĩa – SE62278  Nguyễn Hữu Lâm– SE61234 |
| **Supervisor** | Nguyễn Huy Hùng |
| **Ext. Supervisor** | N/A |
| **Project Code** | FDDP |

– **Ho Chi Minh City, 20 January 2019** –

[A. Introduction 11](#_Toc6510051)

[1. Project Information 11](#_Toc6510052)

[2. Introduction 12](#_Toc6510053)

[3. Current Situation 12](#_Toc6510054)

[4. Problem Definition 12](#_Toc6510055)

[5. Proposed Solution 13](#_Toc6510056)

[**5.1 Feature functions** 13](#_Toc6510057)

[**6.3** **Advantages and disadvantages** 13](#_Toc6510058)

[6. Functional Requirements 13](#_Toc6510059)

[7. Role & Responsibility 14](#_Toc6510060)

[**B.** **Software Project Management** 15](#_Toc6510061)

[**1.** **Problem Definition** 15](#_Toc6510062)

[**1.1** **Name of this Capstone Project** 15](#_Toc6510063)

[**1.2** **Problem Abstract** 15](#_Toc6510064)

[**1.3** **Project Overview** 15](#_Toc6510065)

[**1.3.1** **Current Situation** 15](#_Toc6510066)

[**1.3.2** **The Proposed System** 16](#_Toc6510067)

[**1.3.2.1** **Website** 17](#_Toc6510068)

[**1.3.2.2** **Mobile Application** 17](#_Toc6510069)

[**1.3.3** **Boundaries of the System** 18](#_Toc6510070)

[**1.3.4** **Future Plans** 18](#_Toc6510071)

[**1.3.5** **Development Environment** 18](#_Toc6510072)

[**1.3.5.1** **Hardware requirements** 18](#_Toc6510073)

[**For Server** 18](#_Toc6510074)

[**For Computer** 19](#_Toc6510075)

[**For Tablet/Mobile** 19](#_Toc6510076)

[**1.3.5.2** **Software requirements** 19](#_Toc6510077)

[**For Web** 19](#_Toc6510078)

[**For Mobile Application** 20](#_Toc6510079)

[**2.** **Project Organization** 20](#_Toc6510080)

[2.1 Software Process Model 21](#_Toc6510081)

[2.2 Roles and Responsibilities 22](#_Toc6510082)

[2.3 Tools and Techniques 22](#_Toc6510083)

[**3.** **Project Management Plan** 23](#_Toc6510084)

[**3.1. Sprint Backlog** 23](#_Toc6510085)

[**3.2. Product Backlog** 23](#_Toc6510086)

[**3.3. All Meeting Minutes** 23](#_Toc6510087)

[**4.** **Coding Convention** 23](#_Toc6510088)

[C. Software Requirement Specification 24](#_Toc6510089)

[**1.** **User Requirement Specification** 24](#_Toc6510090)

[**1.1** **Guest Requirement** 24](#_Toc6510091)

[**1.2** **Customer Requirement** 24](#_Toc6510092)

[**1.3** **Admin Requirement** 24](#_Toc6510093)

[**1.4** **Shipper Requirement** 25](#_Toc6510094)

[**2.** **System Requirement Specification** 25](#_Toc6510095)

[2.1 External Interface Requirement 25](#_Toc6510096)

[**2.1.1 User Interface** 25](#_Toc6510097)

[**2.1.2 Hardware Interface** 26](#_Toc6510098)

[**2.1.3 Software Interface** 26](#_Toc6510099)

[**2.1.4 Communication Protocol** 26](#_Toc6510100)

[2.2 System Overview Use Case 27](#_Toc6510101)

[Figure 2: System Overview Use Case 27](#_Toc6510102)

[2.3 List of Use Case 27](#_Toc6510103)

[**2.3.1 <Guest> Overview Use Case** 27](#_Toc6510104)

[**2.3.1.1 <Guest> Register Account** 28](#_Toc6510105)

[Figure 4: <Guest> Registration Use Case 28](#_Toc6510106)

[Table : <Guest> Registration Use Case 30](#_Toc6510107)

[Table : <Guest > Log in 32](#_Toc6510108)

[**2.3.2**  **<Customer> Overview Use Case** 33](#_Toc6510109)

[Figure 6: < Customer > Over view use case 33](#_Toc6510110)

[**2.3.2.1 < Customer > Search store** 33](#_Toc6510111)

[Figure 9: < Customer > Search store 33](#_Toc6510112)

[Table: <Customer> Search store 36](#_Toc6510113)

[**2.3.2.2 < Customer >Order (C02)** 36](#_Toc6510114)

[Figure 10: < Customer > Order 36](#_Toc6510115)

[**2.3.2.2.1 < Customer >View store detail** 36](#_Toc6510116)

[Figure 11: < Customer > View store detail 36](#_Toc6510117)

[Table : <Customer> View store 38](#_Toc6510118)

[**2.3.2.2.2 < Customer >Add to cart** 38](#_Toc6510119)

[Figure 12: < Customer > Add to cart 38](#_Toc6510120)

[Table : <Customer> Add to cart 40](#_Toc6510121)

[**2.3.2.2.3 < Customer >Create order** 40](#_Toc6510122)

[Figure 13: < Customer > Create order 40](#_Toc6510123)

[Table : <Customer> Create order 42](#_Toc6510124)

[**2.3.2.2.4 < Customer >View list order** 42](#_Toc6510125)

[Figure 14: < Customer > View list order 42](#_Toc6510126)

[**2.3.2.2.4 < Customer >View list order** 44](#_Toc6510127)

[Figure 14: < Customer > View list order 45](#_Toc6510128)

[**2.3.3.5 <Customer> Edit Profile** 47](#_Toc6510129)

[Figure 18: < Customer > Edit Profile 47](#_Toc6510130)

[**2.3.3.6 <Customer> Rating** 49](#_Toc6510131)

[Figure 18: < Customer > Edit Profile 49](#_Toc6510132)

[**2.3.3**  **<Shipper> Overview Use Case** 51](#_Toc6510133)

[Figure 15: < Shipper > Over view use case 51](#_Toc6510134)

[**2.3.3.1 <Shipper> Take order** 51](#_Toc6510135)

[Figure 16: < Shipper > Take order 51](#_Toc6510136)

[**2.3.3.2 <Shipper> View Order Detail** 53](#_Toc6510137)

[Figure 17: < Shipper > View Order Detail 53](#_Toc6510138)

[**2.3.3.3 <Shipper> Routing** 56](#_Toc6510139)

[Figure 17: < Shipper > Routing 56](#_Toc6510140)

[**2.3.3.4 <Shipper> Check Out** 58](#_Toc6510141)

[Figure 18: < Shipper > Check out 58](#_Toc6510142)

[**2.3.3.5 <Shipper> Cancel Order** 60](#_Toc6510143)

[Figure 18: < Shipper > Cancel Order 60](#_Toc6510144)

[**2.3.3.6 <Shipper> Edit Profile** 62](#_Toc6510145)

[Figure 18: < Shipper > Edit Profile 62](#_Toc6510146)

[Figure 19: < Admin > Over view use case 65](#_Toc6510147)

[**2.3.4.1 <Admin> Manage order** 65](#_Toc6510148)

[**2.3.4.1.1 <Admin> View history order** 65](#_Toc6510149)

[Table : <Admin> View order 68](#_Toc6510150)

[**2.3.4.1.2 <Admin> Approve face for payment** 68](#_Toc6510151)

[**2.3.4.2 <Admin> Manage account(A02)** 70](#_Toc6510152)

[Figure 22 : < Admin> Manage account 70](#_Toc6510153)

[**2.3.4.2.1 <Admin> Deactive user** 71](#_Toc6510154)

[Figure 23 : < Admin> Deactive user 71](#_Toc6510155)

[Table : <Admin> Deactive user 72](#_Toc6510156)

[**2.3.4.2.2 <Admin> Edit profile** 73](#_Toc6510157)

[Figure 24 : < Admin> Edit profile 73](#_Toc6510158)

[Table : <Admin> Edit profile 75](#_Toc6510159)

[**2.3.4.2.3 <Admin> Create user** 75](#_Toc6510160)

[Figure 25 : < Admin> Create user 75](#_Toc6510161)

[Table : <Admin> Create user 77](#_Toc6510162)

[**2.3.4.3 <Admin> Manage store** 78](#_Toc6510163)

[Figure 26 : < Admin> Manage store overview 78](#_Toc6510164)

[**2.3.4.3.1 <Admin> Create store** 78](#_Toc6510165)

[Figure 27 : < Admin> Create store 78](#_Toc6510166)

[**2.3.4.3.2 <Admin> Edit store** 81](#_Toc6510167)

[Figure 28 : < Admin> Edit store 81](#_Toc6510168)

[**2.3.4.4 <Admin> Manage product** 83](#_Toc6510169)

[Figure 26 : < Admin> Manage product overview 83](#_Toc6510170)

[**2.3.4.4.1 <Admin> Create product** 83](#_Toc6510171)

[Figure 27 : < Admin> Create product 83](#_Toc6510172)

[**2.3.4.4.2 <Admin> Edit product** 86](#_Toc6510173)

[Figure 28 : < Admin> Edit product 86](#_Toc6510174)

[**2.3.4.5 <Admin> Manage report** 88](#_Toc6510175)

[Figure 26 : < Admin> Manage report overview 89](#_Toc6510176)

[**2.3.4.5.1 <Admin> View repor** 89](#_Toc6510177)

[Figure 27 : < Admin> View report 89](#_Toc6510178)

[**3.** **Software System Attribute** 94](#_Toc6510179)

[**3.1** **Usability** 94](#_Toc6510183)

[**3.2** **Reliability** 94](#_Toc6510184)

[The information storing on the database is permanent 94](#_Toc6510185)

[**3.3** **Availability** 95](#_Toc6510186)

[**3.4** **Security** 95](#_Toc6510187)

[Only Manager has fully access to the system 95](#_Toc6510188)

[Input data is validated before saving to database. 95](#_Toc6510189)

[**3.5** **Maintainability** 95](#_Toc6510190)

[The system is divided into separated modules for easy maintain. 95](#_Toc6510191)

[**3.6** **Portability** 95](#_Toc6510192)

[Web application can be run on Chrome browser version 42 or later. 95](#_Toc6510193)

[**3.7** **Performance** 95](#_Toc6510194)

[System has successfully test with basic standard 95](#_Toc6510195)

[**4.** **Conceptual Diagram** 95](#_Toc6510196)

[**Data Dictionary** 95](#_Toc6510197)

[D. Report No. 4 Software Design Description 96](#_Toc6510198)

[1. Design Overview 96](#_Toc6510199)

[2. System Architectural Design 97](#_Toc6510200)

[Figure System Architectural Design 97](#_Toc6510201)

[2.1 Web application architecture description 97](#_Toc6510202)

[3. Component Diagram 98](#_Toc6510203)

[Figure: Component Diagram 98](#_Toc6510204)

[Table Component Dictionary 99](#_Toc6510205)

[4. Detail Description 99](#_Toc6510206)

[4.1 Class Diagram 99](#_Toc6510207)

[Figure Class Diagram 99](#_Toc6510208)

[Table Class Dictionary 100](#_Toc6510209)

[4.2 Class Diagram Explanation 100](#_Toc6510210)

[**4.2.1 Payment Type** 100](#_Toc6510211)

[Table Payment Type Attribute 101](#_Toc6510212)

[Table Payment Type Method 101](#_Toc6510213)

[**4.2.2 Payment Information** 101](#_Toc6510214)

[Table Payment Information Attribute 101](#_Toc6510215)

[Table Payment Information Method 102](#_Toc6510216)

[**4.2.3 Account** 102](#_Toc6510217)

[Table Account Attribute 103](#_Toc6510218)

[Table Account Method 103](#_Toc6510219)

[**4.2.4 City** 103](#_Toc6510220)

[Table City Attribute 103](#_Toc6510221)

[Table City Method 104](#_Toc6510222)

[**4.2.5 District** 104](#_Toc6510223)

[Table District Attribute 104](#_Toc6510224)

[Table District Method 104](#_Toc6510225)

[**4.2.6 Order** 104](#_Toc6510226)

[Table Order Attribute 105](#_Toc6510227)

[Table Order Method 106](#_Toc6510228)

[**4.2.7 Order Detail** 106](#_Toc6510229)

[Table Order Detail Attribute 106](#_Toc6510230)

[Table Order Detail Method 106](#_Toc6510231)

[**4.2.8 Price Level** 106](#_Toc6510232)

[Table Price Level Attribute 107](#_Toc6510233)

[Table Price Level Method 107](#_Toc6510234)

[**4.2.9 Product** 107](#_Toc6510235)

[Table Product Attribute 108](#_Toc6510236)

[Table Product Method 108](#_Toc6510237)

[**4.2.10 Rating** 108](#_Toc6510238)

[Table Rating Attribute 108](#_Toc6510239)

[Table Rating Method 109](#_Toc6510240)

[**4.2.11 Role** 109](#_Toc6510241)

[Table Role Attribute 109](#_Toc6510242)

[Table Role Method 109](#_Toc6510243)

[**4.2.12 Schedule** 109](#_Toc6510244)

[Table Schedule Attribute 110](#_Toc6510245)

[Table Schedule Method 110](#_Toc6510246)

[**4.2.13 Shipper** 110](#_Toc6510247)

[Table Shipper Attribute 111](#_Toc6510248)

[Table Shipper Method 111](#_Toc6510249)

[**4.2.14 Source** 111](#_Toc6510250)

[Table Source Attribute 111](#_Toc6510251)

[Table Source Method 111](#_Toc6510252)

[**4.2.15 Store** 111](#_Toc6510253)

[Table Store Attribute 112](#_Toc6510254)

[Table Store Method 113](#_Toc6510255)

[**4.2.16 Weekday** 113](#_Toc6510256)

[Table Weekday Attribute 113](#_Toc6510257)

[Table Weekday Method 113](#_Toc6510258)

[4.3 Interaction Diagram 113](#_Toc6510259)

[**4.3.1 <User>** 113](#_Toc6510260)

[**4.3.2 <Shipper>** 113](#_Toc6510261)

[**4.3.3 <Admin>** 113](#_Toc6510262)

[**4.3.3.1 <Admin> View list of Admins** 113](#_Toc6510263)

[Figure : < Admin> View list of Admins 113](#_Toc6510264)

[**4.3.3.2 <Admin> Search Admins** 114](#_Toc6510265)

[Figure: < Admin> Search Admins 114](#_Toc6510266)

[**4.3.3.3 <Admin> Add new Admins** 114](#_Toc6510267)

[Figure: < Admin> Add new Admins 114](#_Toc6510268)

[**4.3.3.4 <Admin> View detail of Admins** 114](#_Toc6510269)

[Figure: < Admin> View detail of Admins 114](#_Toc6510270)

[**4.3.3.5 <Admin> Edit Admins** 115](#_Toc6510271)

[Figure: < Admin> Edit Admins 115](#_Toc6510272)

[**4.3.3.6 <Admin> View list of Customers** 115](#_Toc6510273)

[Figure: < Admin> View list of Customers 115](#_Toc6510274)

[**4.3.3.7 <Admin> Search Customer** 116](#_Toc6510275)

[Figure: < Admin> Search Customer 116](#_Toc6510276)

[**4.3.3.8 <Admin> View Detail of Customer** 116](#_Toc6510277)

[Figure: < Admin> View Detail of Customer 116](#_Toc6510278)

[**4.3.3.9 <Admin> View list of Shipper** 117](#_Toc6510279)

[Figure: < Admin> View Detail of Shipper 117](#_Toc6510280)

[**4.3.3.10 <Admin> Search Shipper** 117](#_Toc6510281)

[Figure: < Admin> Search Shipper 117](#_Toc6510282)

[**4.3.3.11 <Admin> Add new Shipper** 117](#_Toc6510283)

[Figure: < Admin> Add new Shipper 118](#_Toc6510284)

[**4.3.3.12 <Admin> View detail of Shipper** 118](#_Toc6510285)

[Figure: < Admin> View Detail of Shipper 118](#_Toc6510286)

[**4.3.3.13 <Admin> Edit Shipper** 118](#_Toc6510287)

[Figure: < Admin> Edit Shipper 118](#_Toc6510288)

[**4.3.3.14 <Admin> View list of Store** 118](#_Toc6510289)

[Figure: < Admin> View List of Store 119](#_Toc6510290)

[**4.3.3.15 <Admin> Search Store** 119](#_Toc6510291)

[Figure: < Admin> Search Store 119](#_Toc6510292)

[**4.3.3.16 <Admin> Add new Store** 119](#_Toc6510293)

[Figure: < Admin> Add new Store 119](#_Toc6510294)

[**4.3.3.17 <Admin> View detail of Store** 120](#_Toc6510295)

[Figure: < Admin> View Detail of Store 120](#_Toc6510296)

[**4.3.3.18 <Admin> Edit Store** 120](#_Toc6510297)

[Figure: < Admin> Edit Store 120](#_Toc6510298)

[**4.3.3.19 <Admin> View list of Product** 120](#_Toc6510299)

[Figure: < Admin> View List of Product 120](#_Toc6510300)

[**4.3.3.20 <Admin> Search Product** 122](#_Toc6510301)

[Figure: < Admin> Search Product 122](#_Toc6510302)

[**4.3.3.21 <Admin> Add new Product** 122](#_Toc6510303)

[Figure: < Admin> Add new Product 122](#_Toc6510304)

[**4.3.3.22 <Admin> View detail of Product** 122](#_Toc6510305)

[Figure: < Admin> View Detail of Product 123](#_Toc6510306)

[**4.3.3.23 <Admin> Edit Product** 124](#_Toc6510307)

[Figure: < Admin> Edit Product 124](#_Toc6510308)

[**4.3.3.24 <Admin> View List of Order** 124](#_Toc6510309)

[Figure: < Admin> View List of Order 124](#_Toc6510310)

[**4.3.3.25 <Admin> View Order** 124](#_Toc6510311)

[Figure: < Admin> View Order 124](#_Toc6510312)

[**4.3.3.26 <Admin> Cancel Order** 125](#_Toc6510313)

[Figure: < Admin> Cancel Order 125](#_Toc6510314)

[**4.3.3.27 <Admin> View Report** 125](#_Toc6510315)

[Figure: < Admin> View Report 125](#_Toc6510316)

[5. Interface 126](#_Toc6510317)

[5.1 Mobile Application 126](#_Toc6510318)

[5.2 Web Application 131](#_Toc6510319)

[6. Database Design 140](#_Toc6510320)

[6.1 Entity Relationship Diagram (ERD) 140](#_Toc6510321)

[Figure Entity Relation Diagram 140](#_Toc6510322)

[6.2 Data Dictionary 140](#_Toc6510323)

[Table Data Dictionary 141](#_Toc6510324)

[7. Algorithms 141](#_Toc6510325)

[E. Report No. 5 System Implementation & Test 141](#_Toc6510326)

[1. Introduction 141](#_Toc6510327)

[1.1 Overview 141](#_Toc6510328)

[1.2 Test Approach 142](#_Toc6510329)

[2. Database Relationship Diagram 142](#_Toc6510330)

[2.1 Physical Diagram 142](#_Toc6510331)

[Figure Physical Diagram 142](#_Toc6510332)

[2.2 Data Dictionary 142](#_Toc6510333)

[Figure Data Dictionary 143](#_Toc6510334)

[Table Payment Type Attribute 144](#_Toc6510335)

[Table Payment Information Attribute 144](#_Toc6510336)

[Table City Attribute 146](#_Toc6510337)

[Table District Attribute 146](#_Toc6510338)

[Table Order Detail Attribute 148](#_Toc6510339)

[Table Price Level Attribute 148](#_Toc6510340)

[Table Product Attribute 149](#_Toc6510341)

[Table Rating Attribute 149](#_Toc6510342)

[Table Role Attribute 149](#_Toc6510343)

[Table Shipper Attribute 150](#_Toc6510344)

[Table Source Attribute 151](#_Toc6510345)

[Table Store Attribute 152](#_Toc6510346)

[3. Performance Measures 152](#_Toc6510347)

[3.1 Features to be tested 152](#_Toc6510348)

[3.2 Features not to be tested 152](#_Toc6510349)

[4. Test Plan 152](#_Toc6510350)

[5. System Testing Test Case 152](#_Toc6510351)

[Figure Admin Core Flow 152](#_Toc6510352)

[Figure Shipper Core Flow 153](#_Toc6510353)

[Figure Customer Core Flow 153](#_Toc6510354)

[5.1 Admin Test Case 153](#_Toc6510355)

[**5.1.1 Report Event** 153](#_Toc6510356)

# Introduction

## Project Information

* Project name:  **Face Detection For Delivery Payment**
* Project Code: FDDP.
* Product Type: **Mobile Application.**
* Start Date: **January 05, 2019.**
* End Date: **--**

## Introduction

Nowadays, In the 4.0 generation we see the rising of millions ecommerce application, we can easily buy and sell something online, following that many kinds of payment solution have been created, such as online payment, COD payment, … If the application has a good payment solution, which help customers easier, safer and more interesting in paying for their product, then that business will grow up a lot faster.

In this document, we introduce a new payment solution for customer: a mobile application called FacePayment Solution (FPS). The FPS application will be a useful assistance to customer when they want to pay for their product through their face recognize. This document also describes our working process in 4 months including our perspective on the system, component designs and detailed core workflows.

## Current Situation

Vietnamese people are more and more shopping through social networks at the rate of 70% in 2018 compared to 66% of the same period last year. Research by Q & Me shows that shopping on social networks offers more fun while e-commerce floors offer better prices and faster delivery times.

According to Vietnam e-commerce report in 2018, which has been announced by market research company Q & Me on November 28, cash is still the main payment method of Vietnamese people when shopping online, with 80% of people asked which form of payment they usually use on delivery choose COD. Forms such as payment cards, e-wallets, and transfers only account for 8%, 6% and 5% respectively.

## Problem Definition

Below are the disadvantages of the current payment solutions:

* In COD payment, when shippers deliver products to customers, there might be potential issues:
* Shippers might not return cash to the company.
* Shippers receive wrong amount of oney.
* Customers might be fake.
* Payment cards, e-wallets, and transfers account for a smaller percentage, only 20%.
* Policies to ensure online shopping and protect buyer rights are still very lax, cases of customers being scammed, paid and not delivered, delivered wrong goods, fake goods, poor quality goods, damaged during transportation.
* Consumers still have not enough time to adapt and change habits payment online.
* The shop owner is not active, only 7.6% of the website integrates online payment facilities, the remaining 92.4% are not integrated for many reasons.

Above payment problems may reduce the effectiveness of product sale, business’s trademark.

## Proposed Solution

Build a Mobile application to support payment when shippers deliver products. It helps create new payment solution through face recognize function. The following workflow and features should be implemented:

### **5.1 Feature functions**

* + Create booking and receive booking system for customer and shipper.
  + Using Face recognize method to pay for order.

### **Advantages and disadvantages**

The advantages and disadvantages of the proposed solution:

* **Advantages**:
* Overcome inherent weaknesses of traditional COD form…
* Taking advantage of COD, Customers can view products before paying, reinforcing trust.
* Quick and convenient payment with relatively high reliability.
* **Disadvantages**:
* There may be problems identifying twins.

## Functional Requirements

Function requirements of the system are listed as below:

**6.1 Customer Application:**

+ Registration.

+ Login.

+ Buy products, book shipper.

+ Estimate: Location, time, distance and money customer have to pay.

+ Face detection payment.

+ Check driver’s information.

+ Cancel booking.

**6.2 Driver Application:**

+ Login.

+ Receive customer booking.

+ Routing for driver.

+ Target define extra earning (optional).

+ Cancel waiting for order.

**6.3 Web Administration Application**

+ Approve face for payment method.

+ Reports.

+ Driver and User manager.

+ Orders manager.

+ Store manager.

## Role & Responsibility

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Full Name** | **Role** | **Position** | **Contact** |
| 1 | Nguyễn Huy Hùng | Project Manager | Supervisor | hungnh@fpt.edu.vn |
| 2 | Nguyễn Văn Hiếu | Developer | Leader | hieunvse61961@fpt.edu.vn |
| 3 | Đinh Phú Thắng | Developer | Member | thangdpse62528@fpt.edu.vn |
| 4 | Trần Trọng Nghĩa | Developer | Member | nghiattse62278@fpt.edu.vn |
| 5 | Nguyễn Hữu Lâm | Developer | Member | lamnhse61234@fpt.edu.vn |

### **Table 1: Roles and Responsibilities**

1. **Software Project Management**
2. **Problem Definition**
   1. **Name of this Capstone Project**

* Official name: Face Detection For Delivery Payment.
* Vietnamese name: Hệ Thống Hỗ Trợ Thanh Toán Thông Qua Nhận Diện Khuôn Mặt Khi Giao Hàng.
* Abbreviation: FDDP.
  1. **Problem Abstract**

Face detection for delivery payment is a system that aims at specifying customer face then use it for paying their order.

Detecting users’ face requires knowledge about AI Machine Learning. For detecting, our system needs customers to provide their face which will use for training AI. Finally, when paying for the order, the user will be taken their face again by the shipper for AI verify with their face which provided.

* 1. **Project Overview**
     1. **Current Situation**
* Strengths:

**+** **Knowledge on software development:** The whole team has experience in software development. All members are familiar with the same server-side programming language**.**

* Weakness:

**+** **Lack of Deep Learning knowledge**: The application using human face detection technology requires Deep Leaning. However, none of the team members has learned Deep Learning or any related techniques. Thus, the research is extremely important and necessary.

**+ Lack of Ionic Framework knowledge:** The application using 2 apps for customer and shipper on mobile device. We are using Ionic framework for apply on multiple platform (Android and IOS). However, none of us has learned and worked with this framework.

**+ Lack of Business knowledge:** The application applied for shipping business. However, none of team members has worked in this business. So we have to research about this business and try our best to done it well. However, maybe some case of business we couldn’t handle.

**+ Lack of support from a payment company:** Our application has to build process of payment to improve it with face recognize. So that, we have to have the permission to access customer’s payment information and the API for approve the payment. However, we have no support for that methods except open source from Paypal. We got many troubles with this problem which we’ll discuss in the another report in this document.

- Opportunities:

* **Implementing Deep Learning not necessary:** Just understanding the mechanism is enough. In addition, there are a lot of libraries to support third-party to work with Deep Learning easily.

* + 1. **The Proposed System**

We create 2 applications in client side for shipper and customer, 1 web application for admin to manage orders, accounts, stores and products to apply a small business. So that we can apply function for payment over detecting human face.

For detecting human face, as mentioned above, we decide to use TensorFlow library as third-party which supported in Deep Learning. This library responses to face points by coordinates. Those points will be assigned to multiple image frames posted by customer as when they assign to system.

After that, the system will compare coordinates on images to find the matched and give result in checkout order, we decide to use Inception Resnet V1 to resolve it. This is the best feasibility solution that our team can perform.

We decide to develop the system with functions:

* + - 1. **Website**
* **For admin (manager)**
* Admin can manage account:
* Admin (Manager)
* View list of admin.
* Search admin with all field.
* View data of admin.
* User (Customer)
* View list of users.
* Search user with all field.
* View data of user.
* Shipper (Staff)
* View list of shippers.
* Search shipper with all field.
* View data of shipper.
* Add new shipper.
* Deactivate shipper.
* Admin can manage order:
* View order list by day.
* Search order with all field.
* Edit order (show who made the actions).
* Admin can manage store:
* View list store partners in the system.
* Search store with all field.
* View data of store.
* Add new data of store.
* Deactivate store.
* Admin can manage product:
* View list of products of store.
* Search product with all field.
* View data of product.
* Add new product depend on store.
* Admin can manage feedback:
* Can view feedback from users.
* Can make feedback for shipper.
  + - 1. **Mobile Application**
* **For driver (staff)**
* Shipper can take orders of customer.
* Shipper can edit their password.
* Shipper can view their profit.
* Shipper can view their routing on google map.
* **For users (customer):**
* User can register new account with their mobile phone.
* User can log in or log out system of mobile phone.
* User can choose the food or item what they like to add to their cart.
* User can edit some of their account’s information.
* User can feedback shipper.
* User can make paying online with face detection.
* User can see list of order they already made.
  + 1. **Boundaries of the System**

**The system can:**

* Allow Manager to manage user accounts .
* Allow Manager to manage stores and products of store.
* Allow Manager to approve face for payment in case shipper can’t use this function.
* Allow Manager to see system report.
* Allow Shipper to receive orders from customer.
* Allow Shipper to see their profile and profit.
* Allow Shipper to view history.
* Allow Shipper verify customer order by face recognize.
* Allow Customer to create new accounts.
* Allow Customer to create order.
* Allow Customer to view history orders.
* Allow Customer to edit their profile
* Allow Customer to edit their receive member face.

The system is not a complete business, it’s just a model to apply payment over face recognize so it can be missing some field in business.

* + 1. **Future Plans**
* Check out payment for shipper.
* Get approved from Paypal to call API accept order.
  + 1. **Development Environment**
       1. **Hardware requirements**

**For Server**

|  |  |  |
| --- | --- | --- |
| **Windows** | **Minimum Requirement** | **Recommended** |
| **Internet Connection** | Cable, Wi-Fi (6 Mbps) | Cable, Wi-Fi (16 Mbps) |
| **Operating System** | Windows 7 or above | Windows 7 or above |
| **Computer Processor** | Core i7 (5th) | Core i7 (7th) |
| **Computer Memory** | 8GB RAM | 16GB RAM |

**For Computer**

|  |  |  |
| --- | --- | --- |
| **Windows** | **Minimum Requirement** | **Recommended** |
| **Internet Connection** | Cable, Wi-Fi (6 Mbps) | Cable, Wi-Fi (16 Mbps) |
| **Operating System** | Windows 7 or above | Windows 7 or above |
| **Computer Processor** | Core i3 (5th). | Core i5 (5th) |
| **Computer Memory** | 4GB RAM | 8GB RAM |

**For Tablet/Mobile**

|  |  |  |
| --- | --- | --- |
| **Windows** | **Minimum Requirement** | **Recommended** |
| **Internet Connection** | Cable, Wi-Fi (6 Mbps) | Cable, Wi-Fi (16 Mbps) |
| **Operating System** | Android 6 | Android 8 |
| **Computer Processor** | Snapdragon 710. | Snapdragon 840. |
| **Computer Memory** | 4GB RAM | 8GB RAM |

* + - 1. **Software requirements**

**For Web**

|  |  |  |
| --- | --- | --- |
| **Software** | **Name / Version** | **Description** |
| **Operating system** | MacOS, windows 7 or above, linux, ….. | Support for mutiple Systems. |
| **Environment** | Java 8 platform | Install java platform jdk8 |
| **Modeling tool** |  |  |
| **IDE** | IntelliJ Idea Ultimate | Jetbrain IDE |
| **DBMS** | Mysql Database | Mysql database |
| **Source control** | Github | Github.com |
| **Web browser** | Chrome, Firefox, Safari, Edge | Support for multiple platform. |

**For Mobile Application**

|  |  |  |
| --- | --- | --- |
| **Software** | **Name / Version** | **Description** |
| **Operating system** | Android OS 6.0 or above, IOS 10 or above. | Support for both Android and IOS. |
| **Environment** | Android or IOS environment. | Mobile device using Android or IOS system. |
| **Modeling tool** | Android, IOS or Chrome browser. | Android or IOS device . |
| **IDE** | Visual Code | Visual Code IDE. |
| **DBMS** | Local Storage | Android local storage. |
| **Source control** | Github | Github.com |

1. **Project Organization**

## 2.1 Software Process Model

This project is developed using Scrum model – part of an agile framework for Software development project. Our team choose Scrum model because of the following reasons:

* Pose Detection is a very new issue with us. We need to focus on researching about deep learning and experiment the most suitable deep learning library with our abilities.
* After detecting human pose, we need to keep researching about algorithm which help us compare the matching percentage between two poses.

For those reasons we choose Scrum model for our process.

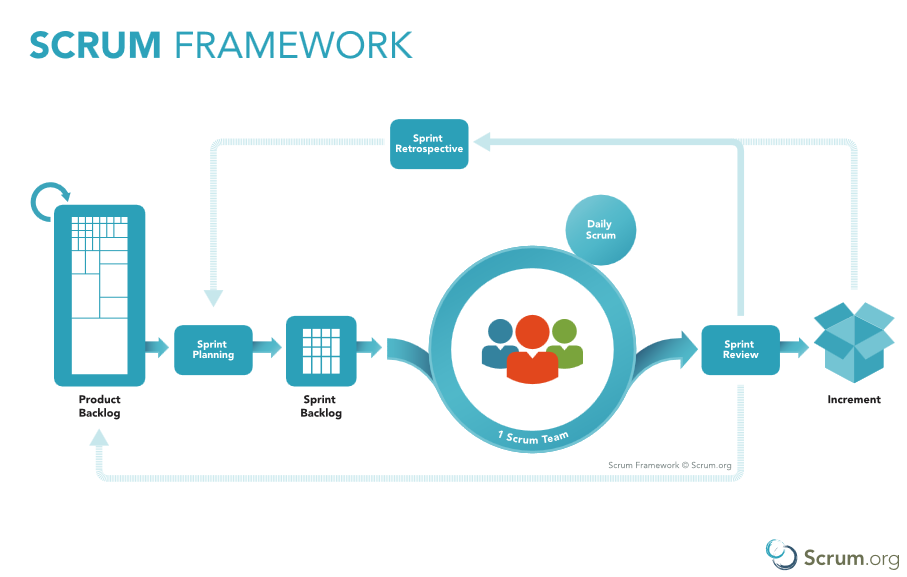


Figure 1: Scrum Framework

(<https://www.scrum.org/resources/what-is-scrum>)

## 2.2 Roles and Responsibilities

|  |  |  |  |
| --- | --- | --- | --- |
| **No** | **Full name** | **Role in Group** | **Responsibilities** |
| **1** | Nguyễn Huy Hùng | Project manager | • Specify user requirement  • Control the development process  • Give out technique and business analysis support |
| **2** | Nguyễn Văn Hiếu | Team Leader, BA, DEV, Tester | • Managing process  • Designing database  • Clarifying requirements  • Prepare documents  • GUI Design  • Create test plan  • Coding  • Testing |
| **3** | Đinh Phú Thắng | Developer/Tester | • Coding  • Testing |
| **4** | Trần Trọng Nghĩa | Developer/Tester | • Coding  • Testing |
| **5** | Nguyễn Hữu Lâm | Developer/Tester | • Coding  • Testing |

## 2.3 Tools and Techniques

|  |  |  |
| --- | --- | --- |
|  | **Techniques** | **Tools** |
| **Developing tool** | Front-end | HTML 5, CSS 3, JavaScript, jQuery 3.1, Bootstrap 4.1 |
| Back-end | Spring Boot MVC Framwork, JPA, Tensorflow, DLib, OpenCV. |
| **Web Servers** | Apache Tomcat | |
| **Mobile** | Ionic Framework | |
| **Database Management System** | Microsoft MySQL server. | |

1. **Project Management Plan**

### **3.1. Sprint Backlog**

Sprint Backlog can be found [here](https://drive.google.com/open?id=1InYqyDqNsp_rIDQsVx8tM0BOcU0HbvMi)

### **3.2. Product Backlog**

Product Backlog can be found [here](https://drive.google.com/open?id=1Kk6lrjphJiMvoETUEk27QsY8z6ZabXZR)

### **3.3. All Meeting Minutes**

Meeting Minutes can be found [here](https://drive.google.com/open?id=1b3XmAZa-3p9I9eWR6zpcKpfYfvhmAft3)

1. **Coding Convention**

* **Naming Convention**:
* Variable names should be short yet meaningful. The choice of a variable name should be designed to indicate to the casual observer the intent of its use.
* Methods should be verbs, in mixed case with the first letter lowercase, with the first letter of each internal word capitalized.
* **Indentation**:
* One declaration per line is recommended since it encourages commenting. In absolutely no case should variables and functions be declared on same line.
* Do not put different types on the same line.
* **Declarations Convention:**
* One declaration per line is recommended since it encourages commenting.
* Using Java Code Convention from:

<http://www.oracle.com/technetwork/java/codeconvtoc-136057.html>

# Software Requirement Specification

1. **User Requirement Specification**
   1. **Guest Requirement**

Guest is a person who doesn’t have access to the system. Guest can use some functions in the system. To use all functions, guest must login. These are some functions guest can use:

* Register.
* Login.
  1. **Customer Requirement**
* View stores list.
* View products of stores.
* Add to cart.
* Make a payment for orders.
* Feedback.
* Cancel orders.
* View history orders.
* Edit profile.
* Add Customer receiver.
* Log out.
  1. **Admin Requirement**
* Manage users
  + - * + Search Users.
        + View detail of users.
        + Send email to users.
* Manage orders
  + - * + Search Orders.
        + View detail of orders.
        + Approve payment for orders.
        + Edit orders.
* Manage shipper
  + - * + Search shipper.
        + View detail of user.
        + Edit-Update detail of user.
* Manage stores
  + - * + Search stores.
        + View detail of stores.
        + Add new stores.
        + Edit stores.
* Manage products
  + - * + Search products.
        + View detail of products.
        + Add new products.
        + Edit products.
* Log out
  1. **Shipper Requirement**
* Receive orders.
* View detail of orders.
* View Shipper profile.
* Edit profile.
* View profit.
* View orders history.
* View routing for shipper.
* Verify Customer face for payment.

1. **System Requirement Specification**

## 2.1 External Interface Requirement

### **2.1.1 User Interface**

* + - GUI should be simple, clear, intuitive, and reminiscent.
    - The interface design is an iterate process includes: design, sketching, prototyping, user assessment.
    - The User Interface uses language is English.

### **2.1.2 Hardware Interface**

* Desktop PC.
* Mobile Device.

### **2.1.3 Software Interface**

* At least android 6.0 or IOS 10 device.
* Web application: work with browsers Firefox (v52 or higher), Chromes (v28 or higher), Internet Explorer (v10 or above) or with any web browser that supports HTML5 & CSS3.

### **2.1.4 Communication Protocol**

* Use HTTP protocol 1.1 for communication between:

Web application and web server.

## 2.2 System Overview Use Case

### Figure 2: System Overview Use Case

## 2.3 List of Use Case

### **2.3.1 <Guest> Overview Use Case**

### **Figure 3: <Guest> Registration Use Case**

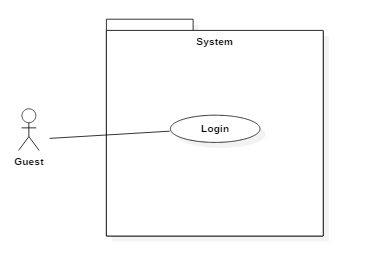
### **2.3.1.1 <Guest> Register Account**

### **Figure 4: <Guest> Registration Use Case**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **USE CASE – UC\_G.01** | | | | | | |
| **Use Case No.** | | UC\_G.01 | | **Use Case Version** | | 1.0 |
| **Use Case Name** | | Register account | | | | |
| **Author** | | ThangDP | | | | |
| **Date** | | 18/1/2019 | | **Priority** | High | |
| **Actor:**   * Guest   **Summary:**   * This use case allows guest to register new account for using system.   **Goal:**   * Guest have new account to use system.   **Triggers:**   * Guest send register command on mobile application.   **Preconditions:**   * N/A   **Post Conditions:**   * Success: navigator to login screens for login * Fail: show error message   **Main Success Scenario:** | | | | | | |
| **Step** | **Actor Action** | | **System Response** | | | |
| 1 | Guest press “Sign Up” link | | System requires identity information from Guest:   * Mobile phone: (+84) number text input, required, length 10-11. * Password: free text input, required, length 8-20. * Email: free text input, required, length 3-250. * Birthdate: date time input, required. * Paypal username: free text input, required, length 3-250. * Paypal password: free text input, required, length 8-20. * Sign Up: button, required. * Take Picture: button, required. | | | |
| 2 | Guest inputs information | |  | | | |
| 3 | Guest press “Sign Up” button. | | System show message  “Notification.  We want to add your face for payment. Do you want to continue sign up?” | | | |
| 4 | Guest press “Add” button. | | System change to camera screen to shot face of customer. | | | |
| 5 | Take Photos | |  | | | |
| 6 | Guest send command to register account for system. | | System validate information, display contract details and requires for confirmation.  [Exception 1] | | | |
| **Alternative Scenario**: N/A | | | | | | |
| **Exceptions**: | | | | | | |
| **Step** | **Actor Action** | | **System Response** | | | |
| 1 | Any field in form is null | | System return message “Form cannot null” | | | |
| **Relationships:** N/A  **Business Rules:**   * Password are encrypted before being sent to database. * If information is invalid, system sends message to inform user and allows user to edit input information * If all information is valid, system checks if phone number has already existed in system or not. * If phone number is duplicated in the system, system sends message to inform user change phone number | | | | | | |

### **Table 2: <Guest> Registration Use Case**

**2.3.1.2 <Guest> Log in**



### **Figure 5: < Guest > Log in**

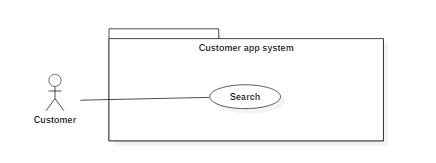
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **USE CASE – UC\_G.02** | | | | | | |
| **Use Case No.** | | UC\_G.02 | | **Use Case Version** | | 1.0 |
| **Use Case Name** | | Log in | | | | |
| **Author** | | ThangDP | | | | |
| **Date** | | 23/1/2019 | | **Priority** | High | |
| **Actor:**   * Guest   **Summary:**   * This use case allows guest to log in.   **Goal:**   * Guest log in.   **Triggers:**   * Guest send log in command on mobile application (Customer, Shipper) or web application (Admin).   **Preconditions:**   * N/A   **Post Conditions:**   * Success: Guest log in to system. * Fail: Show error message.   **Main Success Scenario:** | | | | | | |
| **Step** | **Actor Action** | | **System Response** | | | |
| 1 | Guest goes to login view. | | System requires identity information from Guest:   * Mobile Phone: number text input, required, length 10-11. * Password: free text input, required, length 8-20. * Login: button, required.   Sign Up: link to register page, required. | | | |
| **Alternative Scenario**: N/A | | | | | | |
| **Exceptions**: | | | | | | |
| **Step** | **Actor Action** | | **System Response** | | | |
| 1 | Username and password are not matching or cannot found in system | | System returns message ‘Wrong username or password’ | | | |
| **Relationships:** N/A  **Business Rules:**   * Authorized user inputs login information:   + Phone number: number  + Password: text input   * Then send login command * System will authenticate, let user login to system and authorize to redirect according to their specific role | | | | | | |

### **Table 3: < Guest > Log in**

### **2.3.2 <Customer> Overview Use Case**

### **Figure 6: < Customer > Over view use case**

### **2.3.2.1 < Customer > Search store**

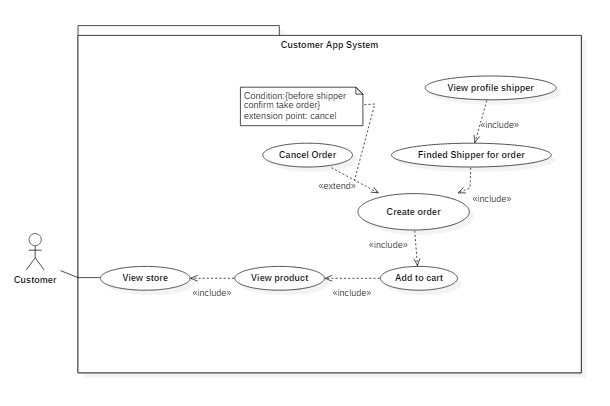


### **Figure 7: < Customer > Search store**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **USE CASE – UC\_C.01** | | | | | | |
| **Use Case No.** | | UC\_C.01 | | **Use Case Version** | | 1.0 |
| **Use Case Name** | | Search store | | | | |
| **Author** | | ThangDP | | | | |
| **Date** | | 23/1/2019 | | **Priority** | Normal | |
| **Actor:**   * Customer   **Summary:**   * This use case allows customers search store   **Goal:**   * Show the store which customers search   **Triggers:**   * Customer send search command   **Preconditions:**   * Account have role “member”   **Post Conditions:**   * Success: Show store on screen * Fail: Send message not found   **Main Success Scenario:** | | | | | | |
| **Step** | **Actor Action** | | **System Response** | | | |
| 1 | Customer go to “Home” page. | | Display on screen:   * Search: search area, required. * Product Suggest: * Store Name + Store Address: text area, required. * Product’s product: image, required. * Price: double, required. * Store suggest: * Store Name + Store Address: text area, required. | | | |
| 2 | Customers send search command | | System search store which customer send.  [Alternative 1] | | | |
| **Alternative Scenario:** | | | | | | |
| **Step** | **Actor Action** | | **System Response** | | | |
| 1 | Store not found | | Show message not found for customer | | | |
| **Exceptions**: N/A | | | | | | |
| **Relationships:** N/A  **Business Rules:** Customer search for store they need, If don’t have any store like they search, System will return no result and client will so message “not found store” for customer. | | | | | | |

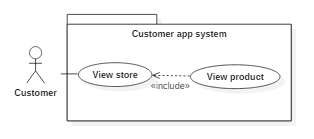
### **Table 4: <Customer> Search store**

### **2.3.2.2 < Customer >Order (C02)**



### **Figure 8: < Customer > Order**

### **2.3.2.2.1 < Customer >View store detail**

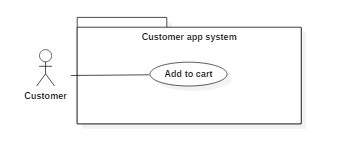


### **Figure 9: < Customer > View store detail**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **USE CASE – UC\_C.02A** | | | | | | |
| **Use Case No.** | | UC\_C.02A | | **Use Case Version** | | 1.0 |
| **Use Case Name** | | View store detail | | | | |
| **Author** | | ThangDP | | | | |
| **Date** | | 23/1/2019 | | **Priority** | High | |
| **Actor:**   * Customer   **Summary:**   * Customer send command to view store   **Goal:**   * Show detail store for customer   **Triggers:**   * Customer send store id command   **Preconditions:**   * Account have role “member”   **Post Conditions:**   * Success: Show store detail on screen. * Fail: Show loading screen and log exception on command line.   **Main Success Scenario:** | | | | | | |
| **Step** | **Actor Action** | | **System Response** | | | |
| 1 | Customers send view store detail command | | System show store follow store id for customer  System show on screen:   * Store’s Images: images, required. * Store Name – Store Address: text, required. * Open time: text, required. * Product List: * Product’s Name: text, required. * Product’s Price: text, required. * ‘\_’: minus button, required. * ‘+’: plus button, required. * “Order Cart-Total Price”: button, required. | | | |
| **Alternative Scenario:** N/A | | | | | | |
| **Exceptions**: N/A | | | | | | |
| **Relationships:** N/A  **Business Rules:** N/A | | | | | | |

### **Table 5 : <Customer> View store**

### **2.3.2.2.2 < Customer >Add to cart**



### **Figure 10: < Customer > Add to cart**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **USE CASE – UC\_C.02B** | | | | | | |
| **Use Case No.** | | UC\_C.02B | | **Use Case Version** | | 1.0 |
| **Use Case Name** | | Add to cart | | | | |
| **Author** | | ThangDP | | | | |
| **Date** | | 25/1/2019 | | **Priority** | Normal | |
| **Actor:**   * Customer   **Summary:**   * Customer send command add product to cart   **Goal:**   * Add product to cart * Show cart for customer   **Triggers:**   * Customer send product id command for add to cart   **Preconditions:**   * Account have role “member”   **Post Conditions:**   * Success: Show cart on screen * Fail:   **Main Success Scenario:** | | | | | | |
| **Step** | **Actor Action** | | **System Response** | | | |
| 1 | Customers send product id for adding to cart | | System add product to cart follow product id for customer.  System will show on screen:   * Customer Address: text, required. * Store Address: text, required. * Product’s Photo: image, required. * Product’s Quantity: int, required. * Product’s Price: double, required. * Subtotal: double, required. * Delivery Fee: double, required. * Total: double, required. * Shipper Earn: double, required. * Custome Information area: * Full Name: text, required. * Phone Number: text, required. | | | |
| **Alternative Scenario:** N/A | | | | | | |
| **Exceptions**: N/A | | | | | | |
| **Relationships:** N/A  **Business Rules:** N/A | | | | | | |

### **Table 6 : <Customer> Add to cart**

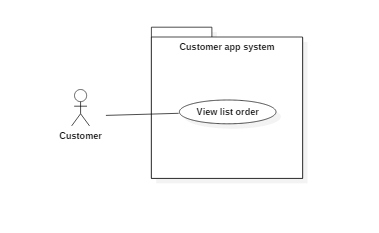
### **2.3.2.2.3 < Customer >Create order**

### **Figure 11: < Customer > Create order**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **USE CASE – UC\_C.02C** | | | | | | |
| **Use Case No.** | | UC\_C.02C | | **Use Case Version** | | 1.0 |
| **Use Case Name** | | Create Order | | | | |
| **Author** | | ThangDP | | | | |
| **Date** | | 25/1/2019 | | **Priority** | High | |
| **Actor:**   * Customer   **Summary:**   * This use case is escribes how to create order, cancel it or view shipper info when taken by shipper   **Goal:**   * Create order   **Triggers:**   * Customer send create order command   **Preconditions:**   * Account have role “member”   **Post Conditions:**   * Success: Show popup loading for finding shipper * Fail:   **Main Success Scenario:** | | | | | | |
| **Step** | **Actor Action** | | **System Response** | | | |
| 1 | Customers send product id for adding to cart | | System add product to cart follow product id for customer  [Alternative 1] | | | |
|  |  | |  | | | |
| **Alternative Scenario:** | | | | | | |
| **Step** | **Actor Action** | | **System Response** | | | |
| 1 | Customer cancel order | | System will cancel order and return to cart for customer | | | |
| **Exceptions**: | | | | | | |
| **Step** | **Actor Action** | | **System Response** | | | |
|  |  | |  | | | |
| **Relationships:** N/A  **Business Rules:** - Customer only can cancel order before shipper confirm take order from store | | | | | | |

### **Table 7: <Customer> Create order**

### **2.3.2.2.4 < Customer >View list order**

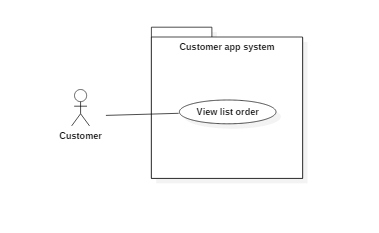


### **Figure 12: < Customer > View list order**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **USE CASE – UC\_C.02D** | | | | | | |
| **Use Case No.** | | UC\_C.02D | | **Use Case Version** | | 1.0 |
| **Use Case Name** | | View list order | | | | |
| **Author** | | ThangDP | | | | |
| **Date** | | 25/1/2019 | | **Priority** | Normal | |
| **Actor:**   * Customer   **Summary:**   * Customer send view list order command   **Goal:**   * Show list of order by customer   **Triggers:**   * Customer send view list order command   **Preconditions:**   * Account have role “member”   **Post Conditions:**   * Success: Show list order of customer * Fail:   **Main Success Scenario:** | | | | | | |
| **Step** | **Actor Action** | | **System Response** | | | |
| 1 | Customer send view list order command | | System show list order on screen:  [Alternative 1]   * Order Code: text, required. * Date Booked: date, required. * Total: double, required. | | | |
| 2 | Customer click link of order list. | | System will show on screen:   * Customer Address: text, required. * Store Address: text, required. * Product’s Photo: image, required. * Product’s Quantity: int, required. * Product’s Price: double, required. * Subtotal: double, required. * Delivery Fee: double, required. * Total: double, required. * Shipper Earn: double, required. * Customer Information area:   Full Name: text, required. | | | |
| **Alternative Scenario:** | | | | | | |
| **Step** | **Actor Action** | | **System Response** | | | |
| 1 | No order in list | | System show “No order is created yet” | | | |
| **Exceptions**: | | | | | | |
| **Step** | **Actor Action** | | **System Response** | | | |
|  |  | |  | | | |
| **Relationships:** N/A  **Business Rules:** N/A | | | | | | |

### **Table 8: <Customer> View list order**

### **2.3.2.2.4 < Customer >View list order**



### **Figure 13: < Customer > View list order**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **USE CASE – UC\_C.02D** | | | | | | |
| **Use Case No.** | | UC\_C.02D | | **Use Case Version** | | 1.0 |
| **Use Case Name** | | View list order | | | | |
| **Author** | | ThangDP | | | | |
| **Date** | | 25/1/2019 | | **Priority** | Normal | |
| **Actor:**   * Customer   **Summary:**   * Customer send view list order command   **Goal:**   * Show list of order by customer   **Triggers:**   * Customer send view list order command   **Preconditions:**   * Account have role “member”   **Post Conditions:**   * Success: Show list order of customer * Fail:   **Main Success Scenario:** | | | | | | |
| **Step** | **Actor Action** | | **System Response** | | | |
| 1 | Customer send view list order command | | System show list order on screen:  [Alternative 1]   * Order Code: text, required. * Date Booked: date, required. * Total: double, required. | | | |
| 2 | Customer click link of order list. | | System will show on screen:   * Customer Address: text, required. * Store Address: text, required. * Product’s Photo: image, required. * Product’s Quantity: int, required. * Product’s Price: double, required. * Subtotal: double, required. * Delivery Fee: double, required. * Total: double, required. * Shipper Earn: double, required. * Custome Information area:   Full Name: text, required. | | | |
| **Alternative Scenario:** | | | | | | |
| **Step** | **Actor Action** | | **System Response** | | | |
| 1 | No order in list | | System show “No order is created yet” | | | |
| **Exceptions**: | | | | | | |
| **Step** | **Actor Action** | | **System Response** | | | |
|  |  | |  | | | |
| **Relationships:** N/A  **Business Rules:** N/A | | | | | | |

### **Table 9: <Customer> View list order**

### **2.3.3.5 <Customer> Edit Profile**

### Figure 18: < Customer > Edit Profile

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **USE CASE – UC\_S.01F** | | | | | | |
| **Use Case No.** | | UC\_S.01F | | **Use Case Version** | | 1.0 |
| **Use Case Name** | | Edit Profile | | | | |
| **Author** | |  | | | | |
| **Date** | | 26/1/2019 | | **Priority** | Normal | |
| **Actor:**   * Customer   **Summary:**   * Customer edit profile.   **Goal:**   * edit profile for Customer.   **Triggers:**   * Customer edit profile success.   **Preconditions:**   * Account have role “Customer”   **Post Conditions:**   * Success: Show edit profile success for shipper * Fail:   **Main Success Scenario:** | | | | | | |
| **Step** | **Actor Action** | | **System Response** | | | |
| 1 | System go to Edit Profile page | | * SAVE: button, required. * Account’s Photo: image, required. * EDIT/ADD AVATAR: button. * Form area: * Phone Number: text, required. * Full name: text, required. * Email: text, required. * Date of birth: date picker, required. * Your member area: * Member’s Photo: images, required. * Edit: button, required. * ADD NEW FACE’S MEMBER: button, required. | | | |
| 2 | Customer send edit profile command | | System will response | | | |
| **Alternative Scenario:** N/A | | | | | | |
| **Exceptions**: N/A | | | | | | |
| **Relationships:** N/A  **Business Rules:** N/A | | | | | | |

### **Table : < Customer > Edit Profile**

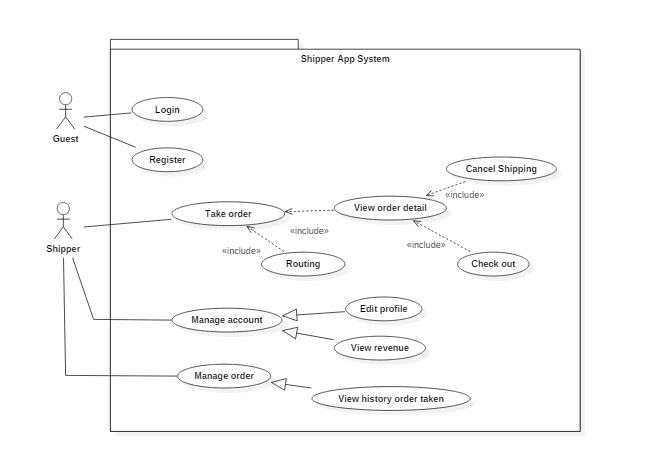
### **2.3.3.6 <Customer> Rating**

### Figure 18: < Customer > Edit Profile

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **USE CASE – UC\_S.01G** | | | | | | |
| **Use Case No.** | | UC\_S.01G | | **Use Case Version** | | 1.0 |
| **Use Case Name** | | Rating | | | | |
| **Author** | |  | | | | |
| **Date** | | 26/1/2019 | | **Priority** | Normal | |
| **Actor:**   * Customer   **Summary:**   * Customer rating order.   **Goal:**   * Rating Order for Customer.   **Triggers:**   * Customer rating order success.   **Preconditions:**   * Account have role “Customer”   **Post Conditions:**   * Success: Show rating order success for customer. * Fail:   **Main Success Scenario:** | | | | | | |
| **Step** | **Actor Action** | | **System Response** | | | |
| 1 | System go to Rating page | | System will show on screen:   * 1-5 start: required. * Submit: button, required. | | | |
| 2 | Customer send edit profile command | | System will response | | | |
| **Alternative Scenario:** N/A | | | | | | |
| **Exceptions**: N/A | | | | | | |
| **Relationships:** N/A  **Business Rules:** N/A | | | | | | |

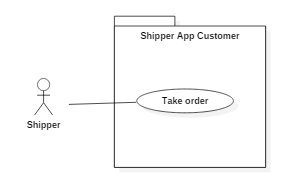
### Table : < Customer > Rating

### **2.3.3 <Shipper> Overview Use Case**



### Figure 15: < Shipper > Over view use case

### **2.3.3.1 <Shipper> Take order**



### Figure 16: < Shipper > Take order

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **USE CASE – UC\_S.01A** | | | | | | |
| **Use Case No.** | | UC\_S.01A | | **Use Case Version** | | 1.0 |
| **Use Case Name** | | Take order | | | | |
| **Author** | |  | | | | |
| **Date** | | 25/1/2019 | | **Priority** | High | |
| **Actor:**   * Shipper   **Summary:**   * Shipper turn on shipper mode   **Goal:**   * Assign order nearest shipper’s location   **Triggers:**   * Shipper send ready ship command   **Preconditions:**   * Account have role “shipper”   **Post Conditions:**   * Success: Assign order for customer * Fail:   **Main Success Scenario:** | | | | | | |
| **Step** | **Actor Action** | | **System Response** | | | |
| 1 | Shipper turn on mode shipper | | System will auto assign order nearest shipper’s loaction (10km radius).  System will show on screen:   * Shipper Mode: toggle button required. * Map: Google Map, required.   [Alternative 1] | | | |
| **Alternative Scenario:** | | | | | | |
| **Step** | **Actor Action** | | **System Response** | | | |
| 1 | Can not find order | | System show “No order is created yet” | | | |
| **Exceptions**: | | | | | | |
| **Step** | **Actor Action** | | **System Response** | | | |
|  |  | |  | | | |
| **Relationships:** N/A  **Business Rules:** N/A | | | | | | |

### **Table : <Shipper> Take order**

### **2.3.3.2 <Shipper> View Order Detail**

### **Figure 17: < Shipper > View Order Detail**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **USE CASE – UC\_S.01B** | | | | | | |
| **Use Case No.** | | UC\_S.01B | | **Use Case Version** | | 1.0 |
| **Use Case Name** | | View order detail | | | | |
| **Author** | |  | | | | |
| **Date** | | 26/1/2019 | | **Priority** | High | |
| **Actor:**   * Shipper   **Summary:**   * Shipper view order detail   **Goal:**   * view order detail for shipper   **Triggers:**   * Shipper send view order detail.   **Preconditions:**   * Account have role “shipper”   **Post Conditions:**   * Success: Show order detail for shipper * Fail:   **Main Success Scenario:** | | | | | | |
| **Step** | **Actor Action** | | **System Response** | | | |
| 1 | Shipper send view order detail command | | System show:   * Customer Address: text, required. * Store Address: text, required. * Product’s Photo: image, required. * Product’s Quantity: int, required. * Product’s Price: double, required. * Subtotal: double, required. * Delivery Fee: double, required. * Total: double, required. * Shipper Earn: double, required. * Customer Information area: * Full Name: text, required. * Phone Number: text, required. * Take Bill: button, required. | | | |
| **Alternative Scenario:** N/A | | | | | | |
| **Exceptions**: N/A | | | | | | |
| **Relationships:** N/A  **Business Rules:** N/A | | | | | | |

### **Table : <Shipper> View Order Detail**

### **2.3.3.3 <Shipper> Routing**

### **Figure 17: < Shipper > Routing**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **USE CASE – UC\_S.01C** | | | | | | |
| **Use Case No.** | | UC\_S.01C | | **Use Case Version** | | 1.0 |
| **Use Case Name** | | Routing | | | | |
| **Author** | |  | | | | |
| **Date** | | 26/1/2019 | | **Priority** | High | |
| **Actor:**   * Shipper   **Summary:**   * Shipper view map to go.   **Goal:**   * view map to go for shipper   **Triggers:**   * Shipper know map.   **Preconditions:**   * Account have role “shipper”   **Post Conditions:**   * Success: Show order detail for shipper * Fail:   **Main Success Scenario:** | | | | | | |
| **Step** | **Actor Action** | | **System Response** | | | |
| 1 | Shipper send view order detail command | | System will show on screen:   * Shipper Mode: toggle button required. * Map: Google Map, required. * Order Code: text, required. * Order detail area: * Customer’s Address: text, required. * Store’s Address: text, required. | | | |
| **Alternative Scenario:** N/A | | | | | | |
| **Exceptions**: N/A | | | | | | |
| **Relationships:** N/A  **Business Rules:** N/A | | | | | | |

### **Table : <Shipper> Routing**

### **2.3.3.4 <Shipper> Check Out**

### **Figure 18: < Shipper > Check out**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **USE CASE – UC\_S.01D** | | | | | | |
| **Use Case No.** | | UC\_S.01D | | **Use Case Version** | | 1.0 |
| **Use Case Name** | | Check out | | | | |
| **Author** | |  | | | | |
| **Date** | | 26/1/2019 | | **Priority** | Normal | |
| **Actor:**   * Shipper   **Summary:**   * Shipper check out order.   **Goal:**   * check out order for shipper   **Triggers:**   * Shipper check out success.   **Preconditions:**   * Account have role “shipper”   **Post Conditions:**   * Success: Show check out for shipper * Fail:   **Main Success Scenario:** | | | | | | |
| **Step** | **Actor Action** | | **System Response** | | | |
| 1 | Shipper send check out command | | System show:   * Customer Address: text, required. * Store Address: text, required. * Product’s Photo: image, required. * Product’s Quantity: int, required. * Product’s Price: double, required. * Subtotal: double, required. * Delivery Fee: double, required. * Total: double, required. * Shipper Earn: double, required. * Customer Information area: * Full Name: text, required. * Phone Number: text, required. * Check out: button, required. | | | |
| **Alternative Scenario:** N/A | | | | | | |
| **Exceptions**: N/A | | | | | | |
| **Relationships:** N/A  **Business Rules:** N/A | | | | | | |

### **Table : <Shipper> Check Out**

### **2.3.3.5 <Shipper> Cancel Order**

### **Figure 18: < Shipper > Cancel Order**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **USE CASE – UC\_S.01E** | | | | | | |
| **Use Case No.** | | UC\_S.01E | | **Use Case Version** | | 1.0 |
| **Use Case Name** | | Cancel Order | | | | |
| **Author** | |  | | | | |
| **Date** | | 26/1/2019 | | **Priority** | Normal | |
| **Actor:**   * Shipper   **Summary:**   * Shipper canel order.   **Goal:**   * cancel order for shipper   **Triggers:**   * Shipper cancel success.   **Preconditions:**   * Account have role “shipper”   **Post Conditions:**   * Success: Show cancel success for shipper * Fail:   **Main Success Scenario:** | | | | | | |
| **Step** | **Actor Action** | | **System Response** | | | |
| 1 | Shipper send cancel order command | | * System will response | | | |
| **Alternative Scenario:** N/A | | | | | | |
| **Exceptions**: N/A | | | | | | |
| **Relationships:** N/A  **Business Rules:** N/A | | | | | | |

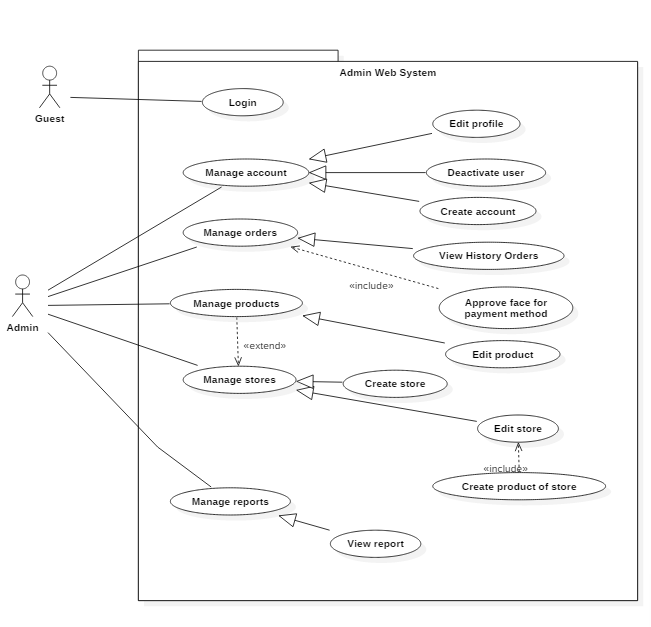
### **Table : <Shipper> Cancel Order**

### **2.3.3.6 <Shipper> Edit Profile**

### Figure 18: < Shipper > Edit Profile

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **USE CASE – UC\_S.01F** | | | | | | |
| **Use Case No.** | | UC\_S.01F | | **Use Case Version** | | 1.0 |
| **Use Case Name** | | Edit Profile | | | | |
| **Author** | |  | | | | |
| **Date** | | 26/1/2019 | | **Priority** | Normal | |
| **Actor:**   * Shipper   **Summary:**   * Shipper edit profile.   **Goal:**   * edit profile for shipper   **Triggers:**   * Shipper edit profile success.   **Preconditions:**   * Account have role “shipper”   **Post Conditions:**   * Success: Show edit profile success for shipper * Fail:   **Main Success Scenario:** | | | | | | |
| **Step** | **Actor Action** | | **System Response** | | | |
| 1 | System go to Edit Profile page | | * SAVE: button, required. * Account’s Photo: image, required. * EDIT/ADD AVATAR: button. * Form area: * Phone Number: text, required. * Full name: text, required. * Email: text, required. * Date of birth: date picker, required. * Your member area: * Member’s Photo: images, required. * Edit: button, required. * ADD NEW FACE’S MEMBER: button, required. | | | |
| 2 | Shipper send edit profile command | | System will response | | | |
| **Alternative Scenario:** N/A | | | | | | |
| **Exceptions**: N/A | | | | | | |
| **Relationships:** N/A  **Business Rules:** N/A | | | | | | |

### **Table : <Shipper> Edit Profile**

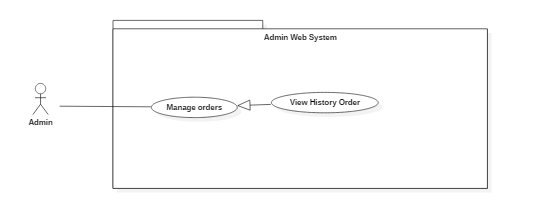
**2.3.4 <Admin> Overview Use Case**

### Figure 19: < Admin > Over view use case

### **2.3.4.1 <Admin> Manage order**

### **Figure 20 : < Admin> Manage Order Overview**

### **2.3.4.1.1 <Admin> View history order**

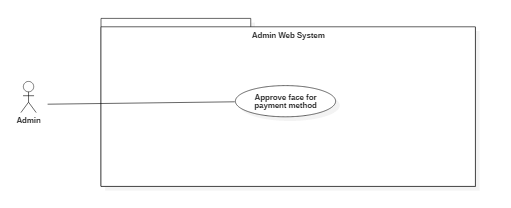


### **Figure 21 : < Admin> View history Order**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **USE CASE – UC\_A.01A** | | | | | | |
| **Use Case No.** | | UC\_A.01A | | **Use Case Version** | | 1.0 |
| **Use Case Name** | | View history order | | | | |
| **Author** | | ThangDP | | | | |
| **Date** | | 28/1/2019 | | **Priority** | Normal | |
| **Actor:**   * Admin   **Summary:**   * Admin view history order.   **Goal:**   * Admin sees list history order.   **Triggers:**   * Admin send view order command.   **Preconditions:**   * User role is “admin”.   **Post Conditions:**   * Success: Show list history order. * Fail:   **Main Success Scenario:** | | | | | | |
| **Step** | **Actor Action** | | **System Response** | | | |
| 1 | Admin send view order command | | System show list history order   * “Search”: text area, required.   Data Table which contains:   * + “Code”: Code of order, text, required.   + “Buyer Phone”: text area, required.   + “Buyer Name”: text, required.   + “Total Price”: integer, required.   + “Status”: text area, required.   The value of status:   1. New: Order is new 2. Assigned: Order is assigned. 3. Bought: Order is bought. 4. Done: Order is done. 5. Cancel: Order is cancelled. | | | |
| 2 | Admin click row of order list to view detail of order. | | Display the view order detail modal on screen:   * “Customer Face”: image area, required. * “Bill”: image area, required. * “Order Code”: text area, required. * “Customer Phone”: text area, required. * “Customer Name”: text area, required. * “Shipper Phone”: text area, required. * “Shipper Name”: text area, required. * “Status”: text area, required. * “Customer Longitude”: text area, required. * “Customer Latitude”: text area, required. * “Store Longitude”: text area, required. * “Store Latitude”: text area, required. * “Store Name”: text area, required. * “Store Address”: text area, required. * “Customer Description”: text area, required. * “Customer Address”: text area, required. * “Receive Time”: text area, required. * “Note”: text area, required.   Data Table:   * + “#”: Code of product, text, required.   + “Product”: text, required.   + “Price”: text, required.   + “Unit”: integer, required.   + “Cost”: double, required.   + “Sub Total”: double, required.   + “Shipping Fee”: double, required.   + “Total”: double, required.   + “Save”: button, required.   + “Close”: button, required. | | | |
| 3 | Admin add photo provide by customer to verify his(her) order | | System will compare with the photo in customer account to verify  [Exception 1] | | | |
| **Alternative Scenario: N/A** | | | | | | |
| **Exceptions**: | | | | | | |
| **Step** | **Actor Action** | | **System Response** | | | |
| 1 | Can not verify customer face | | Show message “Wrong user’s face or try again.” | | | |
| **Relationships:** N/A  **Business Rules:** N/A | | | | | | |

### **Table : <Admin> View order**

### **2.3.4.1.2 <Admin> Approve face for payment**

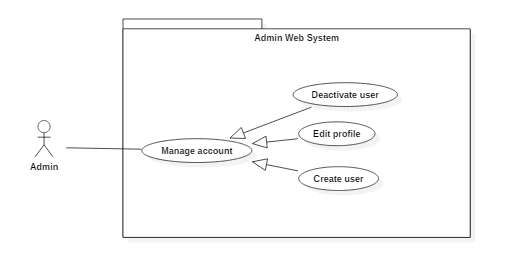


### **Figure 21 : < Admin> Approve face for payment**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **USE CASE – UC\_A.01B** | | | | | | |
| **Use Case No.** | | UC\_A.01B | | **Use Case Version** | | 1.0 |
| **Use Case Name** | | Approve face for payment | | | | |
| **Author** | | ThangDP | | | | |
| **Date** | | 28/1/2019 | | **Priority** | Normal | |
| **Actor:**   * Admin   **Summary:**   * Admin approve customer’s face for payment.   **Goal:**   * Customer’s face is approved.   **Triggers:**   * Admin send approve command.   **Preconditions:**   * User role is “admin”.   **Post Conditions:**   * Success: Show payment success. * Fail:   **Main Success Scenario:** | | | | | | |
| **Step** | **Actor Action** | | **System Response** | | | |
| 1 | Admin add photo provide by customer to verify his(her) order | | System load into screen. | | | |
| 2 | Admin click “Save” button. | | System will compare with the photo in customer account to verify  [Exception 1] | | | |
| **Alternative Scenario: N/A** | | | | | | |
| **Exceptions**: | | | | | | |
| **Step** | **Actor Action** | | **System Response** | | | |
| 1 | Can not verify customer face | | Show message “Wrong user’s face or try again.” | | | |
| **Relationships:** N/A  **Business Rules:** N/A | | | | | | |

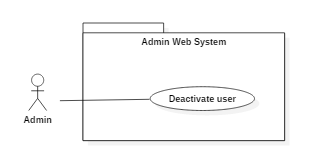
**Table : <Admin> Approve face for payment**

### **2.3.4.2 <Admin> Manage account(A02)**



### **Figure 22 : < Admin> Manage account**

### **2.3.4.2.1 <Admin> Deactive user**

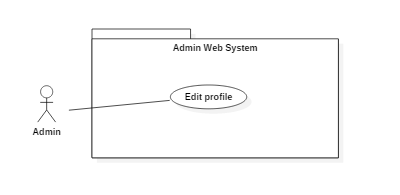


### **Figure 23 : < Admin> Deactive user**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **USE CASE – UC\_A.02A** | | | | | | |
| **Use Case No.** | | UC\_A.02A | | **Use Case Version** | | 1.0 |
| **Use Case Name** | | Deactive user | | | | |
| **Author** | | ThangDP | | | | |
| **Date** | | 30/1/2019 | | **Priority** | Normal | |
| **Actor:**   * Admin   **Summary:**   * Deactive account user   **Goal:**   * Account user is deactived   **Triggers:**   * Admin send deactive account command   **Preconditions:**   * User role is “admin”.   **Post Conditions:**   * Success: Account is deactived * Fail:   **Main Success Scenario:** | | | | | | |
| **Step** | **Actor Action** | | **System Response** | | | |
| 1 | * Admin lick row on list of User. | | * System will show modal: * “Phone”: phone number, text, length 10-11, required. * “Full name”: text area, required. * “Date of Birth”: date time, required. * “Email”: text area, required. * “Status”: select, required. * “Sub”: select, required. * “Note”: text area, required. * “Cancel”: button, required. * “Save”: button, required. | | | |
| 2 | * Admin change Status to “Banned” and click “Save”. | | * System deactive account which specify. | | | |
| **Alternative Scenario:** N/A | | | | | | |
| **Exceptions**: N/A | | | | | | |
| **Relationships:** N/A  **Business Rules:** N/A | | | | | | |

### **Table : <Admin> Deactive user**

### **2.3.4.2.2 <Admin> Edit profile**

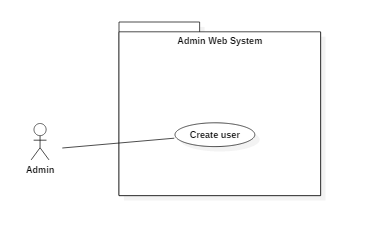


### **Figure 24 : < Admin> Edit profile**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **USE CASE – UC\_A.02B** | | | | | | |
| **Use Case No.** | | UC\_A.02B | | **Use Case Version** | | 1.0 |
| **Use Case Name** | | Edit profile | | | | |
| **Author** | | ThangDP | | | | |
| **Date** | | 30/1/2019 | | **Priority** | Normal | |
| **Actor:**   * Admin   **Summary:**   * Admin will edit profile account which is specified   **Goal:**   * Edit profile account   **Triggers:**   * Admin send request update profile account   **Preconditions:**   * User role is “admin”.   **Post Conditions:**   * Success: Account is updated * Fail:   **Main Success Scenario:** | | | | | | |
| **Step** | **Actor Action** | | **System Response** | | | |
| 1 | Admin lick row on list of Account. | | System will show modal:   * Customer account:   + Full name: string, required  + Email: string, null able  + Note: string, null able   * Shipper account:   + Phone number: number, required, 10-12 characters  + Date of birth: format mm/dd/yyy, required  + Email: format: [abc@gmail.com](mailto:abc@gmail.com), required  + Portrait: image of user account, required  + Nation ID number: number, required  + Issue date: format mm/dd/yyyy,, required  + Bike register ID number: number, required  + Issue date: format mm/dd/yyyy, required   * Admin account:   + Full name: string, required  + Date of birth: format mm/dd/yyyy, required  + Email: format [abc@gmail.com](mailto:abc@gmail.com), required  + Nation ID number: number, required   * + Date of issue: format mm/dd/yyyy, required | | | |
| 2 | Admin change data of customer and click “Save”. | | System will update information profile account  [Exception 1][Exception 2] | | | |
| **Alternative Scenario:** | | | | | | |
| **Step** | **Actor Action** | | **System Response** | | | |
|  |  | |  | | | |
| **Exceptions**: | | | | | | |
| **Step** | **Actor Action** | | **System Response** | | | |
| 1 | Null require field | | System show message “Field is required !” | | | |
| 2 | Not follow the format | | System show message “Field is not follow the format” | | | |
| **Relationships:** N/A  **Business Rules:** N/A | | | | | | |

### **Table : <Admin> Edit profile**

### **2.3.4.2.3 <Admin> Create user**



### **Figure 25 : < Admin> Create user**

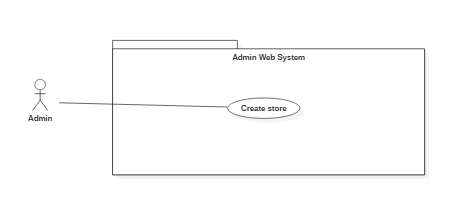
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **USE CASE – UC\_A.02C** | | | | | | |
| **Use Case No.** | | UC\_A.02C | | **Use Case Version** | | 1.0 |
| **Use Case Name** | | Create user | | | | |
| **Author** | | ThangDP | | | | |
| **Date** | | 30/1/2019 | | **Priority** | Normal | |
| **Actor:**   * Admin   **Summary:**   * Create new account   **Goal:**   * Create a new account   **Triggers:**   * Admin send create account command   **Preconditions:**   * User role is “admin”.   **Post Conditions:**   * Success: Account is created * Fail:   **Main Success Scenario:** | | | | | | |
| **Step** | **Actor Action** | | **System Response** | | | |
| 1 | Admin click “Add New” | | * Shipper account:   + Phone number: number, required, 10-12 characters  + Date of birth: format mm/dd/yyy, required  + Email: format: [abc@gmail.com](mailto:abc@gmail.com), required  + Portrait: image of user account, required  + Nation ID number: number, required  + Issue date: format mm/dd/yyyy,, required  + Bike register ID number: number, required  + Issue date: format mm/dd/yyyy, required   * Admin account:   + Full name: string, required  + Date of birth: format mm/dd/yyyy, required  + Email: format [abc@gmail.com](mailto:abc@gmail.com), required  + Nation ID number: number, required  + Date of issue: format mm/dd/yyyy, required | | | |
| 2 | Admin input data and click “Save”. | | System will create new account  [Exception 1] [Exception 2] | | | |
| **Alternative Scenario:** | | | | | | |
| **Step** | **Actor Action** | | **System Response** | | | |
|  |  | |  | | | |
| **Exceptions**: | | | | | | |
| **Step** | **Actor Action** | | **System Response** | | | |
| 1 | Null require field | | System show message “Field is required !” | | | |
| 2 | Not follow the format | | System show message “Field is not follow the format” | | | |
| **Relationships:** N/A  **Business Rules:** | | | | | | |

### **Table : <Admin> Create user**

### **2.3.4.3 <Admin> Manage store**

### **Figure 26 : < Admin> Manage store overview**

### **2.3.4.3.1 <Admin> Create store**



### **Figure 27 : < Admin> Create store**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **USE CASE – UC\_A.03A** | | | | | | |
| **Use Case No.** | | UC\_A.03A | | **Use Case Version** | | 1.0 |
| **Use Case Name** | | Create store | | | | |
| **Author** | | ThangDP | | | | |
| **Date** | | 30/1/2019 | | **Priority** | Normal | |
| **Actor:**   * Admin   **Summary:**   * Admin send create store command   **Goal:**   * Store is created   **Triggers:**   * Admin send create store command   **Preconditions:**   * User role is “admin”.   **Post Conditions:**   * Success: show store created * Fail:   **Main Success Scenario:** | | | | | | |
| **Step** | **Actor Action** | | **System Response** | | | |
| 1 | Admin goes to “Store” link. | | * Display the view on screen.   Store information:  + Name: String, required  + Phone number: number, required  + Address: String, required  + District: ComboBox, required  + City: ComboBox, required  + Longitude: number, required  + Latitude: number, required  + Image store: image, required  + Save: button, required  + Close: image, required | | | |
| 2 | Admin input new data and click “Save”. | | System show store created  [Exception 1] | | | |
| **Alternative Scenario:** | | | | | | |
| **Step** | **Actor Action** | | **System Response** | | | |
|  |  | |  | | | |
| **Exceptions**: | | | | | | |
| **Step** | **Actor Action** | | **System Response** | | | |
| 1 | Any field information is null | | System show message “field is not allow null” | | | |
| **Relationships:** N/A  **Business Rules:** N/A | | | | | | |

### **Table : <Admin> Create store**

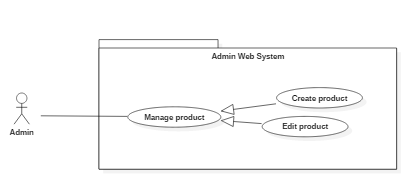
### **2.3.4.3.2 <Admin> Edit store**

### **Figure 28 : < Admin> Edit store**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **USE CASE – UC\_A.03B** | | | | | | |
| **Use Case No.** | | UC\_A.03B | | **Use Case Version** | | 1.0 |
| **Use Case Name** | | Edit store | | | | |
| **Author** | | ThangDP | | | | |
| **Date** | | 31/1/2019 | | **Priority** | Normal | |
| **Actor:**   * Admin   **Summary:**   * Admin send edit store command   **Goal:**   * Update informations’ store   **Triggers:**   * Admin send edit store command.   **Preconditions:**   * User role is “admin”.   **Post Conditions:**   * Success: Store Updated * Fail:   **Main Success Scenario:** | | | | | | |
| **Step** | **Actor Action** | | **System Response** | | | |
| 1 | Admin goes to “Store” link. | | * Display the view on screen.   Store information:  + Name: String, required  + Phone number: number, required  + Address: String, required  + District: ComboBox, required  + City: ComboBox, required  + Longitude: number, required  + Latitude: number, required  + Image store: image, required  + Save: button, required  + Close: image, required | | | |
| 2 | Admin edit data and click “Save” | | System update informations’ store  [Exception 1] | | | |
| **Alternative Scenario:** | | | | | | |
| **Step** | **Actor Action** | | **System Response** | | | |
|  |  | |  | | | |
| **Exceptions**: | | | | | | |
| **Step** | **Actor Action** | | **System Response** | | | |
| 1 | Null or not follow the format | | System show message “Field is not allow null” or  “Field is not correct format” | | | |
| **Relationships:** N/A  **Business Rules:** N/A | | | | | | |

### **Table : <Admin> Edit store**

### **2.3.4.4 <Admin> Manage product**



### **Figure 26 : < Admin> Manage product overview**

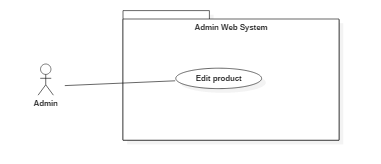
### **2.3.4.4.1 <Admin> Create product**

### **Figure 27 : < Admin> Create product**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **USE CASE – UC\_A.04A** | | | | | | |
| **Use Case No.** | | UC\_A.04A | | **Use Case Version** | | 1.0 |
| **Use Case Name** | | Create product | | | | |
| **Author** | | ThangDP | | | | |
| **Date** | | 30/1/2019 | | **Priority** | Normal | |
| **Actor:**   * Admin   **Summary:**   * Admin send create product command   **Goal:**   * Product is created   **Triggers:**   * Admin send create product command   **Preconditions:**   * User role is “admin”.   **Post Conditions:**   * Success: show product created * Fail:   **Main Success Scenario:** | | | | | | |
| **Step** | **Actor Action** | | **System Response** | | | |
| 1 | * Admin click on row of data table of detail store modal (store need to add new product). | | * System will change view of managing product (with id of store)   Data table:   * “Add new”: button, required. * “Search” text area, required. * “Delete” button, required.   On row of data table   * “Name”: text area, required. * “Price”: text area, required. * “Description”: text area, required. * “Note”: text area, required. * “Status”: text area, required. * “Delete” button group, required. | | | |
| 2 | Admin press “Add new” button. | | * System will modal of add new product: * “Name”: text area, required. * “Name of store”: text area, required. * “Price”: text area, required. * “Description”: text area, required. * “Note”: text area, required. * “Images”: image, required. * “Status”: text area, required. * “Cancel”: button, required. * “Save”: button, required. | | | |
| 3 | Admin input data and click “Save”. | | System show store product  [Exception 1] | | | |
| **Alternative Scenario:** | | | | | | |
| **Step** | **Actor Action** | | **System Response** | | | |
|  |  | |  | | | |
| **Exceptions**: | | | | | | |
| **Step** | **Actor Action** | | **System Response** | | | |
| 1 | Any field information is null | | System show message “field is not allow null” | | | |
| **Relationships:**   * View list of stores. * Search stores. * View detail of stores.   **Business Rules:** N/A | | | | | | |

### **Table : <Admin> Create product**

### **2.3.4.4.2 <Admin> Edit product**

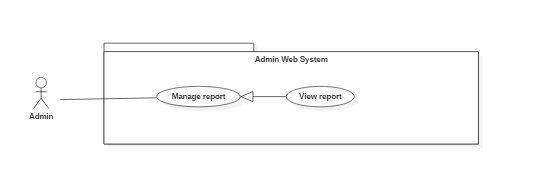


### **Figure 28 : < Admin> Edit product**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **USE CASE – UC\_A.04B** | | | | | | |
| **Use Case No.** | | UC\_A.04B | | **Use Case Version** | | 1.0 |
| **Use Case Name** | | Edit product | | | | |
| **Author** | | ThangDP | | | | |
| **Date** | | 31/1/2019 | | **Priority** | Normal | |
| **Actor:**   * Admin   **Summary:**   * Admin send edit product command   **Goal:**   * Update informations’ product   **Triggers:**   * Admin send edit product command.   **Preconditions:**   * User role is “admin”.   **Post Conditions:**   * Success: Product updated * Fail:   **Main Success Scenario:** | | | | | | |
| **Step** | **Actor Action** | | **System Response** | | | |
| 1 | * Admin click on row of data table of detail store modal (store need to add new product). | | * System will change view of managing product (with id of store)   Data table:   * “Add new”: button, required. * “Search” text area, required. * “Delete” button, required.   On row of data table   * “Name”: text area, required. * “Price”: text area, required. * “Description”: text area, required. * “Note”: text area, required. * “Status”: text area, required. * “Delete” button group, required. | | | |
| 2 | Admin press “Add new” button. | | * System will modal of add new product: * “Name”: text area, required. * “Name of store”: text area, required. * “Price”: text area, required. * “Description”: text area, required. * “Note”: text area, required. * “Images”: image, required. * “Status”: text area, required. * “Cancel”: button, required.   “Save”: button, required. | | | |
| 3 | Admin edit data and click “Save”. | | System update informations’ product  [Exception 1] | | | |
| **Alternative Scenario:** | | | | | | |
| **Step** | **Actor Action** | | **System Response** | | | |
|  |  | |  | | | |
| **Exceptions**: | | | | | | |
| **Step** | **Actor Action** | | **System Response** | | | |
| 1 | Null or not follow the format | | System show message “Field is not allow null” or  “Field is not correct format” | | | |
| **Relationships:**   * View list of stores. * Search stores. * View detail of stores.   **Business Rules:** N/A | | | | | | |

### **Table : <Admin> Edit product**

### **2.3.4.5 <Admin> Manage report**



### **Figure 26 : < Admin> Manage report overview**

### **2.3.4.5.1 <Admin> View repor**

### **Figure 27 : < Admin> View report**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **USE CASE – UC\_A.** | | | | | | |
| **Use Case No.** | | UC\_A. | | **Use Case Version** | | 1.0 |
| **Use Case Name** | | View Report | | | | |
| **Author** | | LamNH | | | | |
| **Date** | | 5/3/2019 | | **Priority** | Normal | |
| **Actor:**   * Admin   **Summary:**   * This use case allows admin to view report and statistics of system.   **Goal:**   * Admin view report of order, customer, shipper, amount.   **Triggers:**   * Admin send log in command on web application (Admin). * System show web’s screen with admin role. * Admin press “Home” link.   **Preconditions:**   * User must be “admin”.   **Post Conditions:**   * Success: Show list of order on screen. * Fail: If list is null, show table with null data on screen.   **Main Success Scenario:** | | | | | | |
| **Step** | **Actor Action** | | **System Response** | | | |
| 1 | Admin goes to admin screen view. | | * Display the view on screen.   + “Home” link.   + “Account” link.   + “Order” link.   + “Product” link.   + “Store” link. | | | |
| 2 | Admin goes to “Home” link. | | * Display the view on screen. * “Actived Shipper”: item, required. * All: number. * Today: number. * This Week: number. * This Month: number. * “Customer Registration”: item, required. * All: number. * Today: number. * This Week: number. * This Month: number. * “Stores Partners”: item, required. * All: number. * Today: number. * This Week: number. * This Month: number. * “Orders Created”: item, required. * All: number. * Today: number. * This Week: number. * This Month: number. * “Ratio (Success)”: item, required. * All: number. * Today: number. * This Week: number. * This Month: number. * “Sold Products”: item, required. * All: number. * Today: number. * This Week: number. * This Month: number. * “Revenue Amount”: item, required. * All: number. * Today: number. * This Week: number. * This Month: number. * “Paid Shipper Amount”: item, required. * All: number. * Today: number. * This Week: number. * This Month: number.   Data Chart which contains:   * + “Type”: Type Report selects, required.   + “Date”: Time Report selects, required.   + “From”: date picker, required.   + “To”: date picker, required.   + “X”: label with date time, required.   + “Y”: label with number of reports, required.   The value of type “Order”:   1. New: Order is new 2. Assigned: Order is assigned. 3. Bought: Order is bought. 4. Received: Order is done. 5. Canceled: Order is cancelled.   The value of type “Product Sold”:   1. Sold Products: Product is sold.   The value of type “Rate Successful”:   1. Success rate: Ratio of success order. 2. Canceled rate: Ratio of canceled order.   The value of type “Total Amount”:   1. Total Amount: Revenue amount. 2. Paid Shipper Amount: Amount paid for shipper.   The value of Date “Day”: dd/mm/yyyy.  The value of Date “Week”: dd/mm/yyyy.  The value of Date “Month”: mm/yyyy.  The value of Date “Year”: yyyy. | | | |
| 3 | User select type, date, from, to for drawing chart report. | | - Make a chart of report. | | | |
| 4 | User select status of chart or label of chart. | | - Show data of report with time of line or label is selected.  +Show entries: select entries for table.  +Type: String.  +Value: String  +Search: search area.  Table which contains:  Data Table which contains:   * + “Buyer Phone”: text area, required.   + “Buyer Name”: text area, required.   + “Shipper Phone”: text area, required.   + “Shipper Name”: text, required.   + “Store”: text area, required.   + “Total Price”: integer, required.   + “Create Time”: integer, required.   + “Status”: text area, required. | | | |
| 5 | User lick link of table. | | Display the view order detail modal on screen:   * “Customer Face”: image area, required. * “Bill”: image area, required. * “Order ID”: text area, required. * “Customer Phone”: text area, required. * “Customer Name”: text area, required. * “Shipper Phone”: text area, required. * “Shipper Name”: text area, required. * “Status”: text area, required. * “longitude”: text area, required. * “Total Price”: text area, required. * “Store Name”: text area, required. * “Store Address”: text area, required. * “Latitude”: text area, required. * “Shipper Earn”: text area, required. * “Customer Description”: text area, required. * “Create Time”: text area, required. * “Receive Time”: text area, required. * “Note”: text area, required.   Data Table:   * + “#”: Code of product, text, required.   + “Product Name”: text area, required.   + “Price”: text, required.   + “Unit”: integer, required.   + “Cost”: double, required.   “Close”: button, required. | | | |
|  | | | | | | |
| **Step** | **Actor Action** | | **System Response** | | | |
| 1 | There is no data of admin on database. | | Show table with null data on screen. | | | |
| **Exceptions**: | | | | | | |
| **Step** | **Actor Action** | | **System Response** | | | |
| 1 | Network is not available | | Show message “Please check your internet”. | | | |
| **Relationships:**   * View Order   **Business Rules:**   * Order only cancel when status is: “New”, “Assigned” | | | | | | |

### **Table : <Admin> View report**

1. **Software System Attribute**

4. 1. **Usability**

Website’s UI is fit for each browser on each device

* Font style: Arial, Sans-serif
* Font size: 12px -26px
* Color: green, black, red, white, blue, Light Slate Grey, Fuego, Honeysuckle…
* Background: White, Catalina Blue, Whisper…
  1. **Reliability**

The information storing on the database is permanent

* 1. **Availability**

The system must be available at any time 24/7.

* 1. **Security**

Only Manager has fully access to the system

Input data is validated before saving to database.

* 1. **Maintainability**

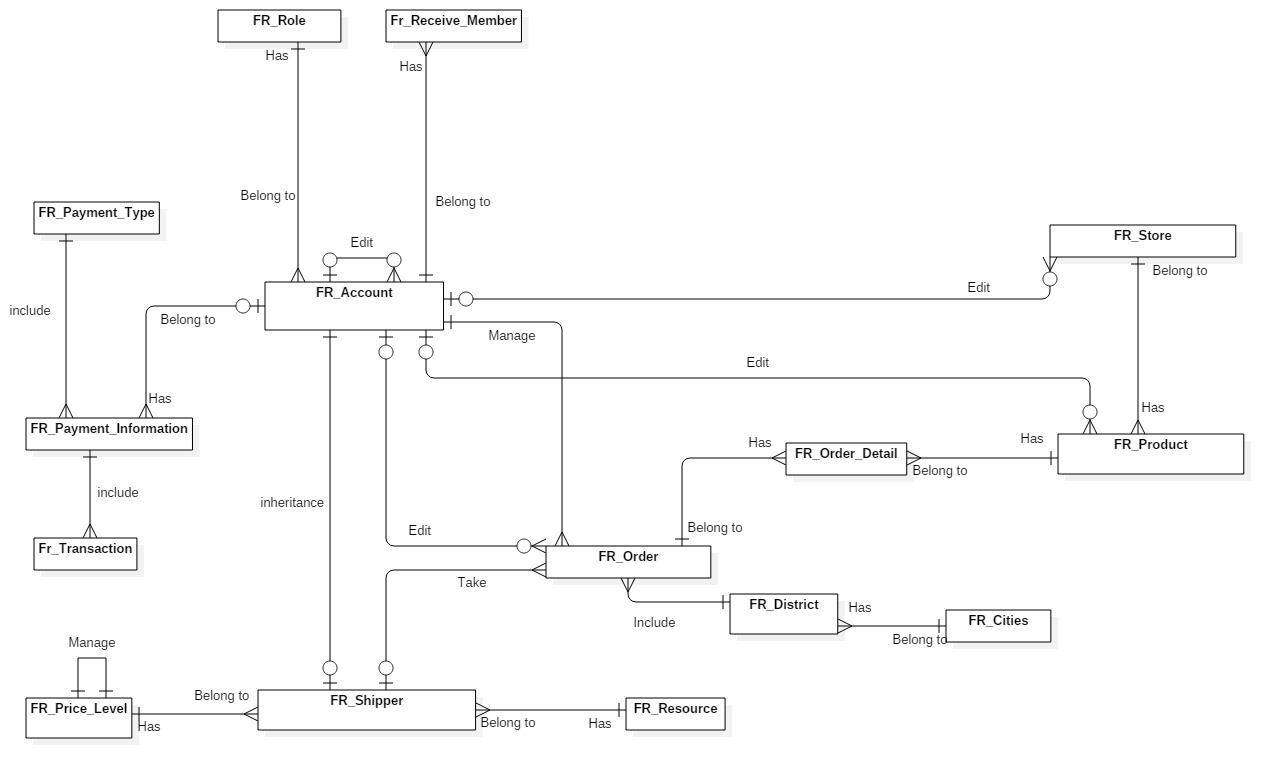
The system is divided into separated modules for easy maintain.

* 1. **Portability**

Web application can be run on Chrome browser version 42 or later.

* 1. **Performance**

System has successfully test with basic standard

1. **Conceptual Diagram**

### **Figure: Data Dictionary**

|  |  |  |
| --- | --- | --- |
| **Entity Data Dictionary: Describe content of all tables** | | |
| **No.** | **Table Name** | **Description** |
| 1 | Fr\_account | Contain account information |
| 2 | Fr\_city | Contain city information |
| 3 | Fr\_district | Contain district data of city |
| 4 | Fr\_order | Contain data of order’s list |
| 5 | Fr\_order\_detail | Contain data of order |
| 6 | Fr\_payment\_information | Contain data of payment |
| 7 | Fr\_payment\_type | Type of payment |
| 8 | Fr\_price\_level | Select price level for shipper |
| 9 | Fr\_product | Contain data of every product of store |
| 10 | Fr\_role | Contain data role of member |
| 13 | Fr\_shipper | Contain data of shipper |
| 14 | Fr\_source | Contain source of data |
| 15 | Fr\_store | Contain data of store |

### **Table: Data Dictionary**

# D. Report No. 4 Software Design Description

# 1. Design Overview

- This document describes the technical and user interface design of MSSC System. It includes the architectural design, the detailed design of common functions and business functions and the design of database model.

- The architectural design describes the overall architecture of the system and the architecture of each main component and subsystem.

- The detailed design describes static and dynamic structure for each component and functions. It includes class diagrams, class explanations and sequence diagrams for each use cases.

- The database design describes the relationships between entities and details of each entity.

- Document overview:

* Section 2: gives an overall description of the system architecture design.
* Section 3: gives component diagrams that describe the connection and integration of the system.
* Section 4: gives the detail design description which includes class diagram, class explanation, and sequence diagram to details the application functions.
* Section 5: describe screens design.
* Section 6: describe a fully attributed ERD.
* Section 7: describe algorithms.

# SystemArchitecture 2. System Architectural Design

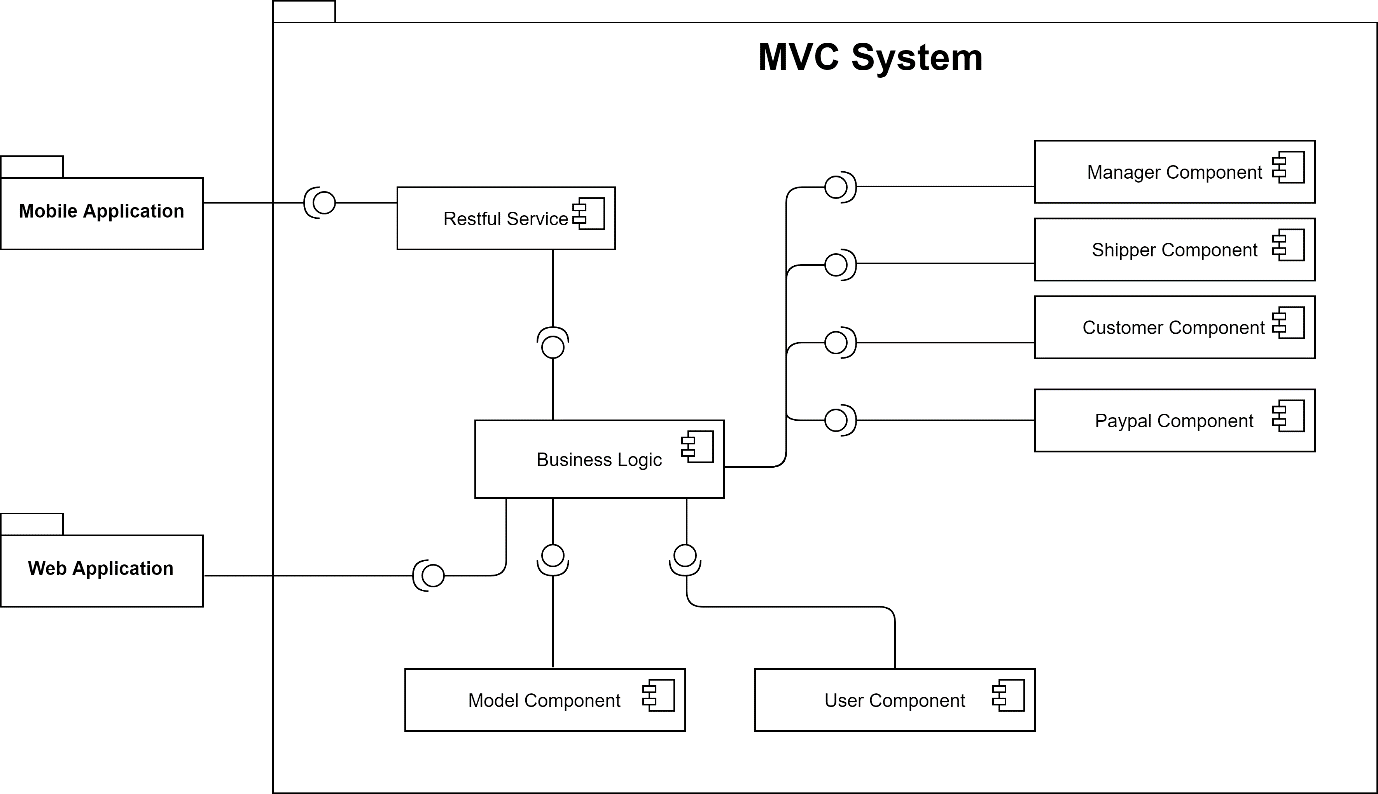
**Figure System Architectural Design**

## 2.1 Web application architecture description

* In web application, the system is developed under MVC architecture style. We choose this architecture for Web application because of following advantages:
  + - Business code is separated with View and Controller that make easy to maintain and easy to understand.
    - Model is used to get and update data from database.
    - Controller takes request from web browser and makes a call to business logic layer.

**2.2 Mobile application architecture description**

* The activity in Mobile client act like a controller to redirect command from view in mobile.
* It takes HTTP Response from server and makes call HTTP Request.



# 3. Component Diagram

**Figure: Component Diagram**

|  |  |
| --- | --- |
| **Component Dictionary: Describe component** | |
| **Component Name** | **Description** |
| Mobile Application | Mobile Application package. |
| Web Application | Web Application package: controller, view. |
| Restful Service | Provide API for Mobile Application. |
| Business Logic | Handle business operation for each component. |
| Model Component | Mapping data from database. |
| User Component | Component to handle User activities in the system. |
| Manager Component | Component to handle Manager activities in the system. |
| Customer Component | Component to handle Customer activities in the system. |
| Shipper Component | Component to handle Shipper activities in the system. |
| PayPal Component | Component to handle PayPal activities in the system. |

**Table Component Dictionary**

# 4. Detail Description

## 4.1 Class Diagram

**Figure Class Diagram**

|  |  |
| --- | --- |
| **Class Dictionary** | |
| **Class Name** | **Description** |
| Payment Type | Abstract class that contains Payment type information. |
| Payment Information | Contain Payment Information |
| Price List | Contain Price List information |
| Role | Contain Role information |
| Account | Contain Account information |
| Shipper | Contain Shipper information |
| Status | Contain Status information |
| Order | Contain Order information |
| Resource | Contain Resource information |
| Order Detail | Contain Order Detail information |
| Town | Contain Town information |
| Store | Contain Store information |
| Product | Contain Product information |
| City | Contain City information |

**Table Class Dictionary**

## 4.2 Class Diagram Explanation

### **4.2.1 Payment Type**

**Attribute**

|  |  |  |  |
| --- | --- | --- | --- |
| Attribute | Type | Visibility | Description |
| id | integer | public | Unique identifier of a payment type. |
| type | String | public | Type of Payment |
| name | String | public | Name of payment’s type. |
| description | String | public | Description of payment’s type. |

Table Payment Type Attribute

**Method**

|  |  |  |  |
| --- | --- | --- | --- |
| Method | Return type | Visibility | Description |
| getter() | string | public | get attribute value |
| setter() | void | public | return value of attribute |

**Table Payment Type Method**

### **4.2.2 Payment Information**

**Attribute**

|  |  |  |  |
| --- | --- | --- | --- |
| Attribute | Type | Visibility | Description |
| id | integer | public | Unique identifier of a payment information. |
| Fr\_Account\_id | integer | public | Linked to table of account |
| Fr\_Payment\_Type\_id | integer | public | Linked to table of payment type |
| username | String | public | Username of account. |
| password | String | public | Password of account. |

**Table Payment Information Attribute**

**Method**

|  |  |  |  |
| --- | --- | --- | --- |
| Method | Return type | Visibility | Description |
| getter() | String | public | get attribute value |
| setter() | void | public | return value of attribute |

**Table Payment Information Method**

### **4.2.3 Account**

**Attribute**

|  |  |  |  |
| --- | --- | --- | --- |
| Attribute | Type | Visibility | Description |
| id | integer | public | Unique identifier of account. |
| Phone\_number | String | public | Phone number of account. |
| Fr\_Role\_id | integer | public | Linked to table of role. |
| password | String | public | Password of account. |
| name | String | public | Full name. |
| email | String | public | Email of account. |
| extra\_point | integer | public | Extra point of account. |
| report\_point | integer | public | Report point of account |
| user\_image | Longblob | public | Image of account. |
| nation\_id | String | public | Nation Id Card of account. |
| nation\_id\_created\_date | Bigint | public | Create Date of Nation Id Card |
| date\_of\_birth | Bigint | public | Brithday of account. |
| create\_time | Bigint | public | Create time of account. |
| update\_time | Bigint | public | Update time of account. |
| delete\_time | Bigint | public | Delete time of account. |
| note | String | public | Note of account. |
| status | integer | public | Status of account. |
| editor\_id | integer | public | Id of editor’s account. |
| avatar | Longblob | public | Avatar of account. |

**Table Account Attribute**

**Method**

|  |  |  |  |
| --- | --- | --- | --- |
| Method | Return type | Visibility | Description |
| getter() | string | public | get attribute value |
| setter() | void | public | return value of attribute |

**Table Account Method**

### **4.2.4 City**

**Attribute**

|  |  |  |  |
| --- | --- | --- | --- |
| Attribute | Type | Visibility | Description |
| id | integer | public | Unique identifier of city. |
| name | String | public | Name of city. |
| postal\_code | String | public | Postal code of city |

**Table City Attribute**

**Method**

|  |  |  |  |
| --- | --- | --- | --- |
| Method | Return type | Visibility | Description |
| getter() | string | public | get attribute value |
| setter() | void | public | return value of attribute |

**Table City Method**

### **4.2.5 District**

**Attribute**

|  |  |  |  |
| --- | --- | --- | --- |
| Attribute | Type | Visibility | Description |
| id | integer | public | Unique identifier of district. |
| name | String | public | Name of city. |
| Fr\_City\_id | integer | public | Linked to table of city. |

**Table District Attribute**

**Method**

|  |  |  |  |
| --- | --- | --- | --- |
| Method | Return type | Visibility | Description |
| getter() | string | public | get attribute value |
| setter() | void | public | return value of attribute |

**Table District Method**

### **4.2.6 Order**

**Attribute**

|  |  |  |  |
| --- | --- | --- | --- |
| Attribute | Type | Visibility | Description |
| id | integer | public | Unique identifier of order. |
| editor\_id | integer | public | Id of editor account. |
| Fr\_Account\_id | integer | public | Linked to table of account(customer). |
| Fr\_Shipper\_id | integer | public | Linked to table of shipper. |
| buyer\_face | Longblob | public | Face’s Image of customer. |
| bill | String | public | Bill of customer. |
| order\_code | String | public | Code of order. |
| total\_price | double | public | Total of price. |
| book\_time | Bigint | public | Book time of order |
| receive\_time | Bigint | public | Receive time of order |
| shipper\_earn | double | public | Shipper earn from order. |
| shipper\_address | String | public | Address of delivery. |
| Fr\_District\_id | integer | public | Linked to table of district. |
| longitude | double | public | Longitude of earth. |
| latitude | double | public | Latitude of earth. |
| customer\_description | String | public | Description of customer. |
| create\_time | Bigint | public | Create time of order. |
| update\_time | Bigint | public | Update time of order. |
| delete\_time | Bigint | public | Delete time of order. |
| note | String | public | Note of order. |
| status | integer | public | Status of order. |

**Table Order Attribute**

**Method**

|  |  |  |  |
| --- | --- | --- | --- |
| Method | Return type | Visibility | Description |
| getter() | string | public | get attribute value |
| setter() | void | public | return value of attribute |

**Table Order Method**

### **4.2.7 Order Detail**

**Attribute**

|  |  |  |  |
| --- | --- | --- | --- |
| Attribute | Type | Visibility | Description |
| id | integer | public | Unique identifier of order detail. |
| Fr\_Order\_id | integer | public | Linked to table of order |
| Fr\_Product\_id | integer | public | Linked to table of product |
| unit\_price | double | public | Unit\_price of product. |
| quantity | integer | public | Quantity of product. |

**Table Order Detail Attribute**

**Method**

|  |  |  |  |
| --- | --- | --- | --- |
| Method | Return type | Visibility | Description |
| getter() | string | public | get attribute value |
| setter() | void | public | return value of attribute |

**Table Order Detail Method**

### **4.2.8 Price Level**

**Attribute**

|  |  |  |  |
| --- | --- | --- | --- |
| Attribute | Type | Visibility | Description |
| id | integer | public | Unique identifier of price level. |
| level\_name | String | public | Name of price level |
| description | String | public | Description of price level. |
| price | double | public | Price of level. |

**Table Price Level Attribute**

**Method**

|  |  |  |  |
| --- | --- | --- | --- |
| Method | Return type | Visibility | Description |
| getter() | string | public | get attribute value |
| setter() | void | public | return value of attribute |

**Table Price Level Method**

### **4.2.9 Product**

**Attribute**

|  |  |  |  |
| --- | --- | --- | --- |
| Attribute | Type | Visibility | Description |
| id | integer | public | Unique identifier of product. |
| Fr\_Store\_id | integer | public | Linked to table of store. |
| price | double | public | Price of product. |
| product\_image | longblob | public | Image og product. |
| rating | double | public | Rating for product. |
| rating\_count | integer | public | Number of rating for product |
| description | String | public | Description for product. |
| create\_time | bigint | public | Create time of product. |
| update\_time | bigint | public | Update time of product |
| delete\_time | bigint | public | Delete time of product |
| note | String | public | Note of product. |
| status | integer | public | Status of product. |
| editor\_id | integer | public | Id of editor’s product. |

**Table Product Attribute**

**Method**

|  |  |  |  |
| --- | --- | --- | --- |
| Method | Return type | Visibility | Description |
| getter() | string | public | get attribute value |
| setter() | void | public | return value of attribute |

**Table Product Method**

### **4.2.10 Rating**

**Attribute**

|  |  |  |  |
| --- | --- | --- | --- |
| Attribute | Type | Visibility | Description |
| id | integer | public | Unique identifier of rating. |
| Fr\_Store\_id | integer | public | Linked to table of store. |
| Fr\_Account\_id | integer | public | Linked to table of account. |
| Fr\_Product\_id | integer | public | Linked to table of product. |
| rating | double | public | Rating for product. |

**Table Rating Attribute**

**Method**

|  |  |  |  |
| --- | --- | --- | --- |
| Method | Return type | Visibility | Description |
| getter() | string | public | get attribute value |
| setter() | void | public | return value of attribute |

**Table Rating Method**

### **4.2.11 Role**

**Attribute**

|  |  |  |  |
| --- | --- | --- | --- |
| Attribute | Type | Visibility | Description |
| id | integer | public | Unique identifier of role. |
| name | String | public | Name of role. |

**Table Role Attribute**

**Method**

|  |  |  |  |
| --- | --- | --- | --- |
| Method | Return type | Visibility | Description |
| getter() | string | public | get attribute value |
| setter() | void | public | return value of attribute |

**Table Role Method**

### **4.2.12 Schedule**

**Attribute**

|  |  |  |  |
| --- | --- | --- | --- |
| Attribute | Type | Visibility | Description |
| id | integer | public | Unique identifier of schedule. |
| Fr\_Store\_id | integer | public | Linked to table of store. |
| Fr\_Weekenday\_id | integer | public | Linked to table of account. |
| open\_hour | integer | public | Open time of store |
| open \_mintues | integer | public | Open time of store |
| close \_hour | integer | public | Close time of store |
| close\_ mintues | integer | public | Close time of store |

**Table Schedule Attribute**

**Method**

|  |  |  |  |
| --- | --- | --- | --- |
| Method | Return type | Visibility | Description |
| getter() | string | public | get attribute value |
| setter() | void | public | return value of attribute |

**Table Schedule Method**

### **4.2.13 Shipper**

**Attribute**

|  |  |  |  |
| --- | --- | --- | --- |
| Attribute | Type | Visibility | Description |
| id | integer | public | Unique identifier of shipper. |
| Fr\_Store\_id | integer | public | Linked to table of store. |
| introduce | String | public | Introduce of shipper. |
| sum\_revenue | integer | public | Sum Revenue of shipper |
| nation\_id\_front\_image | longblob | public | Front image of Nation Id Card |
| nation\_id\_back\_image | longblob | public | Back image of Nation Id Card |
| bike\_registration\_id | String | public | Number of Bike Registration Card |
| bike\_registration\_date | bigint | public | Date of Bike Registration Card |
| bike\_registrion\_front\_image | longblob | public | Front image of Bike Id Card |
| bike \_ registrion\_back\_image | longblob | public | Back image of Bike Id Card |
| Fr\_Source\_id | integer | public | Linked to table of source. |
| Fr\_Price\_Level\_id | integer | public | Linked to table of price level. |

**Table Shipper Attribute**

**Method**

|  |  |  |  |
| --- | --- | --- | --- |
| Method | Return type | Visibility | Description |
| getter() | string | public | get attribute value |
| setter() | void | public | return value of attribute |

**Table Shipper Method**

### **4.2.14 Source**

**Attribute**

|  |  |  |  |
| --- | --- | --- | --- |
| Attribute | Type | Visibility | Description |
| id | integer | public | Unique identifier of source. |
| name | String | public | Name of source. |
| description | String | public | Description of source. |

**Table Source Attribute**

**Method**

|  |  |  |  |
| --- | --- | --- | --- |
| Method | Return type | Visibility | Description |
| getter() | string | public | get attribute value |
| setter() | void | public | return value of attribute |

**Table Source Method**

### **4.2.15 Store**

**Attribute**

|  |  |  |  |
| --- | --- | --- | --- |
| Attribute | Type | Visibility | Description |
| id | integer | public | Unique identifier of store. |
| store\_name | String | public | Name of store. |
| address | String | public | Description of store. |
| Fr\_District\_id | integer | public | Linked to table of district. |
| longitude | double | public | Longitude of store. |
| latitude | double | public | Latitude of store. |
| store\_image | longblob | public | Image of store. |
| rating | double | public | Rating for store. |
| rating\_count | integer | public | Number of rating for store |
| phone | String | public | Phone of store. |
| create\_time | Bigint | public | Create time of store. |
| update\_time | Bigint | public | Update time of store. |
| delete\_time | Bigint | public | Delete time of store. |
| note | String | public | Note of store. |
| status | integer | public | Status of store. |
| editor\_id | integer | public | Id of editor store. |

**Table Store Attribute**

**Method**

|  |  |  |  |
| --- | --- | --- | --- |
| Method | Return type | Visibility | Description |
| getter() | string | public | get attribute value |
| setter() | void | public | return value of attribute |

**Table Store Method**

### **4.2.16 Weekday**

**Attribute**

|  |  |  |  |
| --- | --- | --- | --- |
| Attribute | Type | Visibility | Description |
| id | integer | public | Unique identifier of weekday. |
| name | String | public | Name of day on week. |

**Table Weekday Attribute**

**Method**

|  |  |  |  |
| --- | --- | --- | --- |
| Method | Return type | Visibility | Description |
| getter() | string | public | get attribute value |
| setter() | void | public | return value of attribute |

**Table Weekday Method**

## 4.3 Interaction Diagram

### **4.3.1 <User>**

### **4.3.2 <Shipper>**

### **4.3.3 <Admin>**

### **4.3.3.1 <Admin> View list of Admins**

### **Figure : < Admin> View list of Admins**

### **4.3.3.2 <Admin> Search Admins**

### **Figure: < Admin> Search Admins**

### **4.3.3.3 <Admin> Add new Admins**

### **Figure: < Admin> Add new Admins**

### **4.3.3.4 <Admin> View detail of Admins**

### **Figure: < Admin> View detail of Admins**

### **4.3.3.5 <Admin> Edit Admins**

### **Figure: < Admin> Edit Admins**

### **4.3.3.6 <Admin> View list of Customers**

### **Figure: < Admin> View list of Customers**

### **4.3.3.7 <Admin> Search Customer**

### **Figure: < Admin> Search Customer**

### **4.3.3.8 <Admin> View Detail of Customer**

### **Figure: < Admin> View Detail of Customer**

### **4.3.3.9 <Admin> View list of Shipper**

### **Figure: < Admin> View Detail of Shipper**

### **4.3.3.10 <Admin> Search Shipper**

### **Figure: < Admin> Search Shipper**

### **4.3.3.11 <Admin> Add new Shipper**

### **Figure: < Admin> Add new Shipper**

### **4.3.3.12 <Admin> View detail of Shipper**

### **Figure: < Admin> View Detail of Shipper**

### **4.3.3.13 <Admin> Edit Shipper**

### **Figure: < Admin> Edit Shipper**

### **4.3.3.14 <Admin> View list of Store**

### **Figure: < Admin> View List of Store**

### **4.3.3.15 <Admin> Search Store**

### **Figure: < Admin> Search Store**

### **4.3.3.16 <Admin> Add new Store**

### Figure: < Admin> Add new Store

### **4.3.3.17 <Admin> View detail of Store**

### **Figure: < Admin> View Detail of Store**

### **4.3.3.18 <Admin> Edit Store**

### **Figure: < Admin> Edit Store**

### **4.3.3.19 <Admin> View list of Product**

### **Figure: < Admin> View List of Product**

### **4.3.3.20 <Admin> Search Product**

### **Figure: < Admin> Search Product**

### **4.3.3.21 <Admin> Add new Product**

### **Figure: < Admin> Add new Product**

### **4.3.3.22 <Admin> View detail of Product**

### **Figure: < Admin> View Detail of Product**

### **4.3.3.23 <Admin> Edit Product**

### **Figure: < Admin> Edit Product**

### **4.3.3.24 <Admin> View List of Order**

### **Figure: < Admin> View List of Order**

### **4.3.3.25 <Admin> View Order**

### **Figure: < Admin> View Order**

### **4.3.3.26 <Admin> Cancel Order**

### **Figure: < Admin> Cancel Order**

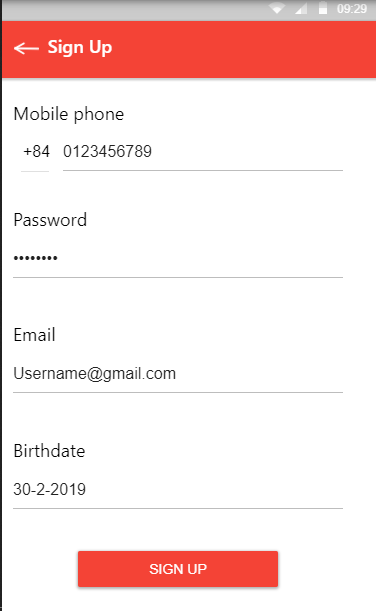
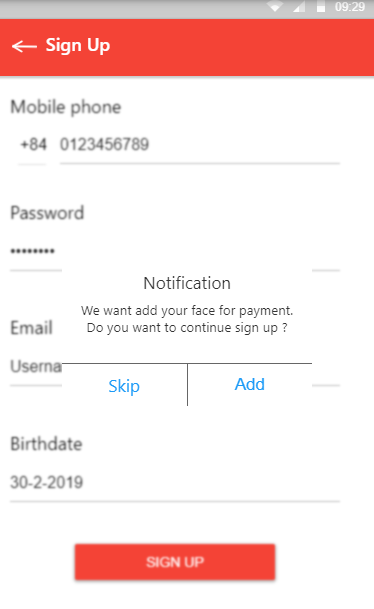
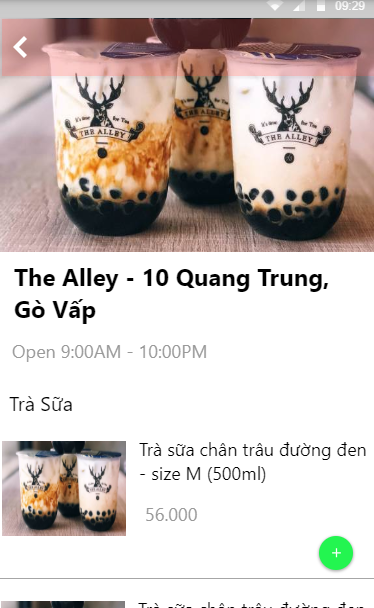
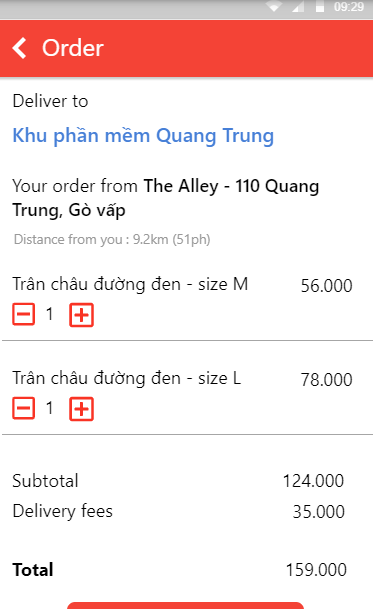
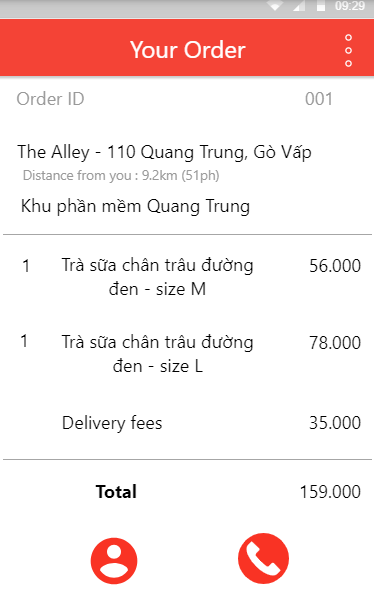
### **4.3.3.27 <Admin> View Report**

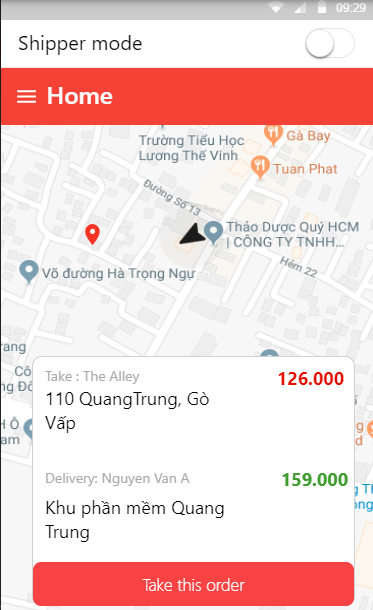
### **Figure: < Admin> View Report**

# 5. Interface

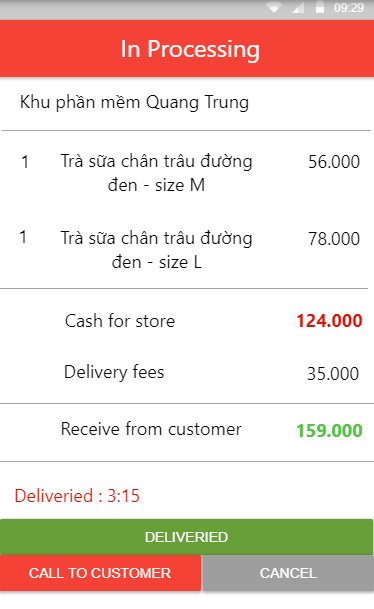
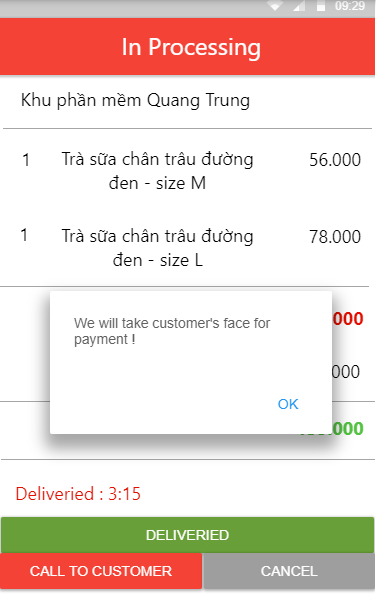
## 5.1 Mobile Application

* **Customer Mode**

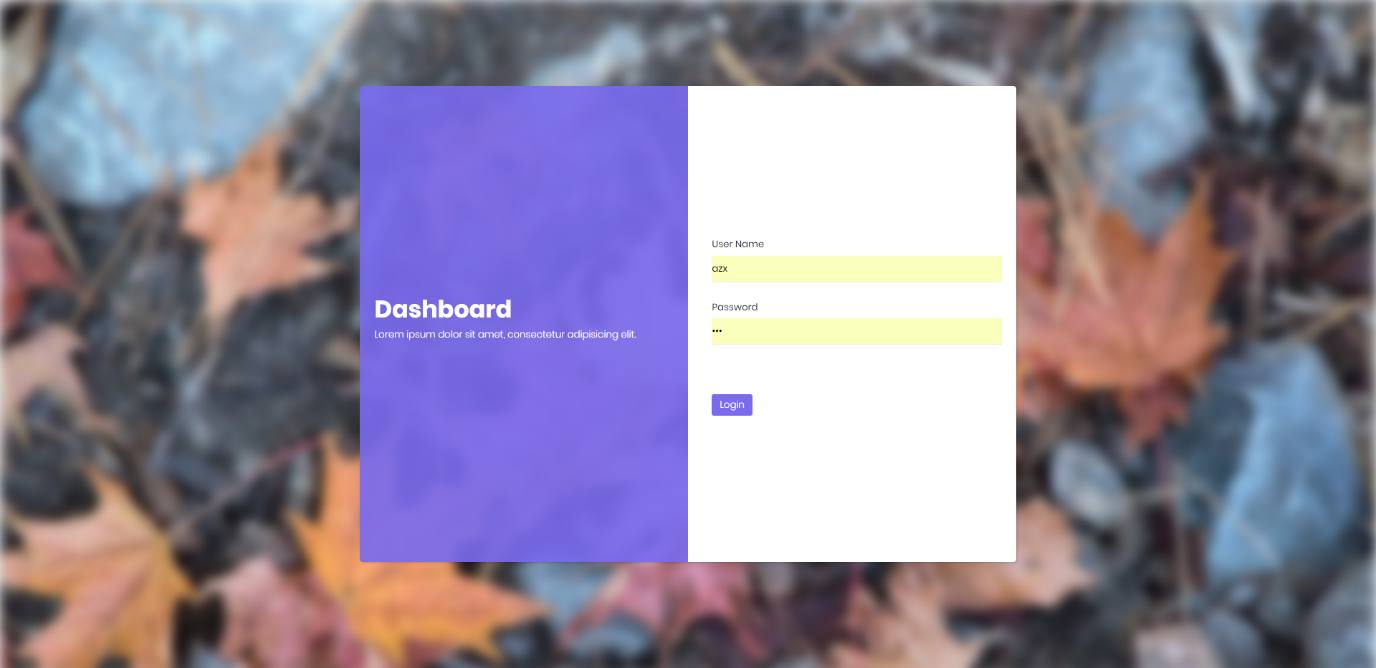
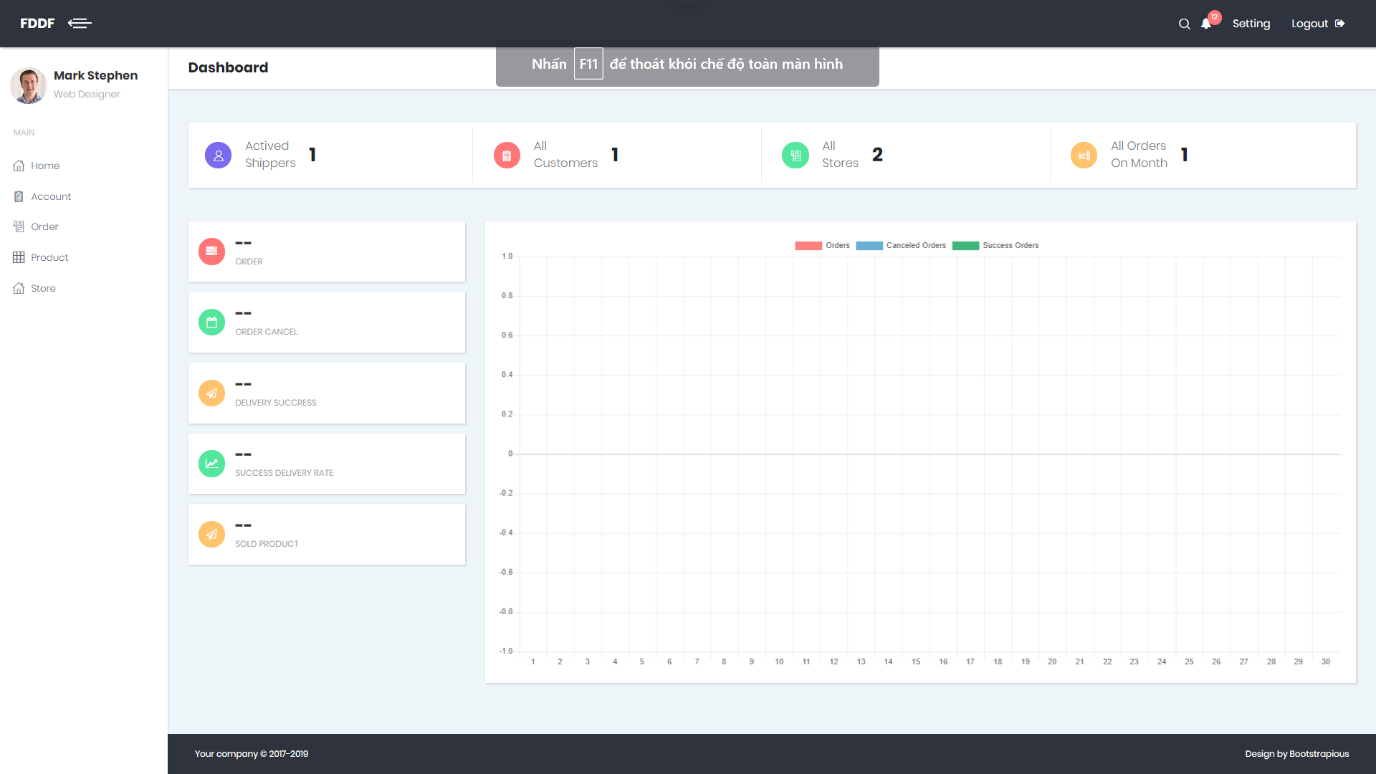
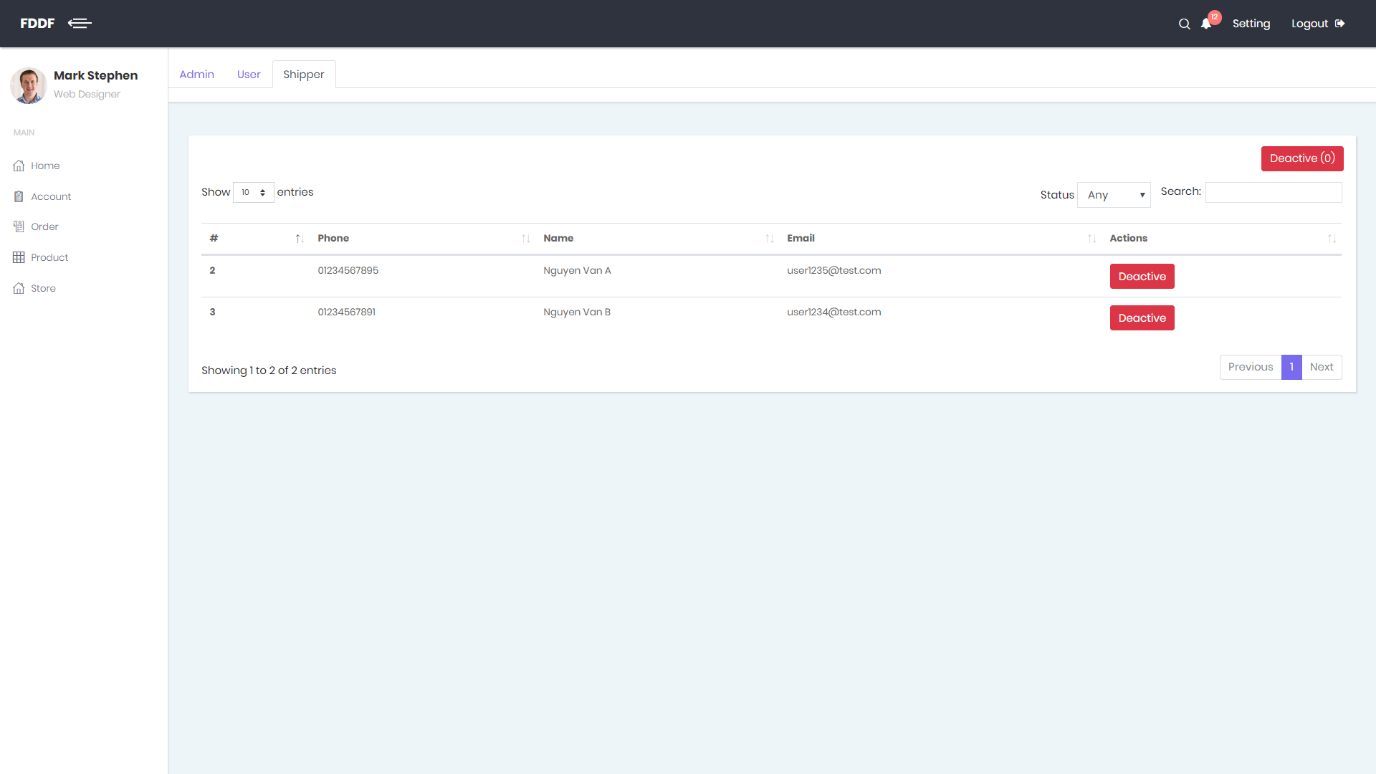
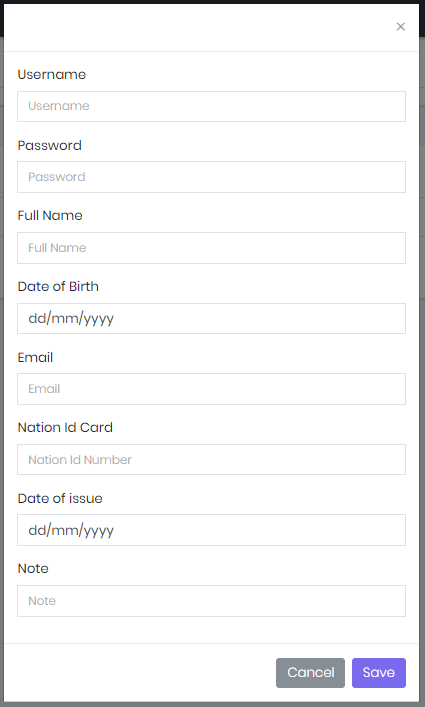
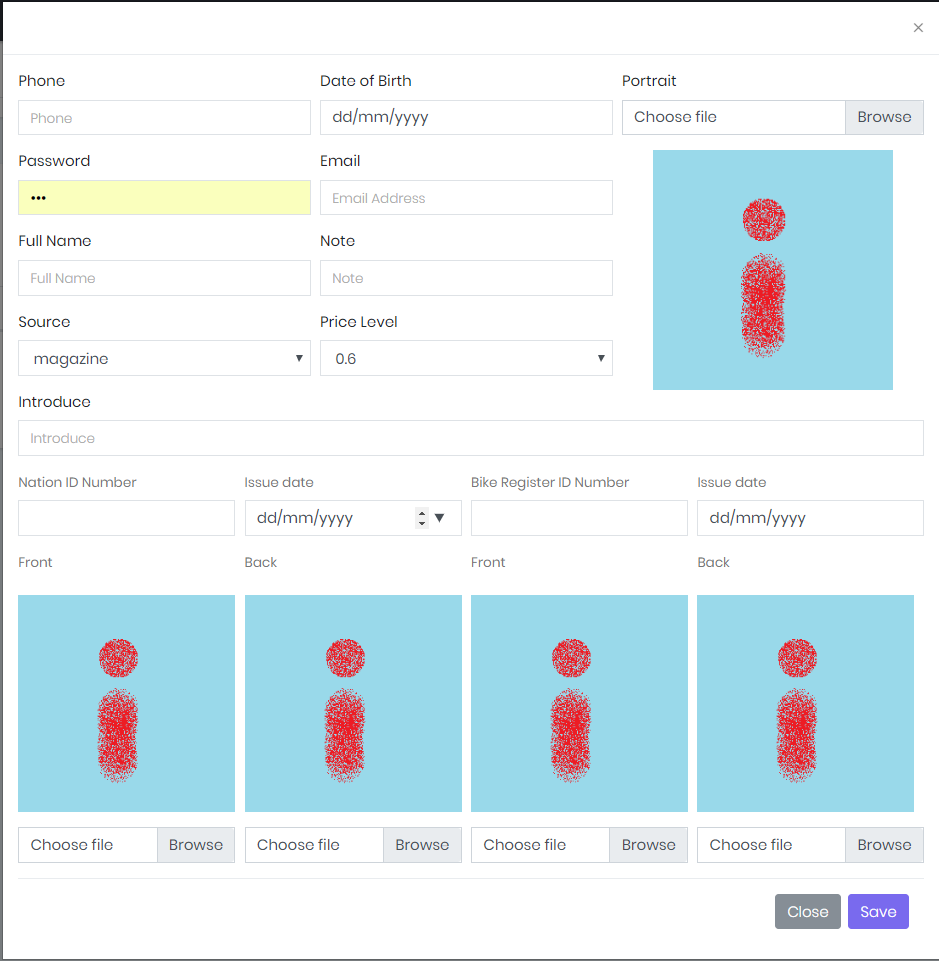
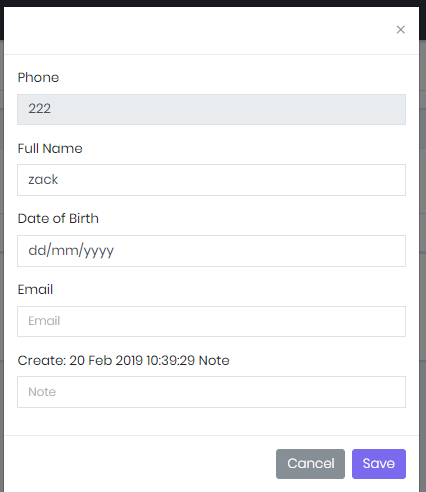
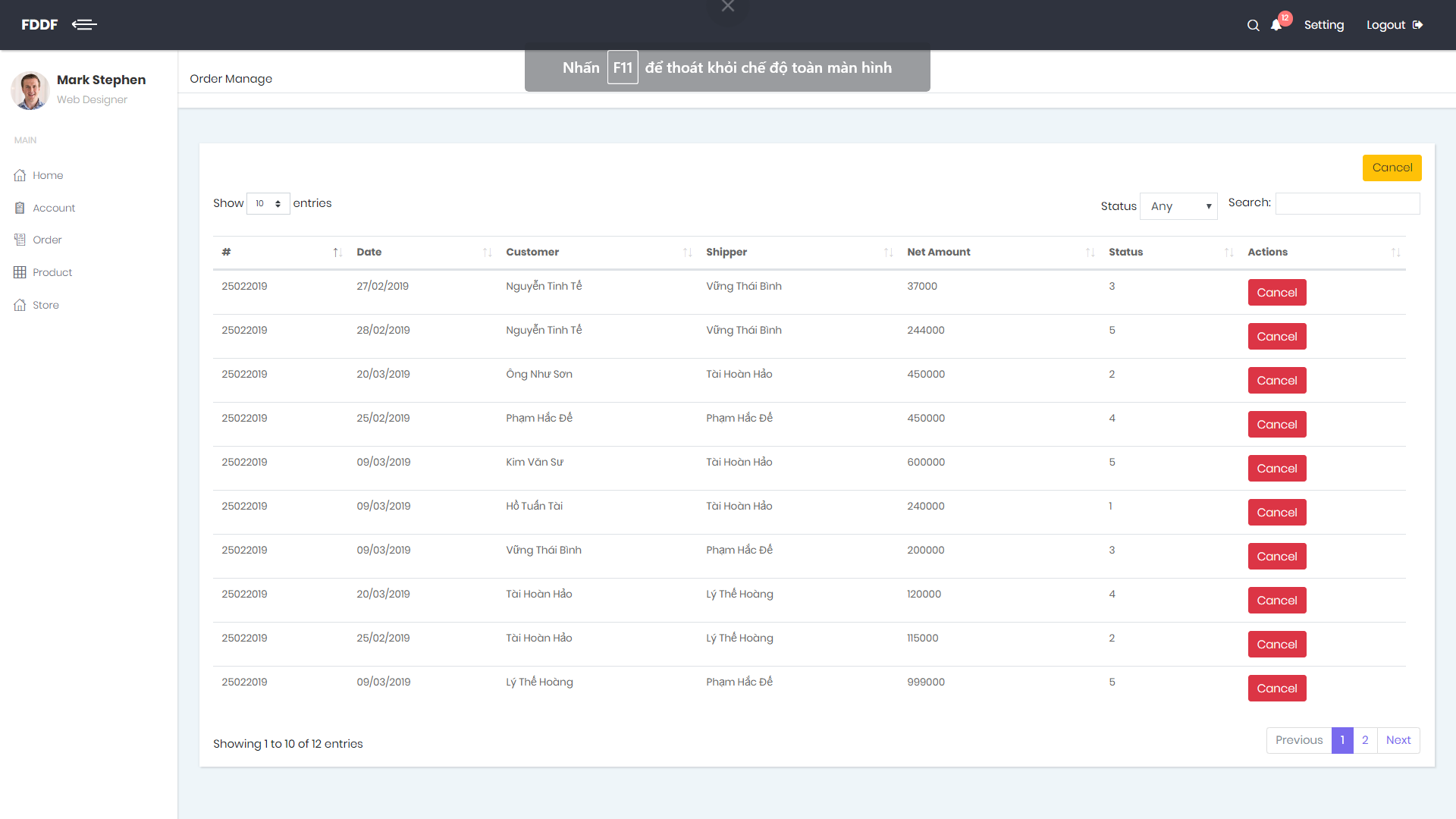
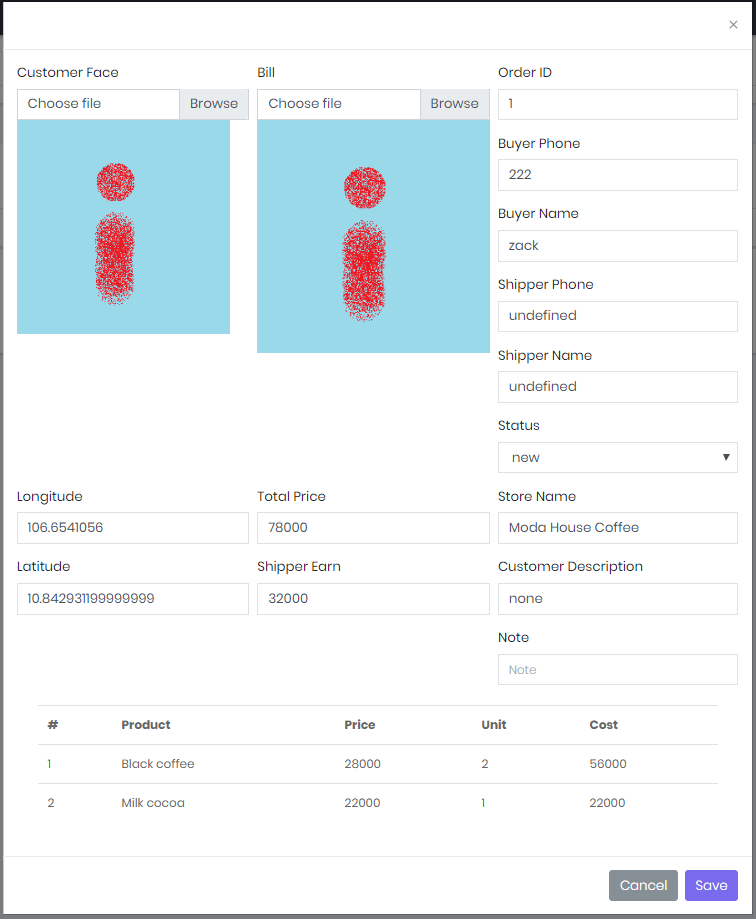
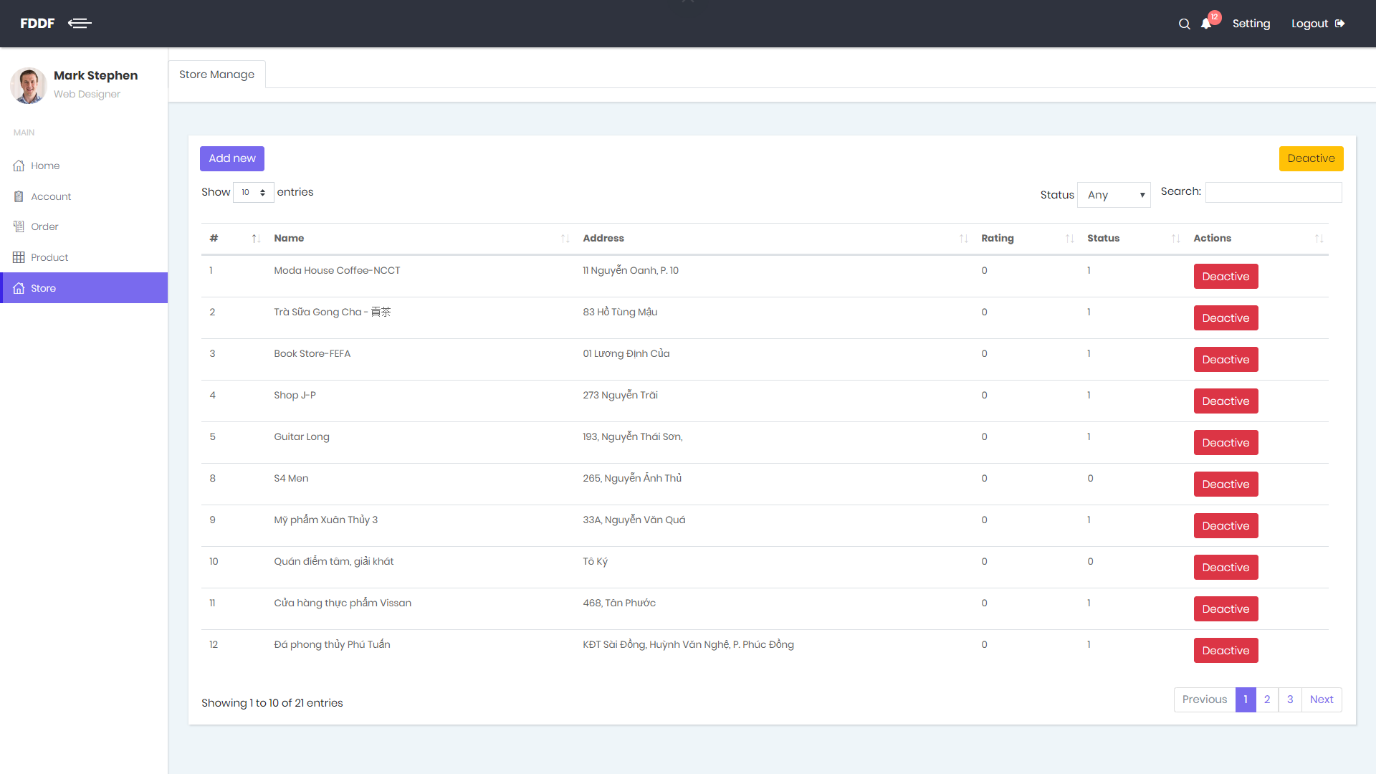
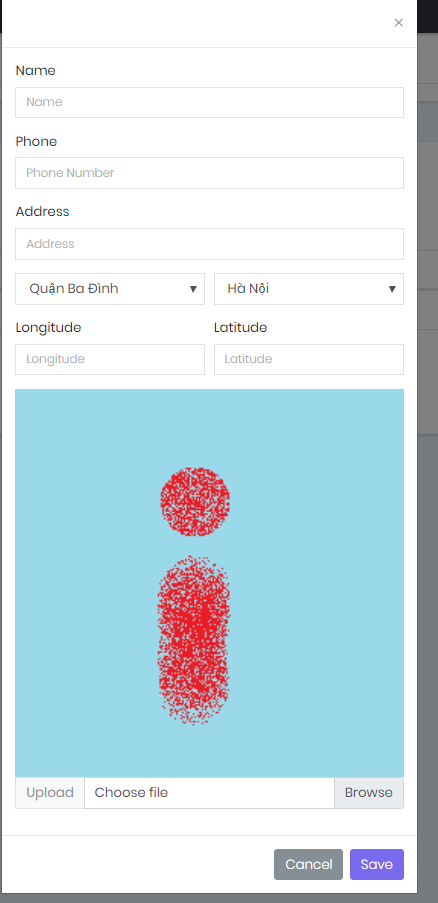
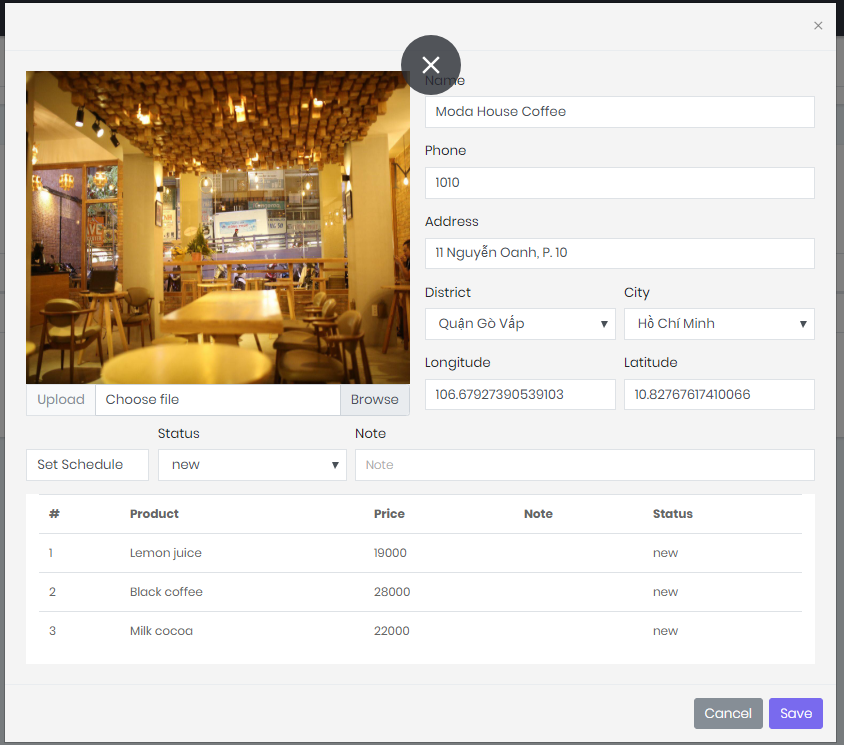
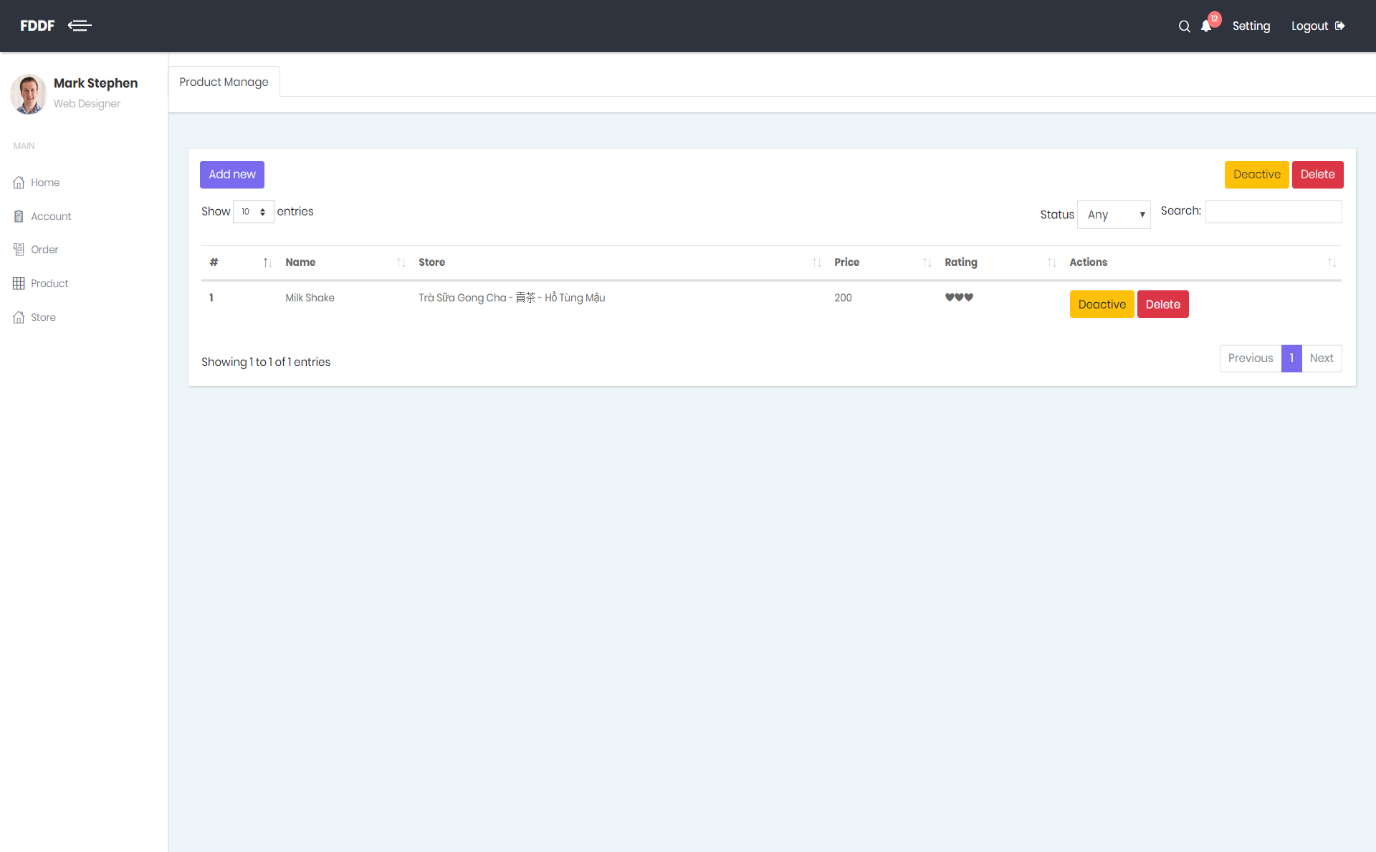
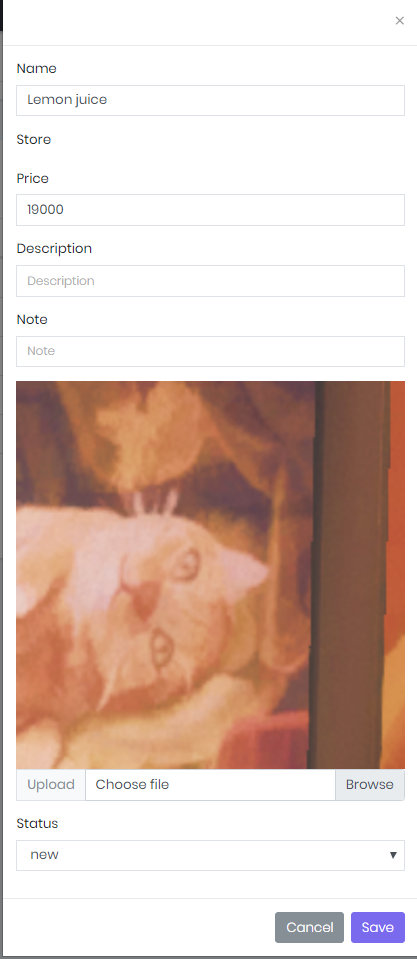
1. Login
2. Register
3. Face Acceptation
4. Home
5. Product/Store
6. Cart Order
7. Map/ Fee
8. Order Detail



* **Shipper Mode**

1. Take Order
2. Order Detail
3. ****Take Payment

## 5.2 Web Application

1. Login
2. Home and Statistic
3. Manage Account
4. Add -Detail Admin Modal
5. Add -Detail Shipper Modal
6. Detail Customer Modal
7. Manage Order
8. Order Detail Modal
9. Manage Store
10. Add New Store
11. Store Detail Popup
12. Manage Product
13. Add-Detail Product Popup

# 6. Database Design

## 6.1 Entity Relationship Diagram (ERD)

**Figure Entity Relation Diagram**

## 6.2 Data Dictionary

|  |  |  |
| --- | --- | --- |
| **Entity Data Dictionary: Describe content of all tables** | | |
| **No.** | **Table Name** | **Description** |
| 1 | Fr\_account | Contain account information |
| 2 | Fr\_city | Contain city information |
| 3 | Fr\_district | Contain district data of city |
| 4 | Fr\_order | Contain data of order’s list |
| 5 | Fr\_order\_detail | Contain data of order |
| 6 | Fr\_payment\_information | Contain data of payment |
| 7 | Fr\_payment\_type | Type of payment |
| 8 | Fr\_price\_level | Select price level for shipper |
| 9 | Fr\_product | Contain data of every product of store |
| 10 | Fr\_rating | Contain data rating of account for product and store |
| 11 | Fr\_role | Contain data role of member |
| 12 | Fr\_schedule | Contain open and close time of store |
| 13 | Fr\_shipper | Contain data of shipper |
| 14 | Fr\_source | Contain source of data |
| 15 | Fr\_store | Contain data of store |
| 16 | Fr\_weekday | Contian id for everyday (Monday, Tuesday, Sunday,…) |

**Table Data Dictionary**

# 7. Algorithms

# E. Report No. 5 System Implementation & Test

# 1. Introduction

## 1.1 Overview

This section provides in detail all necessary information about implementation information and testing procedure of FPS includes test plans, test cases, test result and risks estimations.

## 1.2 Test Approach

- We use black box / white box….

# 2. Database Relationship Diagram

## 2.1 Physical Diagram

**Figure Physical Diagram**

## 2.2 Data Dictionary

|  |  |  |
| --- | --- | --- |
| **Entity Data Dictionary: Describe content of all tables** | | |
| **No.** | **Table Name** | **Description** |
| 1 | Fr\_account | Contain account information |
| 2 | Fr\_city | Contain city information |
| 3 | Fr\_district | Contain district data of city |
| 4 | Fr\_order | Contain data of order’s list |
| 5 | Fr\_order\_detail | Contain data of order |
| 6 | Fr\_payment\_information | Contain data of payment |
| 7 | Fr\_payment\_type | Type of payment |
| 8 | Fr\_price\_level | Select price level for shipper |
| 9 | Fr\_product | Contain data of every product of store |
| 10 | Fr\_rating | Contain data rating of account for product and store |
| 11 | Fr\_role | Contain data role of member |
| 12 | Fr\_schedule | Contain open and close time of store |
| 13 | Fr\_shipper | Contain data of shipper |
| 14 | Fr\_source | Contain source of data |
| 15 | Fr\_store | Contain data of store |
| 16 | Fr\_weekday | Contian id for everyday (Monday, Tuesday, Sunday,…) |

**Figure Data Dictionary**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Entity | Attribute | Type | Visibility | Description |
| Fr\_payment\_type | id | integer | public | Unique identifier of a payment type. |
| type | String | public | Type of Payment |
| name | String | public | Name of payment’s type. |
| description | String | public | Description of payment’s type. |

**Table Payment Type Attribute**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Entity | Attribute | Type | Visibility | Description |
| Fr\_payment\_information | id | integer | public | Unique identifier of a payment information. |
| Fr\_Account\_id | integer | public | Linked to table of account |
| Fr\_Payment\_Type\_id | integer | public | Linked to table of payment type |
| username | String | public | Username of account. |
| password | String | public | Password of account. |

**Table Payment Information Attribute**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Entity | Attribute | Type | Visibility | Description |
| Fr\_account | id | integer | public | Unique identifier of account. |
| Phone\_number | String | public | Phone number of account. |
| Fr\_Role\_id | integer | public | Linked to table of role. |
| password | String | public | Password of account. |
| name | String | public | Full name. |
| email | String | public | Email of account. |
| extra\_point | integer | public | Extra point of account. |
| report\_point | integer | public | Report point of account |
| user\_image | Longblob | public | Image of account. |
| nation\_id | String | public | Nation Id Card of account. |
| nation\_id\_created\_date | Bigint | public | Create Date of Nation Id Card |
| date\_of\_birth | Bigint | public | Brithday of account. |
| create\_time | Bigint | public | Create time of account. |
| update\_time | Bigint | public | Update time of account. |
| delete\_time | Bigint | public | Delete time of account. |
| note | String | public | Note of account. |
| status | integer | public | Status of account. |
| editor\_id | integer | public | Id of editor’s account. |
| avatar | Longblob | public | Avatar of account. |

**Table Account Attribute**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Entity | Attribute | Type | Visibility | Description |
| Fr\_city | id | integer | public | Unique identifier of city. |
| name | String | public | Name of city. |
| postal\_code | String | public | Postal code of city |

**Table City Attribute**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Entity | Attribute | Type | Visibility | Description |
| Fr\_district | id | integer | public | Unique identifier of city. |
| name | String | public | Name of city. |
| Fr\_City\_id | integer | public | Linked to table of city. |

**Table District Attribute**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Entity | Attribute | Type | Visibility | Description |
| Fr\_order | id | integer | public | Unique identifier of order. |
| editor\_id | integer | public | Id of editor account. |
| Fr\_Account\_id | integer | public | Linked to table of account(customer). |
| Fr\_Shipper\_id | integer | public | Linked to table of shipper. |
| buyer\_face | Longblob | public | Face’s Image of customer. |
| bill | String | public | Bill of customer. |
| order\_code | String | public | Code of order. |
| total\_price | double | public | Total of price. |
| book\_time | Bigint | public | Book time of order |
| receive\_time | Bigint | public | Receive time of order |
| shipper\_earn | double | public | Shipper earn from order. |
| shipper\_address | String | public | Address of delivery. |
| Fr\_District\_id | integer | public | Linked to table of district. |
| longitude | double | public | Longitude of earth. |
| latitude | double | public | Latitude of earth. |
| customer\_description | String | public | Description of customer. |
| create\_time | Bigint | public | Create time of order. |
| update\_time | Bigint | public | Update time of order. |
| delete\_time | Bigint | public | Delete time of order. |
| note | String | public | Note of order. |
| status | integer | public | Status of order. |

### **Table Order Attribute**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Entity | Attribute | Type | Visibility | Description |
| Fr\_order\_detail | id | integer | public | Unique identifier of order detail. |
| Fr\_Order\_id | integer | public | Linked to table of order |
| Fr\_Product\_id | integer | public | Linked to table of product |
| unit\_price | double | public | Unit\_price of product. |
| quantity | integer | public | Quantity of product. |

**Table Order Detail Attribute**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Entity | Attribute | Type | Visibility | Description |
| Fr\_price\_level | id | integer | public | Unique identifier of price level. |
| level\_name | String | public | Name of price level |
| description | String | public | Description of price level. |
| price | double | public | Price of level. |

**Table Price Level Attribute**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Entity | Attribute | Type | Visibility | Description |
| Fr\_product | id | integer | public | Unique identifier of product. |
| Fr\_Store\_id | integer | public | Linked to table of store. |
| price | double | public | Price of product. |
| product\_image | longblob | public | Image og product. |
| rating | double | public | Rating for product. |
| rating\_count | integer | public | Number of rating for product |
| description | String | public | Description for product. |
| create\_time | bigint | public | Create time of product. |
| update\_time | bigint | public | Update time of product |
| delete\_time | bigint | public | Delete time of product |
| note | String | public | Note of product. |
| status | integer | public | Status of product. |
| editor\_id | integer | public | Id of editor’s product. |

**Table Product Attribute**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Entity | Attribute | Type | Visibility | Description |
| Fr\_rating | id | integer | public | Unique identifier of rating. |
| Fr\_Store\_id | integer | public | Linked to table of store. |
| Fr\_Account\_id | integer | public | Linked to table of account. |
| Fr\_Product\_id | integer | public | Linked to table of product. |
| rating | double | public | Rating for product. |

**Table Rating Attribute**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Entity | Attribute | Type | Visibility | Description |
| Fr\_role | id | integer | public | Unique identifier of rating. |
| name | String | public | Name of role. |

**Table Role Attribute**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Entity | Attribute | Type | Visibility | Description |
| Fr\_Shipper | id | integer | public | Unique identifier of shipper. |
| Fr\_Store\_id | integer | public | Linked to table of store. |
| introduce | String | public | Introduce of shipper. |
| sum\_revenue | integer | public | Sum Revenue of shipper |
| nation\_id\_front\_image | longblob | public | Front image of Nation Id Card |
| nation\_id\_back\_image | longblob | public | Back image of Nation Id Card |
| bike\_registration\_id | String | public | Number of Bike Registration Card |
| bike\_registration\_date | bigint | public | Date of Bike Registration Card |
| bike\_registrion\_front\_image | longblob | public | Front image of Bike Id Card |
| bike \_ registrion\_back\_image | longblob | public | Back image of Bike Id Card |
| Fr\_Source\_id | integer | public | Linked to table of source. |
| Fr\_Price\_Level\_id | integer | public | Linked to table of price level. |

**Table Shipper Attribute**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Entity | Attribute | Type | Visibility | Description |
| Fr\_source | id | integer | public | Unique identifier of source. |
| name | String | public | Name of source. |
| description | String | public | Description of source. |

**Table Source Attribute**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Entity | Attribute | Type | Visibility | Description |
| Fr\_Store | id | integer | public | Unique identifier of store. |
| store\_name | String | public | Name of store. |
| address | String | public | Description of store. |
| Fr\_District\_id | integer | public | Linked to table of district. |
| longitude | double | public | Longitude of store. |
| latitude | double | public | Latitude of store. |
| store\_image | longblob | public | Image of store. |
| rating | double | public | Rating for store. |
| rating\_count | integer | public | Number of rating for store |
| phone | String | public | Phone of store. |
| create\_time | Bigint | public | Create time of store. |
| update\_time | Bigint | public | Update time of store. |
| delete\_time | Bigint | public | Delete time of store. |
| note | String | public | Note of store. |
| status | integer | public | Status of store. |
| editor\_id | integer | public | Id of editor store. |

**Table Store Attribute**

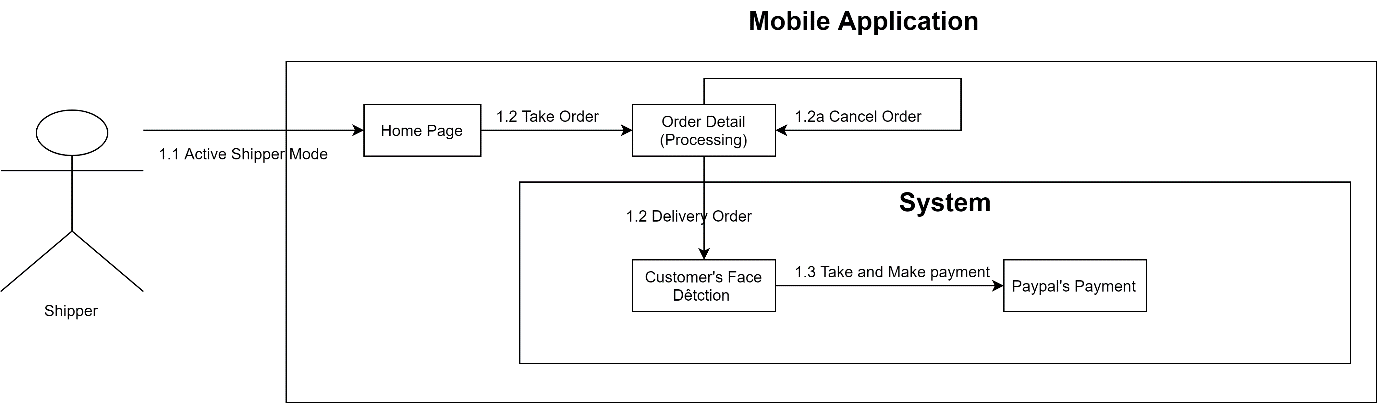
# 3. Performance Measures

## 3.1 Features to be tested

## 3.2 Features not to be tested

# 4. Test Plan

# 5. System Testing Test Case

Figure Admin Core Flow

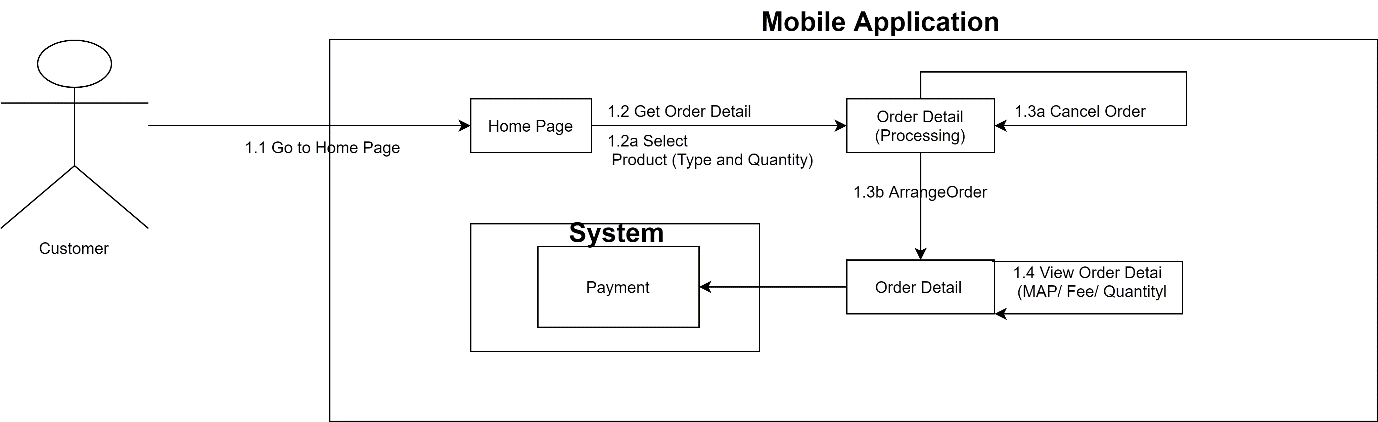
Figure Shipper Core Flow

Figure Customer Core Flow

## 5.1 Admin Test Case

### **5.1.1 Report Event**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Test Case Description** | **Test Case Procedure** | **Expected output** | **Depend** | **Result** | **Date** | **Note** |
| 1 | Homepage | Click to Home Page | Show Homepage on browser | No | Show Homepage on browser | 14/04 |  |
| 2 | Report | Automatic | Show:   * Shipper * Customer * Store * Order * Ratio * Sold Product * Revenue * Paid Shipper | No | Show Automatic | 14/04 |  |

### Table: Report Event

**5.1.2 Statistic Event**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Test Case Description** | **Test Case Procedure** | **Expected output** | **Depend** | **Result** | **Date** | **Note** |
| 1 | Statistic | Select Chart | Show Statistic Chart | **Report Event** | Show Statistic Chart | 14/04/19 |  |

### Table: Statistic Event