|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| |  |  |  | | --- | --- | --- | | **Logo_FPT_University_doc** |  | | | **FPT UNIVERSITY** | | |
| Capstone Project |
| Software Module for Delivery Helper  Software Requirements Specification |
| |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | |  |  |  |  | | --- | --- | --- | --- | | **Group 5** | | | | | **Group Members** | Nguyễn Hoàng Việt Khánh |  | 60566 | | Nguyễn Thị Yên Thịnh |  | 60269 | | Nguyễn Đỗ Vượng |  | 60267 | | An Ngọc Anh |  | 60277 | | **Supervisor** | Lâm Hữu Khánh Phương | | | | **Capstone Project code** | SMDH | | | | |
| * Ho Chi Minh City, Oct / 2012 - |

# Table of Contents

[Table of Contents 2](#_Toc353754554)

[3.1 User Requirement Specification 3](#_Toc353754555)

[3.1.1 Common Features 3](#_Toc353754556)

[3.1.2 Tiktak staff Management 3](#_Toc353754557)

[3.1.3 Item Management 3](#_Toc353754558)

[3.1.4 Hub’s Order Tracking 3](#_Toc353754559)

[3.1.5 Order info checking 4](#_Toc353754560)

[3.1.6 Order Management 4](#_Toc353754561)

[3.2 System Requirement Specification 4](#_Toc353754562)

[3.2.1 External Interface Requirements 4](#_Toc353754563)

[3.2.1.1 User Interfaces 4](#_Toc353754564)

[3.2.1.2 Hardware Interfaces 4](#_Toc353754565)

[3.2.1.3 Software Interfaces 5](#_Toc353754566)

[3.2.1.4 Communications Protocol 5](#_Toc353754567)

[3.2.2 System Features 6](#_Toc353754568)

[3.2.2.1 Manage Plan 7](#_Toc353754569)

[3.2.2.2 Customer 14](#_Toc353754570)

[3.2.2.3 Manage Item 27](#_Toc353754571)

[3.2.2.4 Manage Request (not implement yet) 31](#_Toc353754572)

[3.2.3 Software System Attributes 33](#_Toc353754573)

[3.2.3.2 Reliability 33](#_Toc353754574)

[3.2.3.3 Availability 33](#_Toc353754575)

[3.2.3.4 Security 34](#_Toc353754576)

[3.2.3.5 Maintainability 34](#_Toc353754577)

# User Requirement Specification

## Common Features

* Only authenticated users can access the system (except Shopping Cart page). Users can log in and log out using their own accounts.
* Users can change their password.
* Only authorized users can use specific functions of the system.

## Buyer

* User with this role can view all products and create new orders. After that they have two ways for receiving ordered products
* Direct delivery: products will be delivered directly to address in created order
* Buffer delivery: if buyers don’t have a stable address they can order and receive at a nearly hub
* With buffer delivery, when creating successfully a new order, buyer will get a security pass-code. At the time products are available at ordered hub, buyer can get them with pass-code.
* Besides for created orders, buyer can review them as list by entering registered phone number to show. However to view all detailed information of order, buyer must have pass-code for that order.

## Customer

* Those who have Customer role can approve or reject existed orders
* Importing Buyers’ created orders into system through excel files.
* Adding order to a request, and set it to Tiktak. After Tiktak has set price for order and responded, customer can approve, request again for reset the price or cancel the request.
* When creating a new request, customers have two ways to add an order to request, get an approved order or creating new order as:
* Add new order: creating order with Direct delivery option, customer can create a new product for order without inserting into Database
* Add order to request: creating order follow Buffer delivery option

## Hub Staff

* For each hub, staffs here can get list of order currently in hub and tend to deliver to buyers.
* If delivery time is expired and no one come to get products, order status will change to “expired”, or in case buyer have some problems, does not satisfy with delivered products and want to change another, order status will change to “waiting for return” and wait for Tiktak staff to collect those back to company.
* When buyer comes to hub for receiving ordered products, they must have completed order checking by giving hub staff a correct pass-code before getting products, and order status will change to “delivered”

## Tiktak Staff

* Respond for setting orders’ delivered prices submitted by customer or reset price of order when customer request
* After customer have approved set price, staff will start to create collection plan and assign delivery men on each plan for collecting product back to company
* After completing product collecting, staff continually to plan for delivering product to hubs
* For direct delivery, if failure, product will be bring back to company
* For buffer delivery, if failure, staff will create return plan for collecting products back to company and find some ways to send back to customer or redeliver products

# System Requirement Specification

## External Interface Requirements

### User Interfaces

* The design should be simple and user-friendly. White and dark will be 2 main colors of the website while charts may have more color to visually express data more effectively.
* The menu bar should be on the left and not take too much space of the screen. It consists of the list of main functions of the system that users can access.
* The design should be responsive. It means that the web components should be scaled according to a range of resolutions and devices to provide a consistent experience, no matter what.

### Hardware Interfaces

* To access to the system, users only need any type of computer, tablet, or mobile phone with a fair internet connection.

### Software Interfaces

* At the server side, the system should run on top of Windows 7, Windows Server 2008 or later versions of Windows Server. Besides, Microsoft .NET Framework 4 and MVC 3 should be installed on the server. The database management system use for SMDH is SQL Server 2008 R2.
* At the client side, users can use any modern browser that supports JavaScript and HTML 5 to access to the system.

### Communications Protocol

HTTP is the protocol used for loading the web site in browsers.

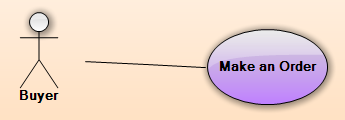
## System Features

Figure 1 - General Use Case Diagram

### Buyer

Figure 2 – Buyer

#### Make an Order

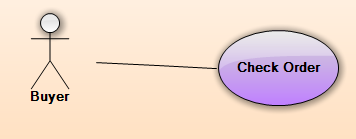


|  |  |  |  |
| --- | --- | --- | --- |
| USE CASE – MAKE AN ORDER | | | |
| Use-case No. | UC01.1 | Use-case Version | 1.0 |
| Use-case Name | Make an order | | |
| Author | KhanhNHV | | |
| Date | 03/02/2013 | Priority | Normal |
| **Actor:** Buyer  **Summary:** This use case is about how to order one or more products  **Goal:** Make an order successful  **Triggers:** N/A  **Pre-conditions:** N/A  **Post-conditions:** Order is created  **Main Success Scenario:**   1. Open Shopping Cart page, choose Product and add to Cart Add box 2. Click Next button 3. Fill in receiver information. 4. Click OK button.   **Alternative Scenario:**  1. User can remove product if necessary.  3. User can click *Back* button to add or remove product in cart.  **Exceptions:** 3. In step 3, if some required fields is not filled in, user click *OK* button, the validation message will be shown.  **Relationships:**  N/A  **Business Rules:** N/A | | | |

#### Receive Passcode

|  |  |  |  |
| --- | --- | --- | --- |
| USE CASE – RECEIVE PASSCODE | | | |
| Use-case No. | UC01.2 | Use-case Version | 1.0 |
| Use-case Name | Receive passcode | | |
| Author | KhanhNHV | | |
| Date | 03/02/2013 | Priority | Normal |
| **Actor:** Buyer  **Summary:** This use case is about how to receive passcode after making an order  **Goal:** Buyer gets passcode after making an order.  **Triggers:** N/A  **Pre-conditions:** User makes an order successful.  **Post-conditions:** User gets passcode  **Main Success Scenario:**   1. After created order, system will show confirm message and passcode.   **Alternative Scenario:**  1. If the order is direct delivery, system will show confirm message only.  **Exceptions:** N/A  **Relationships:**  N/A  **Business Rules:** N/A | | | |

#### Check Order



|  |  |  |  |
| --- | --- | --- | --- |
| USE CASE – CHECK ORDER | | | |
| Use-case No. | UC01.3 | Use-case Version | 1.0 |
| Use-case Name | Check order | | |
| Author | KhanhNHV | | |
| Date | 03/02/2013 | Priority | Normal |
| **Actor:** Buyer  **Summary:** This use case is about how to view order info by Buyer side  **Goal:** View order info successfully  **Triggers:** N/A  **Pre-conditions:** Plan has created successfully.  **Post-conditions:** View order info successfully via phone number and passcode  **Main Success Scenario:**   1. Click *View Information Order* button on Shopping Cart page 2. System will show the page to enter phone number 3. Enter phone number 4. Click *OK* button 5. All orders which has this phone number are displayed in data table 6. Click *Detail* button on order which user wants to view detail 7. Enter passcode 8. Click *OK* button   **Alternative Scenario:**  N/A  **Exceptions:** 3. If user inputs invalid phone number, system will show validate message and ask user to input value again  7**.** If user inputs invalid passcode, system will redirect to error page  **Relationships:**  N/A  **Business Rules:** N/A | | | |

### Customer

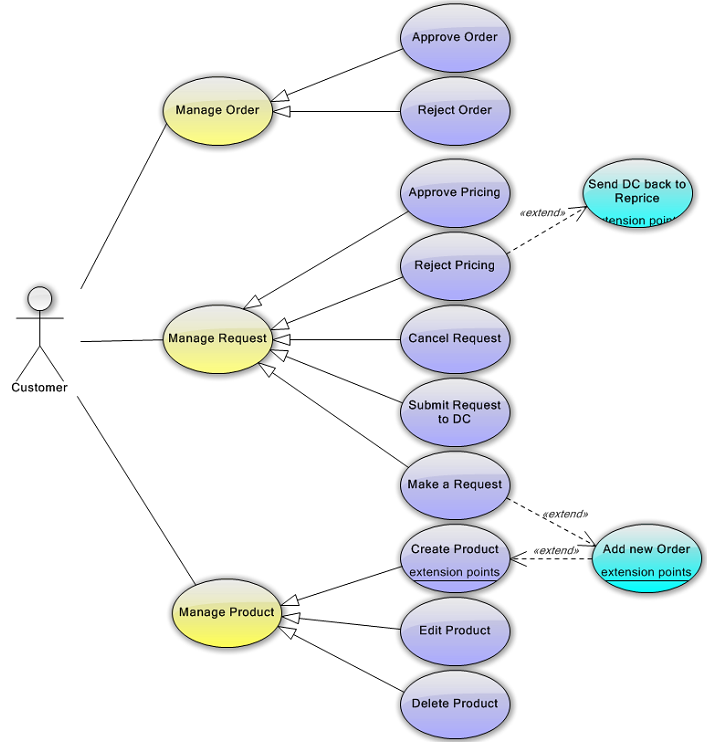


Figure 3 – Customer

#### Make a Request

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| USE CASE – MAKE A REQUEST | | | | |
| Use-case No. | UC02.1 | Use-case Version | | 1.0 |
| Use-case Name | Make a request | | | |
| Author | ThinhNTY | | | |
| Date | 03/02/2013 | Priority | Normal | |
| **Actor:** Customer  **Summary:** This use case is about how customer make a new request  **Goal:** Customer can make a new request successfully  **Triggers:** N/A  **Pre-conditions:**  N/A  **Post-conditions:** The request is created successfully.  **Main Success Scenario:**   1. In Waiting for Price page of customer, user clicks “Add” button. 2. System will appear a popup to add order to request. 3. User add existing order to request and clicks “Create” button. 4. New request has been created with Draft status.   **Alternative Scenario:** In step 3, user has also 2 ways to add order to request.   1. Add new Order, it means order is Direct Delivery type. 2. Add Order to Request, it means order is Buffer type.   **Exceptions:**  **Relationships:**  N/A  **Business Rules:** N/A | | | | |

#### Add new Order

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| USE CASE – Add new Order | | | | |
| Use-case No. | UC02.2 | Use-case Version | | 1.0 |
| Use-case Name | Add new Order | | | |
| Author | ThinhNTY | | | |
| Date | 03/02/2013 | Priority | Normal | |
| **Actor:** Customer  **Summary:** This use case is about how to add new order  **Goal:** Customer can add new order successfully  **Triggers:**  **Pre-conditions:** User logged in with customer role.  **Post-conditions:** Add new Order when make a request successful  **Main Success Scenario:**  1. In Waiting for Price page of Customer, Click Add button and then, click Add new Order button  2. System will show a pop-up including textbox related to product info, buyer info.  3. Input product info, buyer info and then, click Create button.  **Alternative Scenario:** 2. If the product is not existed in database, user can add (or create) product to make an order.  **Exceptions:** N/A  **Relationships:**  N/A  **Business Rules:** N/A | | | | |

#### Create Product

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| USE CASE – CREATE PRODUCT | | | | |
| Use-case No. | UC02.3 | Use-case Version | | 1.0 |
| Use-case Name | CREATE PRODUCT | | | |
| Author | ThinhNTY | | | |
| Date | 03/02/2013 | Priority | Normal | |
| **Actor:** Customer  **Summary:** This use case is about how to change status create new product  **Goal:** User can create product successfully.  **Triggers:**  **Pre-conditions:** User logged in with customer role.  **Post-conditions:** New product is created successfully.  **Main Success Scenario:**   1. In main page of product management, user click *Add* button. 2. System will show Create pop-up screen. 3. User input valid fields and click Create button. 4. New product is created and save to database. System will be redirected to product management page, and the product has just created will be shown in the product table as first record.   **Alternative Scenario:** In step 1, user create product by making a request with Add new order type. Product is saved to database when user click on check box isPermanent. If not, product just use temporary for that order.  **Exceptions:** If user input some invalid fields, and click Create button. System will show the validation message  **Relationships:**  N/A  **Business Rules:** N/A | | | | |

#### 3.2.2.2.4 Edit Product

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| USE CASE – EDIT PRODUCT | | | | |
| Use-case No. | UC02.4 | Use-case Version | | 1.0 |
| Use-case Name | Edit product | | | |
| Author | AnhAN | | | |
| Date | 03/02/2013 | Priority | Normal | |
| **Actor:** Customer  **Summary:** This use case is about how to modify an existing product in system.  **Goal:** User can modify product and update to database successfully.  **Triggers:** N/A  **Pre-conditions:** User must log in as customer role  **Post-conditions:** Product is modified and update to database successful  **Main Success Scenario:**   1. In product management page, user click on Edit button of the product user want to modify. 2. Edit pop-up screen is displayed and User change value of some fields. 3. User clicks *Save* button. 4. System redirect to product management page, that product is modified and it has updated to database also.   **Alternative Scenario:** In step 3, user click cancel button. System will redirect to product management page and nothing is changed.  **Exceptions:**  **Relationships:**  N/A  **Business Rules:** N/A | | | | |

#### Delete product

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| USE CASE – DELETE PRODUCT | | | | |
| Use-case No. | UC02.5 | Use-case Version | | 1.0 |
| Use-case Name | Delete product | | | |
| Author | AnhAN | | | |
| Date | 03/02/2013 | Priority | Normal | |
| **Actor:** Customer  **Summary:** This use case is about how to delete a product.  **Goal:** User can delete product and update to database successfully.  **Triggers:** N/A  **Pre-conditions:** User must log in as customer role  **Post-conditions:** Product is deactive and update to database successful  **Main Success Scenario:**   1. In product management page, user click on Delete button of the product user want to modify. 2. The confirm message is displayed 3. User clicks *OK* button. 4. That product is deactive and update to database. That product is not used in progress.   **Alternative Scenario:** In step 3, user click cancel button. System will redirect to product management page and nothing is changed.  **Exceptions:** If the product has not used in any orders, product is removed from database.  **Relationships:**  N/A  **Business Rules:** N/A | | | | |

#### 3.2.2.2.6 Submit Request to tiktak staff

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| USE CASE – SUBMIT REQUEST TO TIKTAK | | | | |
| Use-case No. | UC02.6 | Use-case Version | | 1.0 |
| Use-case Name | Submit request to tiktak | | | |
| Author | AnhAN | | | |
| Date | 03/02/2013 | Priority | Normal | |
| **Actor:** Customer  **Summary:** This use case is about how to submit a request to tiktak.  **Goal:** User can submit a request successfully.  **Triggers:** N/A  **Pre-conditions:** User must log in as customer role  **Post-conditions:** Tiktak staff can see request to start setting pricing.  **Main Success Scenario:**   1. In Waiting for Price page, user click on request and click Submit button. 2. That request will be changed status to New.   **Alternative Scenario:**  **Exceptions:** If the request is not including order, error message must be displayed.  **Relationships:**  N/A  **Business Rules:** N/A | | | | |

#### 3.2.2.2.7 Approve pricing

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| USE CASE – APPROVE PRICING | | | | |
| Use-case No. | UC02.7 | Use-case Version | | 1.0 |
| Use-case Name | Approve pricing | | | |
| Author | AnhAN | | | |
| Date | 03/02/2013 | Priority | Normal | |
| **Actor:** Customer  **Summary:** This use case is about how to approve price after tiktak set price for each order of request.  **Goal:** User can approve price if the price is ok.  **Triggers:** N/A  **Pre-conditions:** User must log in as customer role  **Post-conditions:** All orders in request have price is accepted.  **Main Success Scenario:**   1. In *Need to approve request* page of customer, user can click *Action* dropdown list to choose Approve action. 2. That request is removed from this table and change status to Approved.   **Alternative Scenario:**  **Exceptions:**  **Relationships:**  N/A  **Business Rules:** N/A | | | | |

#### 3.2.2.2.8 Reject Price

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| USE CASE – REJECT PRICING | | | | |
| Use-case No. | UC02.8 | Use-case Version | | 1.0 |
| Use-case Name | Reject pricing | | | |
| Author | AnhAN | | | |
| Date | 03/02/2013 | Priority | Normal | |
| **Actor:** Customer  **Summary:** This use case is about how to reject price after tiktak set price for each order of request.  **Goal:** User can reject price if the price is not fine.  **Triggers:** N/A  **Pre-conditions:** User must log in as customer role  **Post-conditions:** Order in request have price is denied.  **Main Success Scenario:**   1. In *Need to approve request* page of customer, user can click *Action* dropdown list to choose Cancel/ Repricing action. 2. That request is removed from this table and change status to Cancel/Repricing.   **Alternative Scenario:** In step 2, if user click Repricing action, please refer to TC 3.2.2.9  **Exceptions:**  **Relationships:**  N/A  **Business Rules:** N/A | | | | |

#### 3.2.2.2.9 Send Delivery Company back to reprice

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| USE CASE – SEND DC BACK TO REPRICE | | | | |
| Use-case No. | UC02.9 | Use-case Version | | 1.0 |
| Use-case Name | Send DC back to reprice | | | |
| Author | AnhAN | | | |
| Date | 03/02/2013 | Priority | Normal | |
| **Actor:** Customer  **Summary:** This use case is about how to re-price after Delivery company set price for each order of request.  **Goal:** User can require tiktak set price again if the price is not fine.  **Triggers:** N/A  **Pre-conditions:** User must log in as customer role  **Post-conditions:** Request is sent tiktak back to set price again.  **Main Success Scenario:**   1. In *Need to approve request* page of customer, user can click *Action* dropdown list to choose Request *Repricing* action. 2. That request is removed from this table and change status to *Repricing*. 3. That request has new price after pricing and send back to customer.   **Alternative Scenario:**  **Exceptions:**  **Relationships:**  N/A  **Business Rules:** N/A | | | | |

#### 3.2.2.2.10 Cancel Request

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| USE CASE – CANCEL REQUEST | | | | |
| Use-case No. | UC02.10 | Use-case Version | | 1.0 |
| Use-case Name | Cancel request | | | |
| Author | AnhAN | | | |
| Date | 03/02/2013 | Priority | Normal | |
| **Actor:** Customer  **Summary:** This use case is about how to cancel request  **Goal:** User can cancel request successful  **Triggers:** N/A  **Pre-conditions:** User must log in as customer role  **Post-conditions:** Request is canceled  **Main Success Scenario:**   1. In *Need to approve request* page of customer, user can click *Action* dropdown list to choose Request *Repricing* action. 2. That request is removed from this table and change status to *Repricing*. 3. That request has new price after pricing and send back to customer.   **Alternative Scenario:**  **Exceptions:** N/A  **Relationships:**  N/A  **Business Rules:** N/A | | | | |

#### 3.2.2.2.11 Approve Order

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| USE CASE – APPROVE ORDER | | | | |
| Use-case No. | UC02.11 | Use-case Version | | 1.0 |
| Use-case Name | Approve Order | | | |
| Author | AnhAN | | | |
| Date | 03/02/2013 | Priority | Normal | |
| **Actor:** Customer  **Summary:** This use case is about how to approve order which is created by buyer  **Goal:** User can approve order  **Triggers:** N/A  **Pre-conditions:** User must log in as customer role  **Post-conditions:** Order is approved  **Main Success Scenario:**   1. In *Dashboard* of customer, user click *Need to Approve* box to review and approve order which has created by buyer. 2. Click on *Action* dropdown list and choose Approve action. 3. Order will be removed from table and change status to Approved.   **Alternative Scenario:**  **Exceptions:** 2. User can choose View to see order detail before rejected.  **Relationships:**  N/A  **Business Rules:** N/A | | | | |

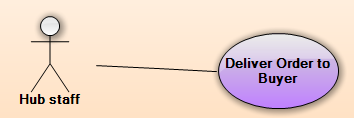
#### Reject Order

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| USE CASE – REJECT ORDER | | | | |
| Use-case No. | UC02.12 | Use-case Version | | 1.0 |
| Use-case Name | Reject Order | | | |
| Author | AnhAN | | | |
| Date | 03/02/2013 | Priority | Normal | |
| **Actor:** Customer  **Summary:** This use case is about how to reject order which is created by buyer  **Goal:** User can reject order  **Triggers:** N/A  **Pre-conditions:** User must log in as customer role  **Post-conditions:** Order is rejected  **Main Success Scenario:**   1. In *Dashboard* of customer, user click *Need to Approve* box to review and reject order which has created by buyer. 2. Click on *Action* dropdown list and choose Reject action. 3. Order will be removed from table and change status to Rejected.   **Alternative Scenario:** 2. User can choose View to see order detail before rejected.  **Exceptions:** N/A  **Relationships:**  N/A  **Business Rules:** N/A | | | | |

### Hub staff

Figure 4 – Hub staff

#### Deliver Order to Buyer



|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| USE CASE – DELIVER ORDER TO BUYER | | | | |
| Use-case No. | UC03.1 | Use-case Version | | 1.0 |
| Use-case Name | Deliver order to buyer | | | |
| Author | VuongND | | | |
| Date | 03/02/2013 | Priority | Normal | |
| **Actor:** Hub staff  **Summary:** This use case is about how hub staff deliver order to buyer.  **Goal:** User can check order and deliver to buyer successful  **Triggers:** N/A  **Pre-conditions:** User logged in with hub staff role.  **Post-conditions:** Check order info and deliver to buyer  **Main Success Scenario:**   1. In main page of Hub, user input passcode that buyer provides passcode to user 2. System shows all orders which have this passcode. 3. User enters buyer’s phone number into search box 4. System shows order suitable for 2 above conditions. 5. Click Customer Received button. 6. Order status will be changed to Delivered.   **Alternative Scenario:** 5. If buyer is not satisfied with product, user click Customer rejected and order status changes to Waiting to Return.  **Exceptions:** N/A  **Relationships:**  N/A  **Business Rules:** N/A | | | | |

#### View Order

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| USE CASE – VIEW ORDER AT HUB | | | | |
| Use-case No. | UC03.2 | Use-case Version | | 1.0 |
| Use-case Name | View Order at Hub | | | |
| Author | VuongND | | | |
| Date | 03/02/2013 | Priority | Normal | |
| **Actor:** Hub staff  **Summary:** This use case is about how to view order at hub  **Goal:** User can view order at hub  **Triggers:** N/A  **Pre-conditions:** User logged in with hub staff role.  **Post-conditions:** View order at hub by status successfully  **Main Success Scenario:**   1. In main page of Hub, user clicks tab to view order.   **Alternative Scenario:** There are 5 statuses that user can view at hub.  **Exceptions:** N/A  **Relationships:**  N/A  **Business Rules:** N/A | | | | |

### Tiktak Staff

Figure 5 – Tiktak staff

#### 3.2.2.4.1 Set price

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| USE CASE – SET PRICE | | | | |
| Use-case No. | UC04.1 | Use-case Version | | 1.0 |
| Use-case Name | Set price | | | |
| Author | ThinhNTY | | | |
| Date | 03/02/2013 | Priority | Normal | |
| **Actor:** Tiktak staff  **Summary:** This use case is about how to set price for order  **Goal:** Customer can create a new order successfully  **Triggers:** N/A  **Pre-conditions:**  User must be logged in with staff role  **Post-conditions:** All orders in request has set price successfully  **Main Success Scenario:**   1. In Dashboard of staff, user clicks Request tab. 2. System will show all request in database. 3. User clicks Detail button on Order. 4. Choose price type in dropdown list. 5. Click add button. 6. Click Close button   **Alternative Scenario:** 5. User can choose more than one price type for each order.  **Exceptions:** N/A  **Relationships:**  N/A  **Business Rules:** N/A | | | | |

#### 3.2.2.4.2 Repricing

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| USE CASE – REPRICING | | | | |
| Use-case No. | UC04.2 | Use-case Version | | 1.0 |
| Use-case Name | Repricing | | | |
| Author | ThinhNTY | | | |
| Date | 03/02/2013 | Priority | Normal | |
| **Actor:** Tiktak staff  **Summary:** This use case is about how to repricing  **Goal:** Customer can create a new order successfully  **Triggers:** N/A  **Pre-conditions:**  N/A  **Post-conditions:** The order is created successfully.  **Main Success Scenario:**   1. In main page of customer, user clicks “Add” button. 2. System will appear a popup to input information. 3. User inputs request information and clicks “Create” button. 4. New customer appears in customer list.   **Alternative Scenario:** In step 3, if user wants to cancel the process, user can clicks to “Cancel” button to back the main page of customer.  **Exceptions:** When inputs require information is invalid format or missed require information, the system will notify and require input again.  **Relationships:**  N/A  **Business Rules:** N/A | | | | |

#### 3.2.2.4.3 Make an Auto Collection Plan

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| USE CASE – MAKE AN AUTO COLLECTION PLAN | | | | |
| Use-case No. | UC04.1 | Use-case Version | | 1.0 |
| Use-case Name | Make an auto collection plan | | | |
| Author | ThinhNTY | | | |
| Date | 03/02/2013 | Priority | Normal | |
| **Actor:** Customer  **Summary:** This use case is about how staff create an auto collection plan  **Goal:** Customer can create an auto collection plan successfully  **Triggers:** N/A  **Pre-conditions:**  User must log in as Staff role  **Post-conditions:** Collection plan is automatically created successful  **Main Success Scenario:**   1. In main page of staff, user clicks “PLANS” tab and choose “Collection Plan”. 2. System will appear “Create Auto Collection Plan” page. 3. User inputs number of plan, add request and click “Solve” button. 4. Plan is created temporary. Click “Save” button. 5. Plan is created and saved to database.   **Alternative Scenario:** In step 3, if user wants to back to “Create Auto Collection Plan”, click on “Back to Create Collection Plan”  **Exceptions:** In step 3, if user don’t input number of plan or add request, when click “Solve” button, the validation message is displayed.  **Relationships:**  N/A  **Business Rules:** N/A | | | | |

#### Cancel Plan

|  |  |  |  |
| --- | --- | --- | --- |
| USE CASE – CANCEL PLAN | | | |
| Use-case No. | UC04.5 | Use-case Version | 1.0 |
| Use-case Name | Cancel Plan | | |
| Author | KhanhNHV | | |
| Date | 03/02/2013 | Priority | Normal |
| **Actor:** Tiktak staff  **Summary:** This use case is about how the plan is canceled  **Goal:** Plan is canceled successfully  **Triggers:** N/A  **Pre-conditions:** Plan has created successfully.  **Post-conditions:** Plan is canceled successfully  **Main Success Scenario:**   1. In Index page of staff, Click *Cancel* button on plan which wants to be canceled. 2. System will show confirm message. 3. Click “OK” button. 4. Plan is changed to “Canceled” status   **Alternative Scenario:**  N/A  **Exceptions:** N/A  **Relationships:**  N/A  **Business Rules:** N/A | | | |

#### Assign Delivery Men

|  |  |  |  |
| --- | --- | --- | --- |
| USE CASE – ASSIGN DELIVERY MEN | | | |
| Use-case No. | UC04.6 | Use-case Version | 1.0 |
| Use-case Name | Assign delivery men | | |
| Author | KhanhNHV | | |
| Date | 03/02/2013 | Priority | Normal |
| **Actor:** Tiktak staff  **Summary:** This use case is about how the tiktak staff assigns delivery men to plan  **Goal:** Assign available delivery men to plan  **Triggers:** N/A  **Pre-conditions:** Plan has created successfully.  **Post-conditions:** Assign delivery men to plan successful.  **Main Success Scenario:**   1. Click *View&Assign* button on plan which wants to assign. 2. System will show “Assign Delivery Men” popup 3. Check the DM(s) who will implement this plan. 4. Click *Assign* button   **Alternative Scenario:**  3. If the plan has been assigned, check the DM(s) and click “Reassign” button  **Exceptions:** N/A  **Relationships:**  N/A  **Business Rules:** N/A | | | |

#### Mark as Finished

|  |  |  |  |
| --- | --- | --- | --- |
| USE CASE – MARK AS FINISHED | | | |
| Use-case No. | UC04.7 | Use-case Version | 1.0 |
| Use-case Name | Mark as Finished | | |
| Author | KhanhNHV | | |
| Date | 03/02/2013 | Priority | Normal |
| **Actor:** Tiktak staff  **Summary:** This use case is about how to finish a plan  **Goal:** Plan is finished  **Triggers:** N/A  **Pre-conditions:** Plan has created and assigned already.  **Post-conditions:** Assign plan to delivery men successful.  **Main Success Scenario:**   1. Click *View&Assign* button on plan which wants to finish. 2. System will show “Assign Delivery Men” pop up. 3. Click “Mark as Finished” button   **Alternative Scenario:**  N/A  **Exceptions:** N/A  **Relationships:**  N/A  **Business Rules:** N/A | | | |

#### Make an auto Delivery Plan

|  |  |  |  |
| --- | --- | --- | --- |
| USE CASE – MAKE AN AUTO DELIVERY PLAN | | | |
| Use-case No. | UC04.8 | Use-case Version | 1.0 |
| Use-case Name | Make an auto Delivery Plan | | |
| Author | KhanhNHV | | |
| Date | 03/02/2013 | Priority | Normal |
| **Actor:** Tiktak staff  **Summary:** This use case is about how to make an auto delivery plan  **Goal:** Delivery plan is created successfully  **Triggers:** N/A  **Pre-conditions:** User log in with Tiktak staff role  **Post-conditions:** Delivery plan is created successfully  **Main Success Scenario:**   1. In the Delivery Plan page, enter number of plan, add order to make a plan 2. Click “Solve” button 3. Plan is created temporary. Click “Save” button 4. Plan is really created and saved to database   **Alternative Scenario:**  3. If user wants to make another plan without saving this plan, click “Back to Create Delivery Plan” button  **Exceptions:** 2a. If number of plan is empty or invalid. System will show validate message  2b. If user don’t add order to Cart Add, system will show validate message  **Relationships:**  N/A  **Business Rules:** N/A | | | |

#### Make an auto Return Plan

|  |  |  |  |
| --- | --- | --- | --- |
| USE CASE – MAKE AN AUTO RETURN PLAN | | | |
| Use-case No. | UC04.8 | Use-case Version | 1.0 |
| Use-case Name | Make an auto Return Plan | | |
| Author | KhanhNHV | | |
| Date | 03/02/2013 | Priority | Normal |
| **Actor:** Tiktak staff  **Summary:** This use case is about how to make an auto return plan  **Goal:** Return plan is created successfully  **Triggers:** N/A  **Pre-conditions:** User log in with Tiktak staff role  **Post-conditions:** Return plan is created successfully  **Main Success Scenario:**   1. In the Return Plan page, enter number of plan, add order to make a plan 2. Click “Solve” button 3. Plan is created temporary. Click “Save” button 4. Plan is really created and saved to database   **Alternative Scenario:**  3. If user wants to make another plan without saving this plan, click “Back to Create Delivery Plan” button  **Exceptions:** 2a. If number of plan is empty or invalid. System will show validate message  2b. If user don’t add order to Cart Add, system will show validate message  **Relationships:**  N/A  **Business Rules:** N/A | | | |

## Software System Attributes

### Reliability

* The database can be backed up easily and recovered if necessary.
* The system should never crash or hang, except for the cause from an operating system or network error.
* Mean Time Between Failures (MTBF): The acceptable failure is once a year. The failure should not be because of the database, or else the data may be lost and cannot be recovered.
* Mean Time To Repair (MTTR): When the failure occurs, it should take as little time as possible to repair. The acceptable mean time for a particular failure must be less than 8 hours.
* Maximum Bugs or Defect Rate: 05 bugs / KLOC

### Availability

* The uptime percentage should be at least 99.95%.
* The acceptable time for maintenance or backup should not be more than 8 hours per month.
* When the system goes in under-maintenance, the website should display a message to inform that.

### Security

* Only users with proper account can access certain information of the system. All the information of users must not be available for anyone or software that is not part of the system. User password is also encrypted and not available to the system administrators.

### Maintainability

* The code must follow C#.NET coding and naming convention.
* There should be comments in code files that explain the functions of each code segment.
* All the errors should be logged, which supports bug fixing and maintenance.

# Entity Relationship Diagram