**Web-based Ordering &Ingredient Estimating for Bakery Manufacturer**

Software Design Document

By

**Miss. NontraMahachanont 542115026**

**Mr. ParinyaPanyanak 542115034**

Department of Software Engineering

College of Arts, Media and Technology

Chiang Mai University

Project Advisor

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Mr.PhudinanSingkhamfu**

|  |  |  |
| --- | --- | --- |
| **Software Requirement Specification** | | |
| Cross Ref. VSE-29110 | **Coverage Level:** | **Version:** |
|  | V.1.0 | V.1.0 |

|  |  |
| --- | --- |
| **Process Ownership** | **Approving Authority** |
| Web-based Ordering & Ingredient Estimating for Bakery Manufacturer | Aj.Phudinan Singkhamfu |
| **Scope** | **Approved Date** |
| Senior Project | 23 July 2014 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Version Number** | **Record Date** | **Prepared/**  **Modified By** | **Reviewed By** | **Change Details** |
| **Documents History** | | | | |
| V.0.1 | 06-05-2014 | NM, PP | NM, PP | - Add Chapter 1  - Add Chapter 2 |
| V.0.2 | 07-05-2014 | NM, PP | NM, PP | - Add Chapter 3  - Fix inappropriate words |
| V.0.4 | 09-05-2014 | NM, PP | NM, PP | - Add Chapter 4  - Fix inappropriate words |
| V.0.6 | 12-05-2014 | NM, PP | NM, PP | - Adjust Chapter 2  - Fix inappropriate words |
| V.1.0 | 24-06-2014 | NM, PP | NM, PP | - Adjust Chapter 2  - Adjust Chapter 3  - Adjust Chapter 4  - Fix inappropriate words |

**\*NM = NontraMahachanont**

**\*PP = ParinyaPanyanak**

**\*PS = PhudinanSingkhamfu**

**Table of contents**

[**Chapter One | System Architecture** 6](#_Toc396897368)

[**Chapter Two |Detailed Design** 9](#_Toc396897369)

[2.1 Class Diagram 9](#_Toc396897370)

[2.2 Class Diagram Description 10](#_Toc396897371)

[2.2.1 Controller Class 10](#_Toc396897372)

[2.2.2 Model Class 14](#_Toc396897373)

[2.2.3 ServiceClass 18](#_Toc396897375)

[2.3 Sequence Diagram 19](#_Toc396897376)

[2.3.1 Admin can activate member 19](#_Toc396897377)

[2.3.2 User can login 20](#_Toc396897378)

[2.3.3 User can logout 21](#_Toc396897379)

[2.3.4 Admin can edit user information 22](#_Toc396897380)

[2.3.5 Admin can delete user 23](#_Toc396897381)

[2.3.6 Admin can view user information 24](#_Toc396897382)

[2.3.7 Admin can create bakery product 25](#_Toc396897383)

[2.3.8 Admin can edit bakery product information 26](#_Toc396897384)

[2.3.9 Admin can delete bakery product 27](#_Toc396897385)

[2.3.10 Usercan view bakery productlist 28](#_Toc396897386)

[**Chapter Three | Data Architecture** 29](#_Toc396897387)

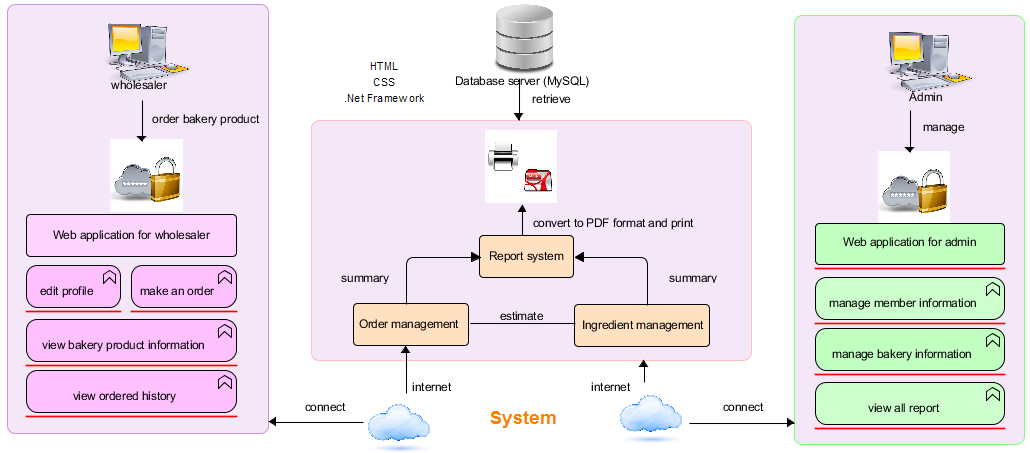
[3.1 ER Diagram 29](#_Toc396897388)

[3.2 ER Diagram Description 30](#_Toc396897389)

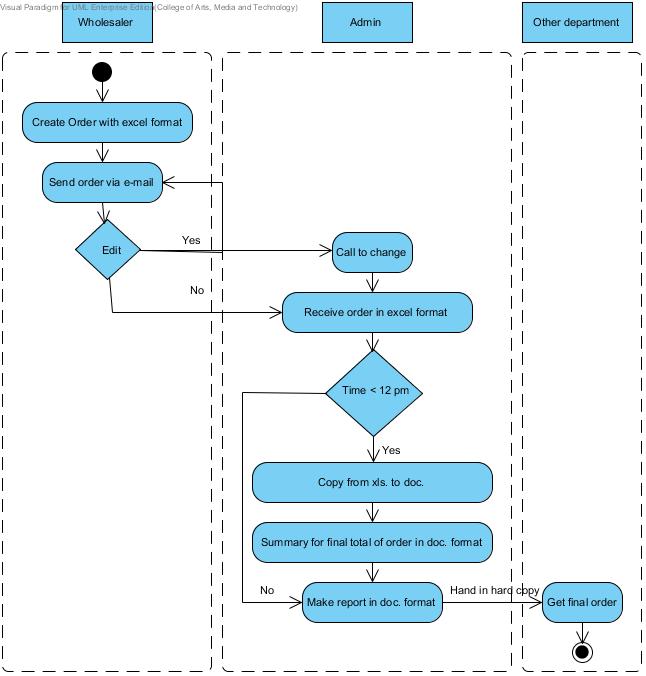
[\* Only database in Progress Report I 31](#_Toc396897390)

[**Chapter Four | User Interface** 32](#_Toc396897391)

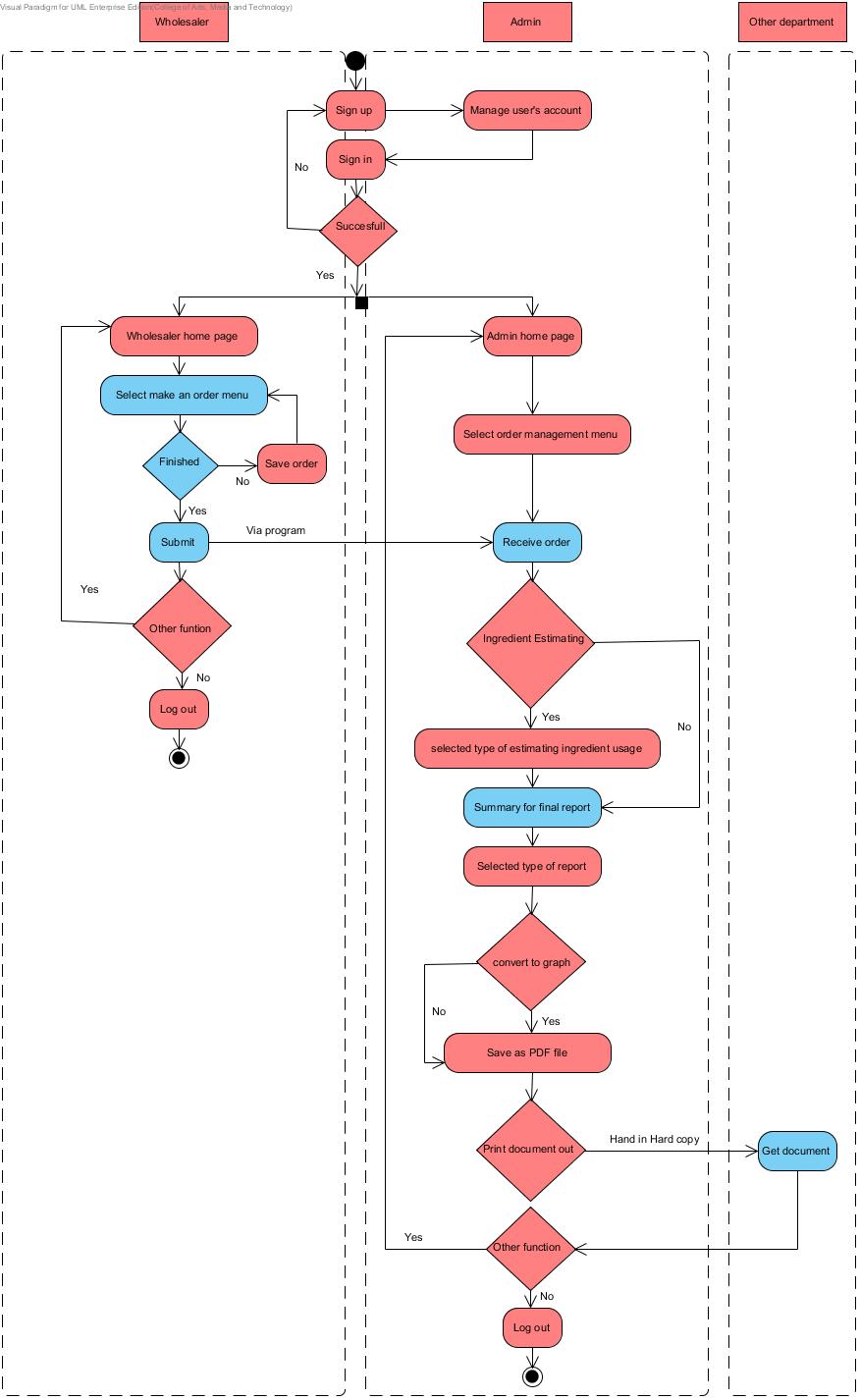
# **Chapter One |** **System Architecture**



**Figure 1: Web-based Ordering & Ingredient Estimating for Bakery Manufacturer**



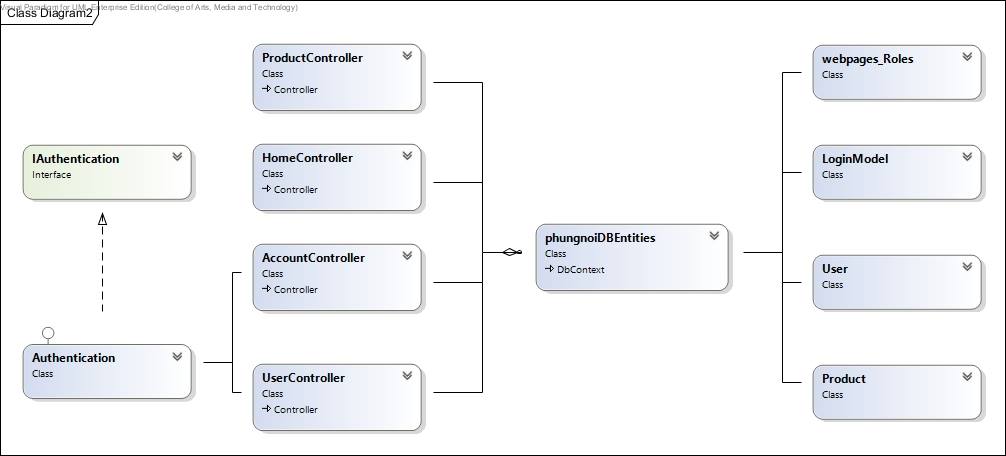
**Figure 2: Flow chart of original workflow**



**Figure 3: Flow chart of Web-based Ordering & Ingredient Estimating for Bakery Manufacturer**

# **Chapter Two |Detailed Design**

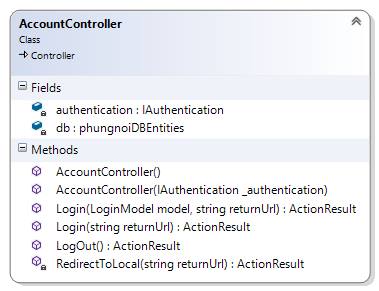
## Class Diagram



**Figure 4: Class Diagram**

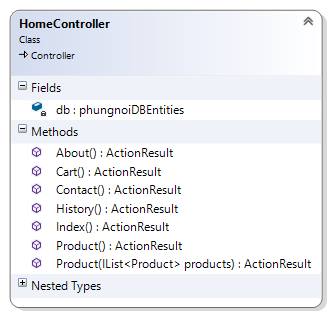
## Class Diagram Description

### 2.2.1 Controller Class



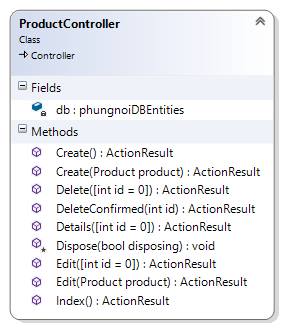
**Method:**

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Name** | **Description** | **Remark** |
| 1 | Login() | This method uses for logging in to the system. | **Return Type**: ActionResult  **Parameters:**model; LoginModel, returnUrl;String |
| 2 | Logout() | This method uses for logging out from the system. | **Return Type**: ActionResult |



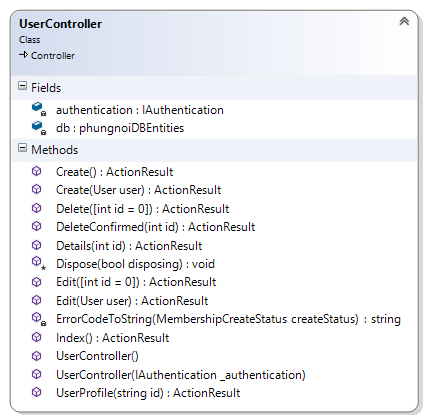
**Method:**

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Name** | **Description** | **Remark** |
| 1 | About() | This method uses to call the About page represent. | **Return Type**: ActionResult |
| 2 | Cart() | This method uses to call the Cart page represent. | **Return Type**: ActionResult |
| 3 | Contact() | This method uses to call the Contact page represent. | **Return Type**: ActionResult |
| 4 | History() | This method uses to call the History page represent. | **Return Type**: ActionResult |
| 5 | Index() | This method uses to call the Index page represent. | **Return Type**: ActionResult |
| 6 | Product() | This method uses to call the Product page represent that will be shown list of product. | **Return Type**: ActionResult  **Parameters:** products; List<Product> |



**Method:**

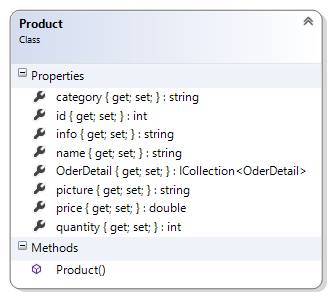
|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Name** | **Description** | **Remark** |
| 1 | Create() | This method uses to call the Create page represent. | **Return Type**: ActionResult |
| 2 | Create() | This method used to add the product in to database. | **Return Type**: ActionResult  **Parameters:**product; Product |
| 3 | Delete() | This method uses to call the Delete page represent. | **Return Type**: ActionResult  **Parameters:**id = 0; int |
| 4 | DeleteConfirmed() | This method uses to delete the product into database**.** | **Return Type**: ActionResult  **Parameters:**id; int |
| 5 | Edit() | This method uses to call the Edit page represent. | **Return Type**: ActionResult  **Parameters:**id=0;int |
| 6 | Edit() | This method used to update the product information into database. | **Return Type**: ActionResult  **Parameters:**producr;Product |
| 7 | Details() | This method uses to call the details of the product. | **Return Type**: ActionResult  **Parameters:**int id = 0 |
| 8 | Index() | This method uses to call the index page represent. | **Return Type**: ActionResult |



**Method:**

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Name** | **Description** | **Remark** |
| 1 | Create() | This method uses to call the Create page represent. | **Return Type**: ActionResult |
| 2 | Create() | This method used to add the product in to database. | **Return Type**: ActionResult  **Parameters:** user; User |
| 3 | Delete() | This method uses to call the Delete page represent. | **Return Type**: ActionResult  **Parameters:**id = 0; int |
| 4 | DeleteConfirmed() | This method uses to delete the product into database**.** | **Return Type**: ActionResult  **Parameters:**id; int |
| 5 | Edit() | This method uses to call the Edit page represent. | **Return Type**: ActionResult  **Parameters:**id=0;int |
| 6 | Edit() | This method used to update the product information into database. | **Return Type**: ActionResult  **Parameters:** user;User |
| 7 | Details() | This method uses to call the details of the product. | **Return Type**: ActionResult  **Parameters:**int id = 0 |
| 8 | Index() | This method uses to call the index page represent. | **Return Type**: ActionResult |
| 9 | UserProfile() | This method uses to call the User Profile page represent. | **Return Type**: ActionResult  **Parameters:**int id = 0 |

### 2.2.2 Model Class



**Property:**

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Name** | **Description** | **Remark** |
| 1 | Category | This properties uses for storing the Product category into the database. | **Type**: String |
| 2 | Id | This properties uses for storing the product id into the database. | **Type**: Int |
| 3 | info | This properties uses for storing the product information into the database. | **Type**: String |
| 4 | name | This properties uses for storing the product name into the database. | **Type**: String |
| 5 | picture | This properties uses for storing the product picture into the database. | **Type**: String |
| 6 | price | This properties uses for storing the product price into the database. | **Type**: String |

**Method:**

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Name** | **Description** | **Remark** |
| 1 | Product() | This method is a constructor of Product. | **Return Type**: ActionResult |

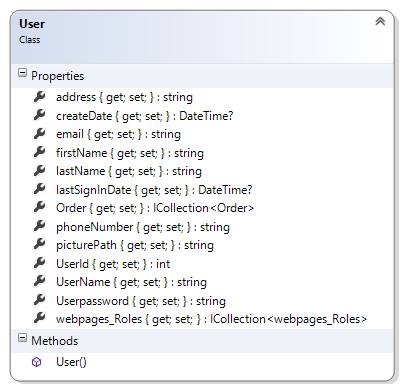


**Property:**

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Name** | **Description** | **Remark** |
| 1 | RoleId | This property uses for separating the role of user from the database. | **Type**: Int |
| 2 | RoleName | This property uses for separating the name of user into the database. | **Type**: String |
| 3 | UserProfile | This property uses for storing the product information into the database. | **Type**: Collection <User> |

**Method:**

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Name** | **Description** | **Remark** |
| 1 | Webpage\_Roles() | This method is a constructor of Webpage\_Roles. | **Return Type**: ActionResult |



**Property:**

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Name** | **Description** | **Remark** |
| 1 | address | This property uses for storing the address of user into the database. | **Type**:String |
| 2 | CreateDate | This property uses for storing the date of activated user into the database. | **Type**: DateTime |
| 3 | firstName | This property uses for storing the first name of user into the database. | **Type**: String |
| 4 | lastName | This property uses for storing the last name of user into the database. | **Type**: String |
| 5 | lastSignInDate | This property uses for storing the last sign in date of user into the database. | **Type**: DateTime |
| 6 | phoneNumber | This property uses for storing the phone number of user into the database. | **Type**: String |
| 7 | picturePath | This property uses for storing the picture of user into the database. | **Type**: String |
| 8 | userId | This property uses for storing the id of user into the database. | **Type**: Int |
| 9 | userName | This property uses for storing the email of user into the database. | **Return Type**: String |
| 10 | userPassword | This property uses for storing the password of user into the database. | **Return Type**: String |
| 11 | Webpage\_roles | This property uses for storing the role of user into the database. | **Return Type**: Collection <Webpage\_roles> |

**Method:**

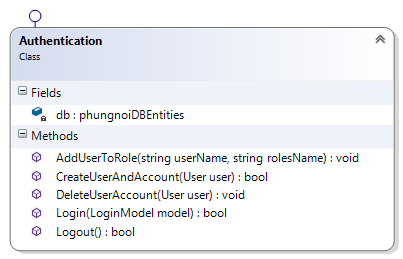
|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Name** | **Description** | **Remark** |
| 1 | User() | This method is a constructor of User. | **Return Type**: ActionResult |



**Property:**

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Name** | **Description** | **Remark** |
| 1 | Password | This property uses for storing the password of user into the database. | **Return Type**: String |
| 2 | UserName | This property uses for storing the username into the database. | **Return Type**: String |
| 3 | RememberMe | This property uses for storing the login into the database. | **Return Type**: Boolean |

### 2.2.3 ServiceClass



**Method:**

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Name** | **Description** | **Remark** |
| 1 | AddUserToRole() | This method uses to adding a role of user. | **Return Type**:void  **Parameters:**username ;string,rolesName; string |
| 2 | CreateUserAndAccount() | This method uses to creating a new user. | **Return Type**:Boolean  **Parameters:**user; User |
| 3 | DeleteUserAccount() | This method uses to deleting a user from database. | **Return Type**:void  **Parameters:**user; User |
| 4 | Login() | This method uses to logging in before using all functional website. | **Return Type**:Boolean  **Parameters:**model; LoginModel |
| 5 | Logout() | This method uses to logging a user out from the website. | **Return Type**:Boolean |

## 2.3 Sequence Diagram

### 2.3.1 Admin can activate member



**Figure 5: SD-01 Admin can activate user**

First,Admincreates user by input the information and submits to the controller. Then, the controller will validate the existing of the user and the model state availability. If the information passed those validations. It will assign the role for User. Then, it will redirect to user detail page.

### 2.3.2 User can login



**Figure 6: SD-02 User can login**

Admin and wholesalermust have to login before using the system. First, both of user type must inputs username and password. Then select login button,AccountController will be checking the model. If authentication valid will be call action RedirectTolocal. Elsethe user no permission, the controller will redirect tologin page.

### 2.3.3 User can logout

****

**Figure 7: SD-03 User can logout**

Admin and wholesalercan logout from the system. First, both of user type select menu list on the top right of homepage. Then select logout to the system in UI-15. When user confirms logging out, the system willbe call action Logout from AccountController to Authentication. Then the system returns logout to AccoutController in UI-01.

### 2.3.4 Admin can edit user information



**Figure 08: SD-04 Admin can edit user information**

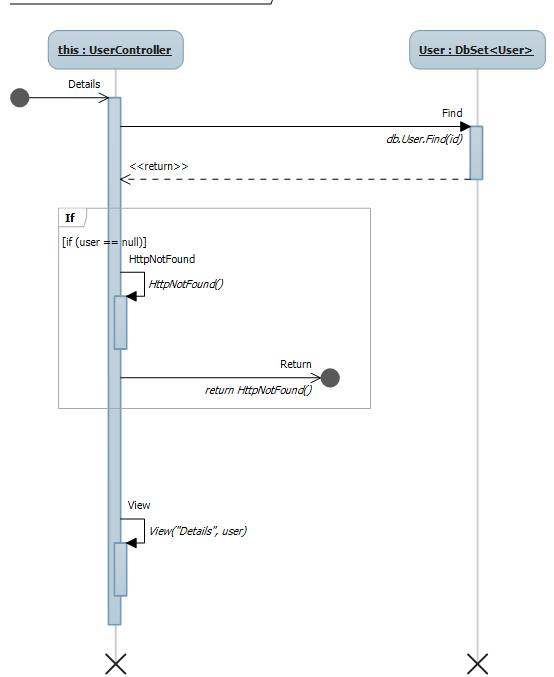
First, Admin edits user by select edit link and inputs new update information. Then the system call action edit from UserController andcall FirstOrDefault from Dbset<user>to found username of user in the database. If id of this username is equal to founded id, the system can update new user information to the database.

### 2.3.5 Admin can delete user

**Figure 09: SD-05 Admin can delete user**

First, Admin deletes user by select delete button and submits to the controller. Then, the controller will find id of user and delete from the database. Then, it will redirect to user index page.

### 2.3.6 Admin can view user information



**Figure 10: SD-06 Admin can view user information**

First, Admin view user information by select detail link. Then, the controller will find id of user and return view to display information from the database. Then, it will redirect to user details page.

### 2.3.7 Admin can create bakery product



**Figure 11: SD-07Admin can create bakery product**

Admin can create new bakery product information. First, select product management button on the main page of admin. Then select create new bakery product and input bakery product information in UI-09. When admin confirms adding new bakery product information, the system will be call create action to ProductController and check model. If valid model, ProductController will be call add action to Dbset<Product> in the database. After save changes, the system will be return to index page of admin.

### 2.3.8 Admin can edit bakery product information



**Figure 12: SD-08Admin can edit bakery product information**

First, Admin edits bakery product by inputs update bakery product and submits to the controller. Then, the system will be call edit action to ProductController and check model. If model is valid, ProductController will be call Entry<product> action to updates new bakery product information and then save change in the database. Then, controller will be call list of all productsbefore return to the index page.

### 2.3.9 Admin can delete bakery product

****

**Figure 13: SD-09Admin can delete bakery product**

First, admin deletes bakery product by select delete button and submit to the controller. The system will be call DeleteConfired action to Productcontroller and find product id that match to the require product. Then return this product back to ProductController and call remove action to remove the product from the database. After finishedcontroller will be remove that product from list of product and return to index page of admin.

### 2.3.10 Usercan view bakery productlist

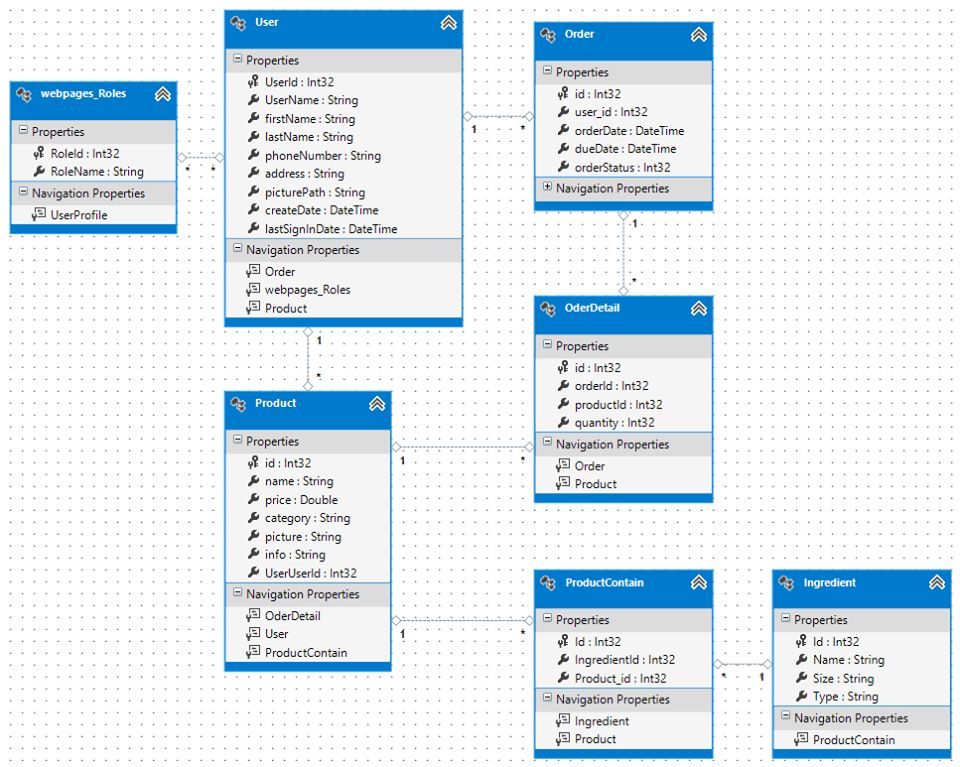


**Figure 14: SD-10 User can view bakery productlist**

First, main menu user can view bakery product information byselect product management page in UI-03 or UI-04, separate by type of users. The system will be call product action to HomeController then check keyword from search. If have keyword from userHomeController will call LINQ action to found list of products from Dbset<product> and return list of product UI-12. Else, HomeController display list of all product from the database.

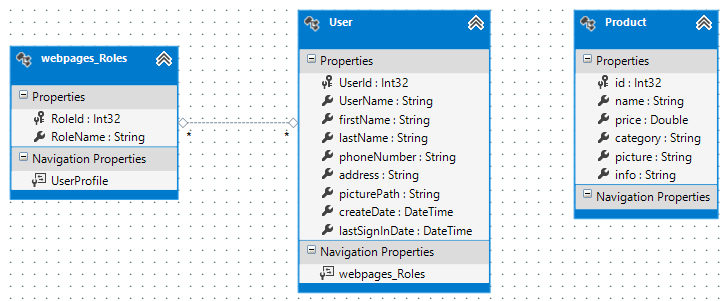
# **Chapter Three | Data Architecture**

## 3.1 ER Diagram



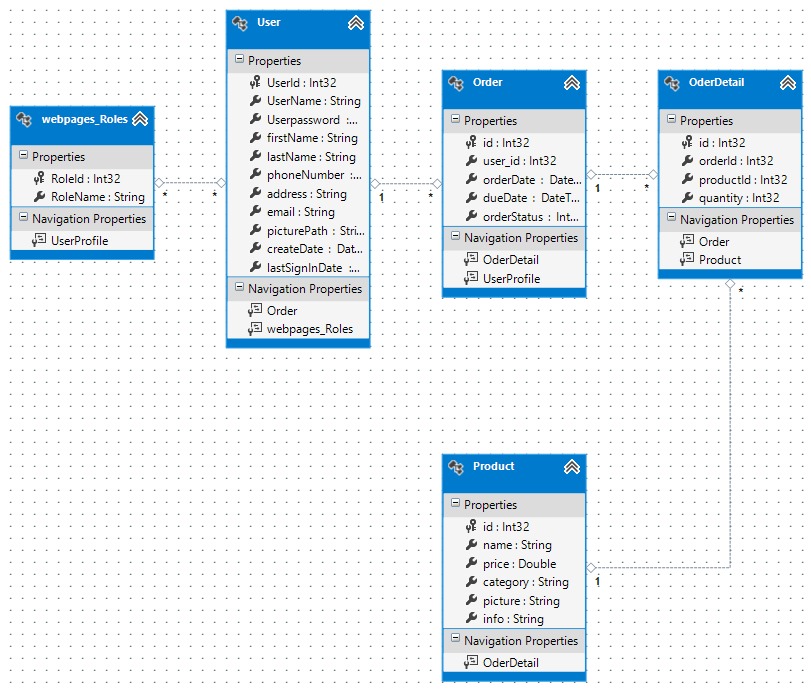
**Figure 15: ER Diagram**

## 3.2 ER Diagram Description

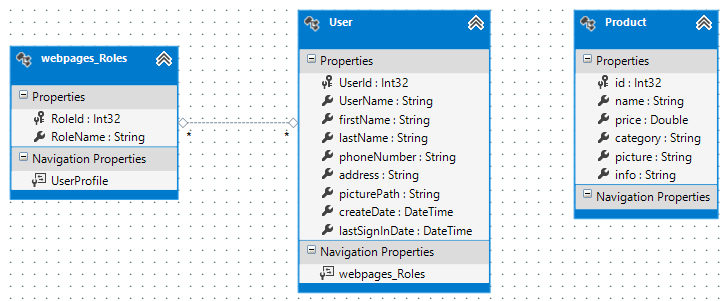


**DB-01: Product**

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Extra** |
| id | Int(32) | - Primary Key  - Not Null  - Auto Incremental |
| name | String | - Not Null |
| price | Double | - Not Null |
| category | String | - Not Null |
| picture | String | - Not Null |
| info | String | - Not Null |

**DB-02: Webpage\_Roles**

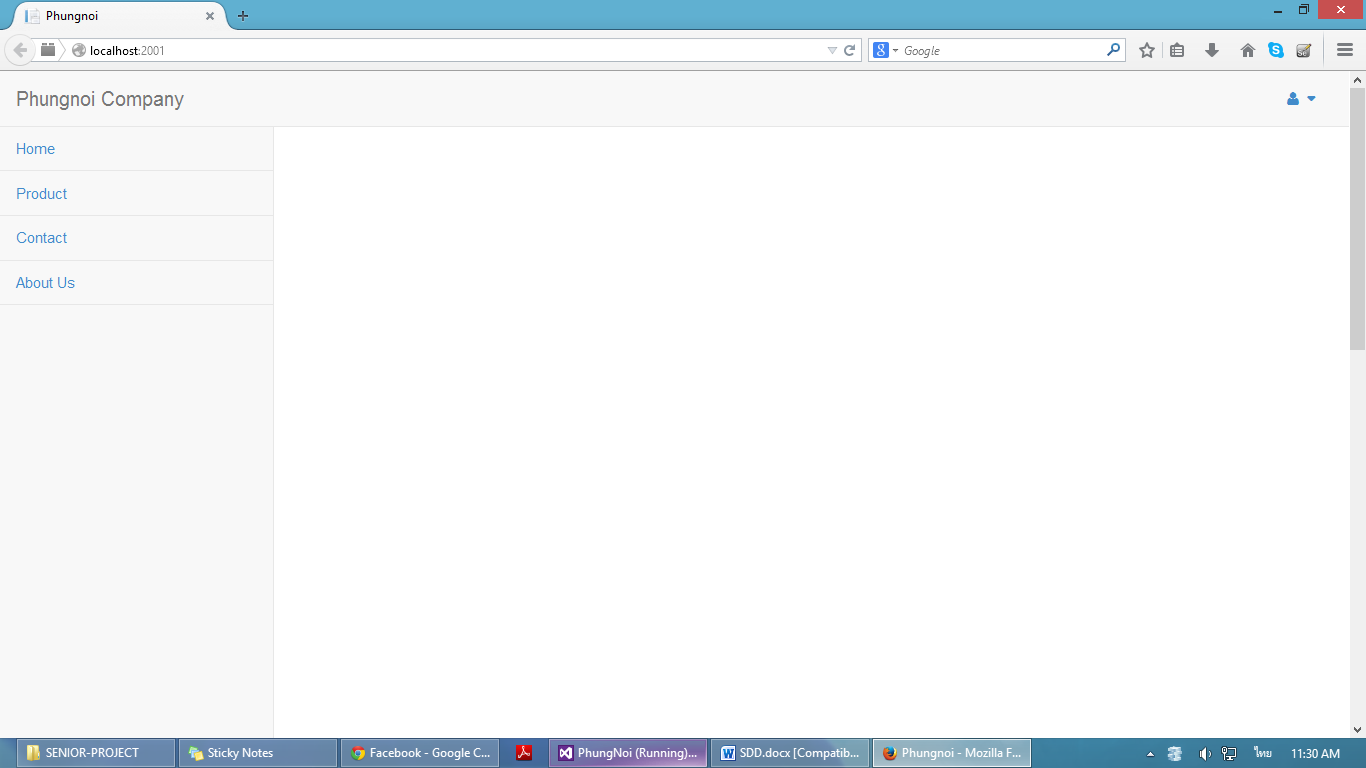
|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Extra** |
| RoleId | Int(32) | - Primary Key  - Not Null |
| RoleName | String | - Not Null |

**DB-03: User**

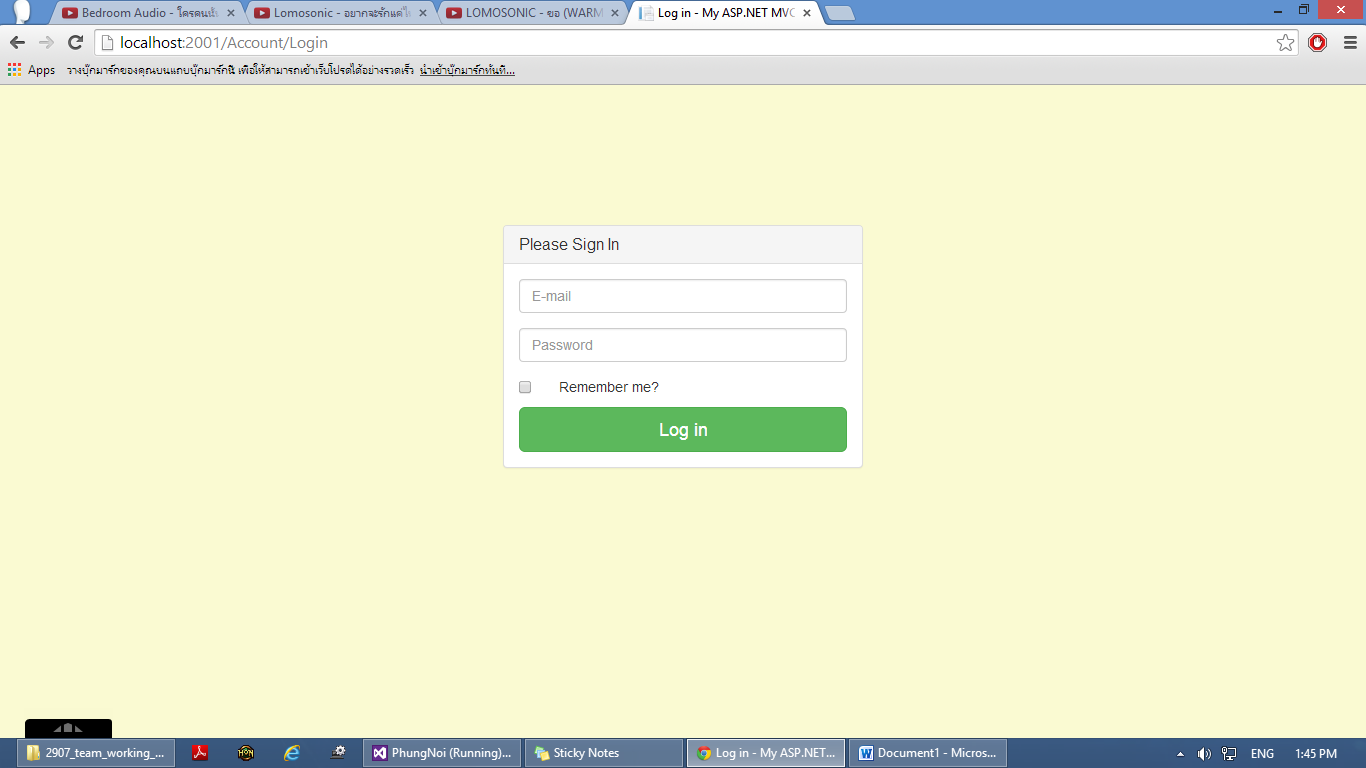
|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Extra** |
| userId | Int(32) | - Primary Key  - Not Null  - Auto Incremental |
| userName | String | - Not Null |
| firstName | String | - Not Null |
| lastName | String | - Not Null |
| phoneNumber | String | - Not Null |
| address | String | - Not Null |
| picturePath | String | - Not Null |
| createDate | DateTime | - Not Null |
| lastSigninDate | DateTime | - Not Null |

\* Only database in Progress Report I

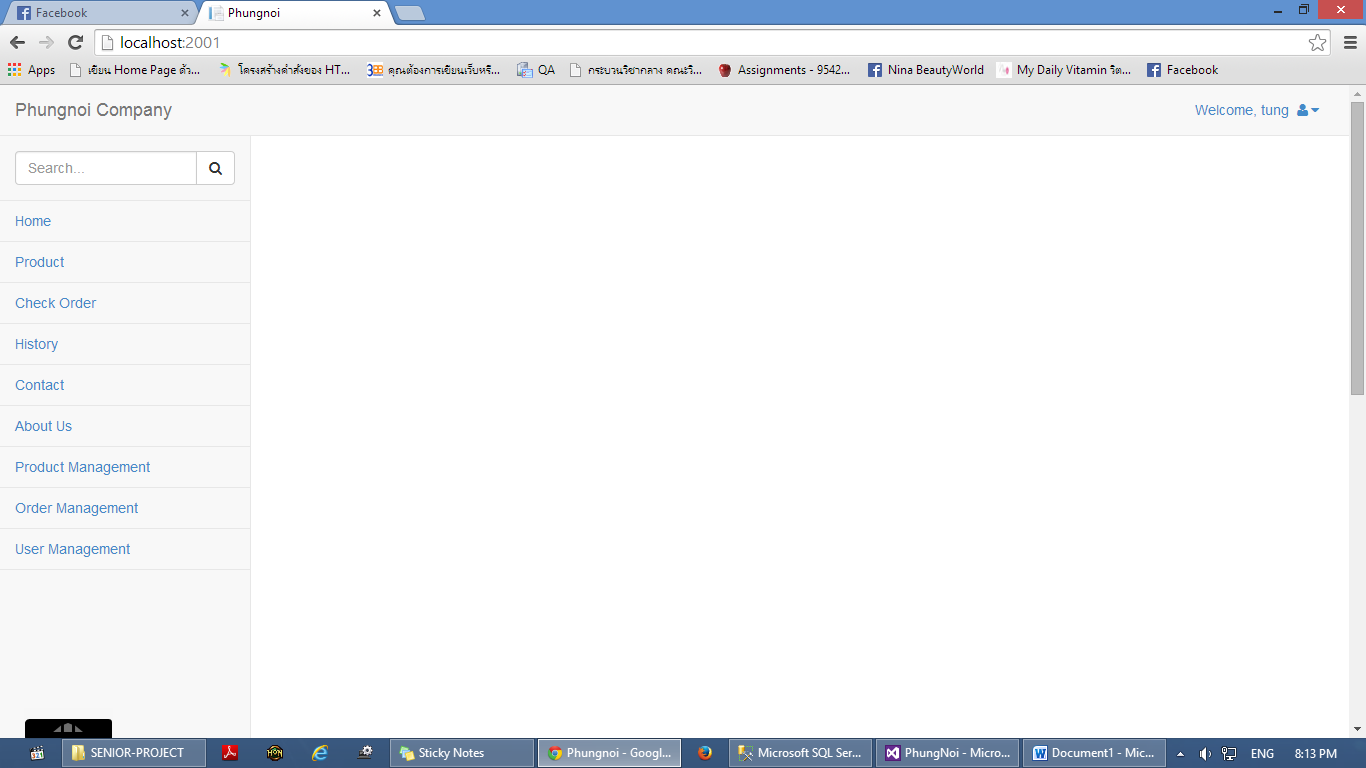
# **Chapter Four | User Interface**



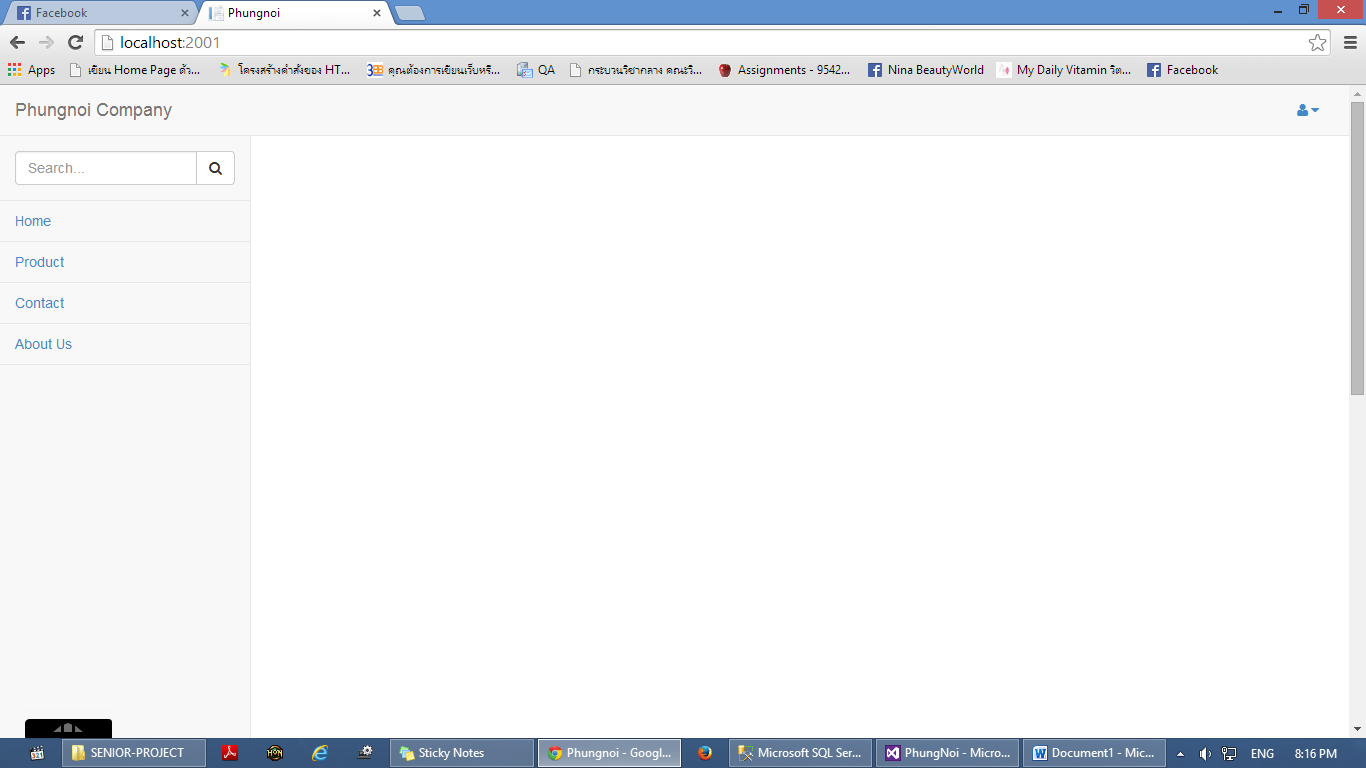
**Figure 16:** UI-01 Home page



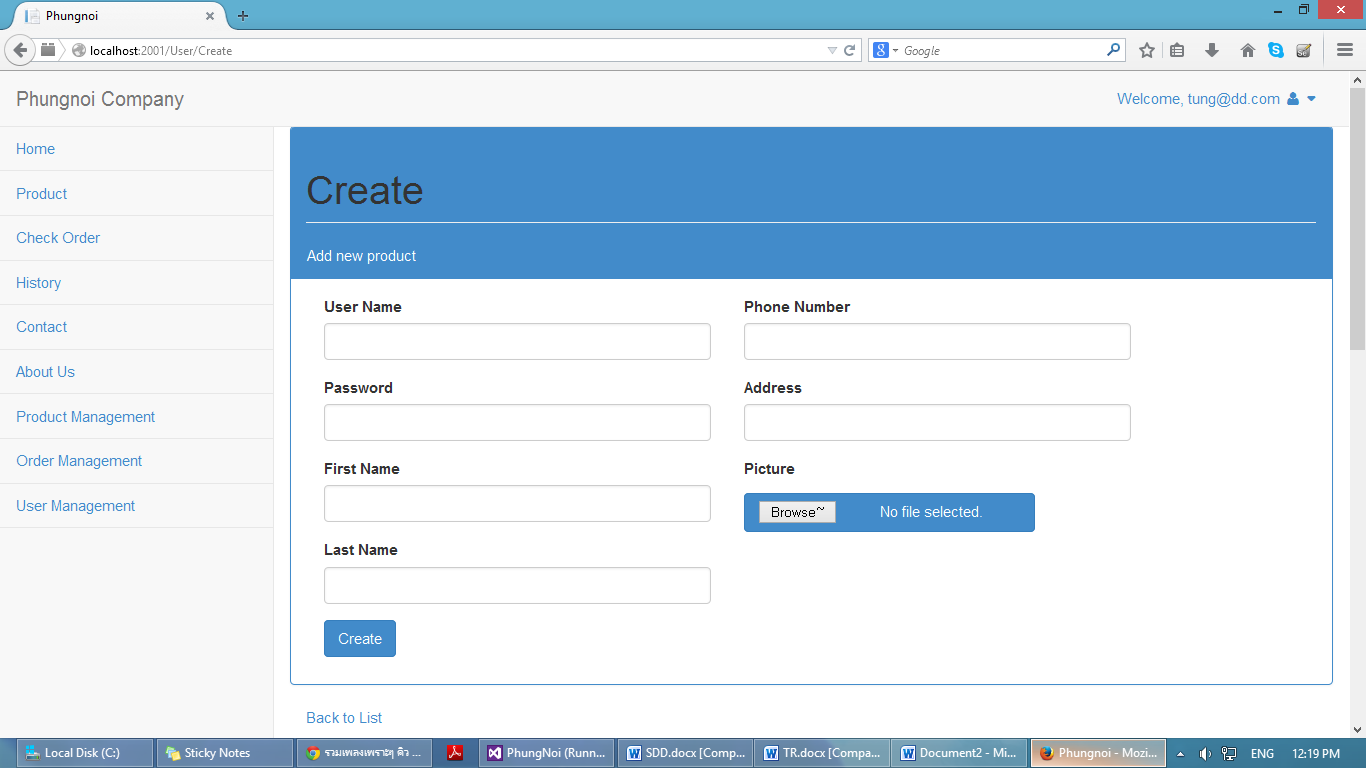
**Figure 17:** UI-02Login page

****

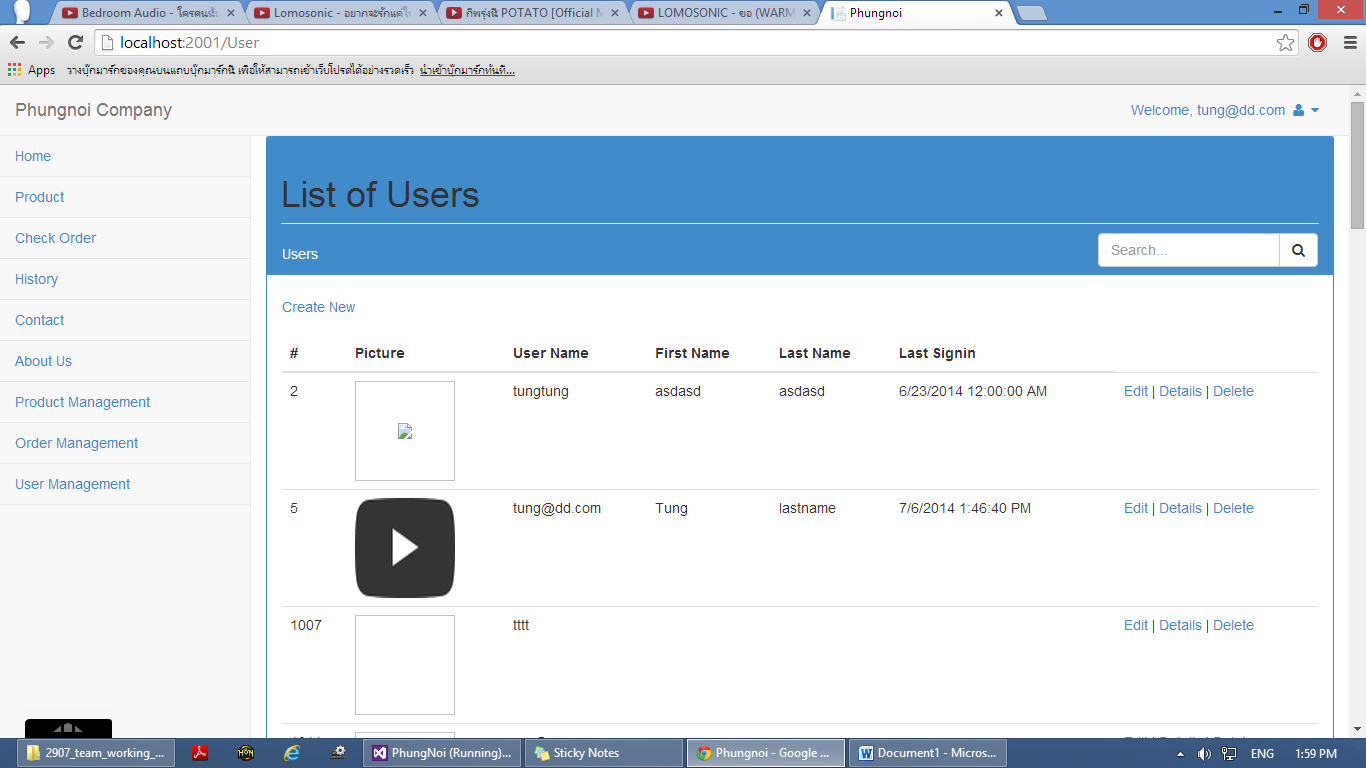
**Figure 18:** UI-03 Admin main layout page



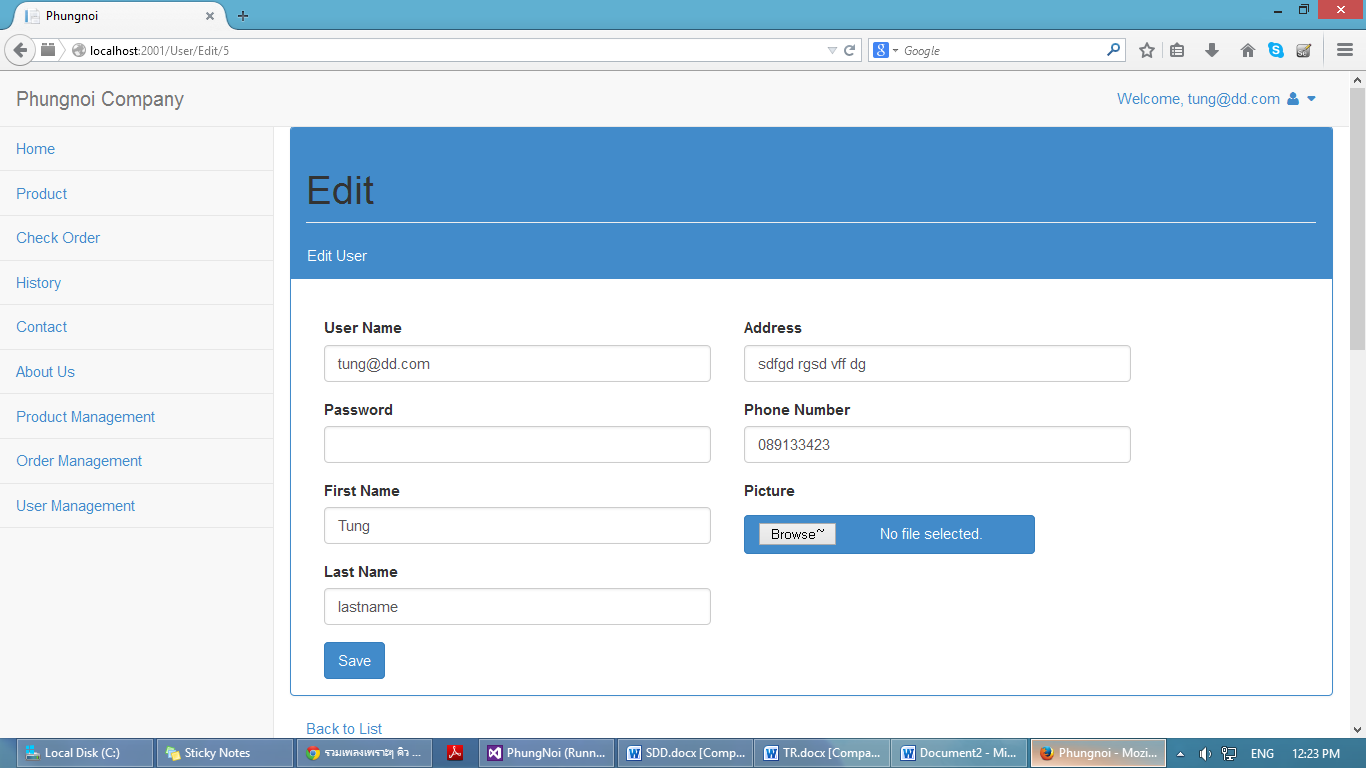
**Figure 19:** UI-04Wholesaler main layout page



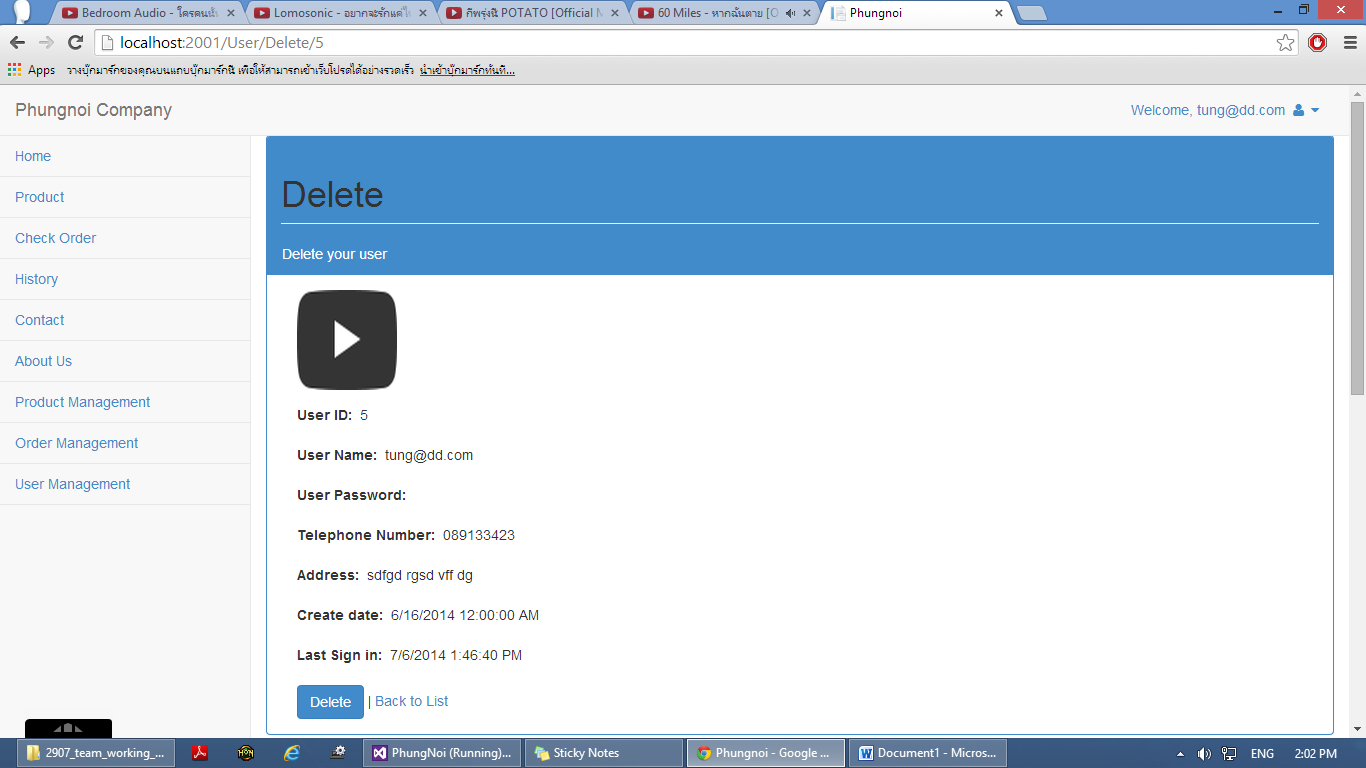
**Figure 20:** UI-05 Create user page



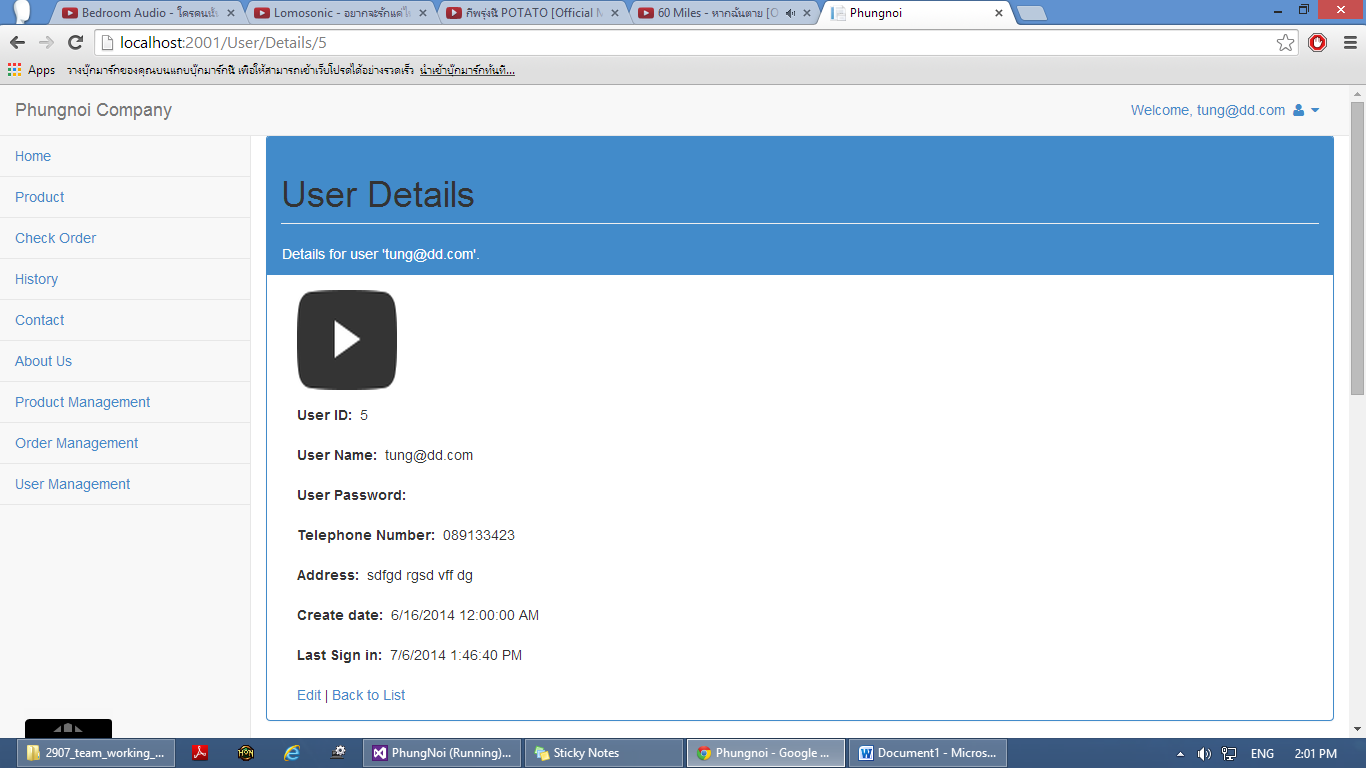
**Figure 21:** UI-06User management page



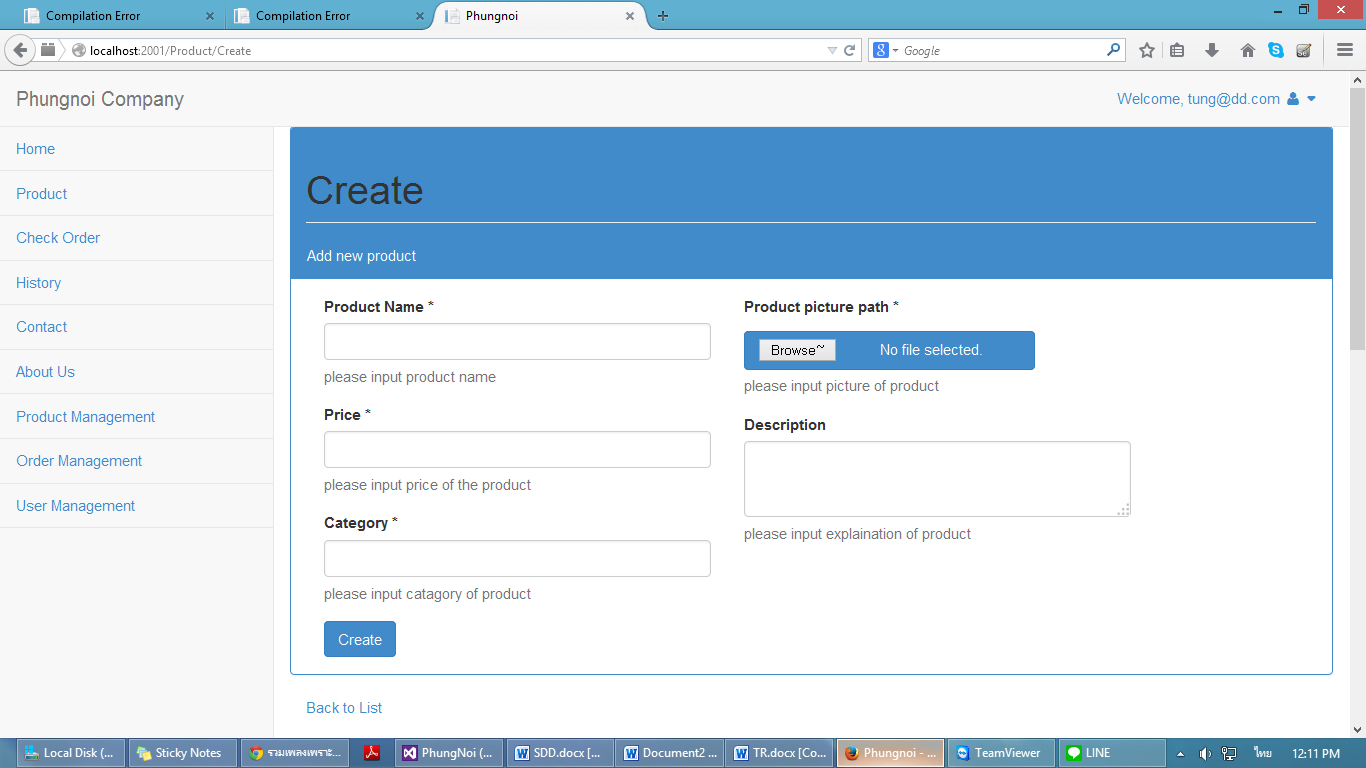
**Figure 22:** UI-07 Edit user information page



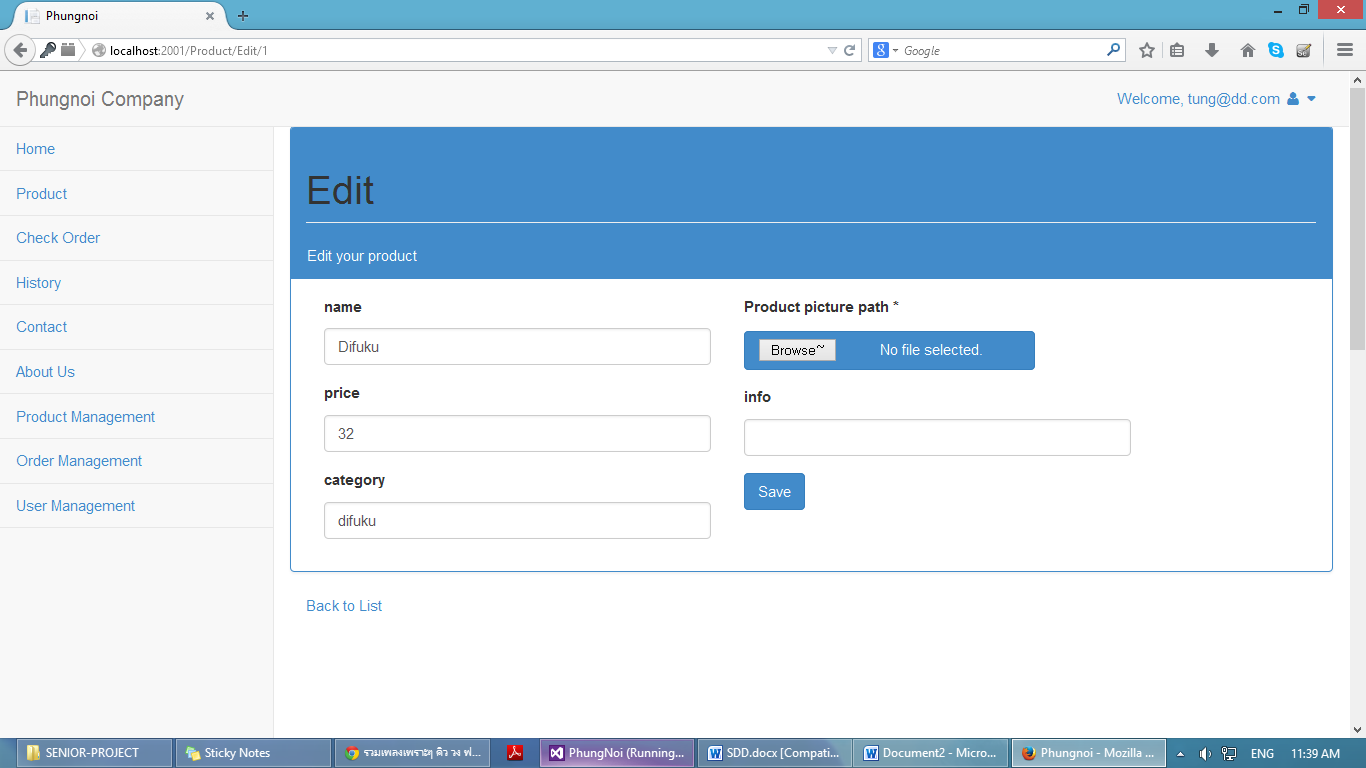
**Figure 23:** UI-08 Delete user page



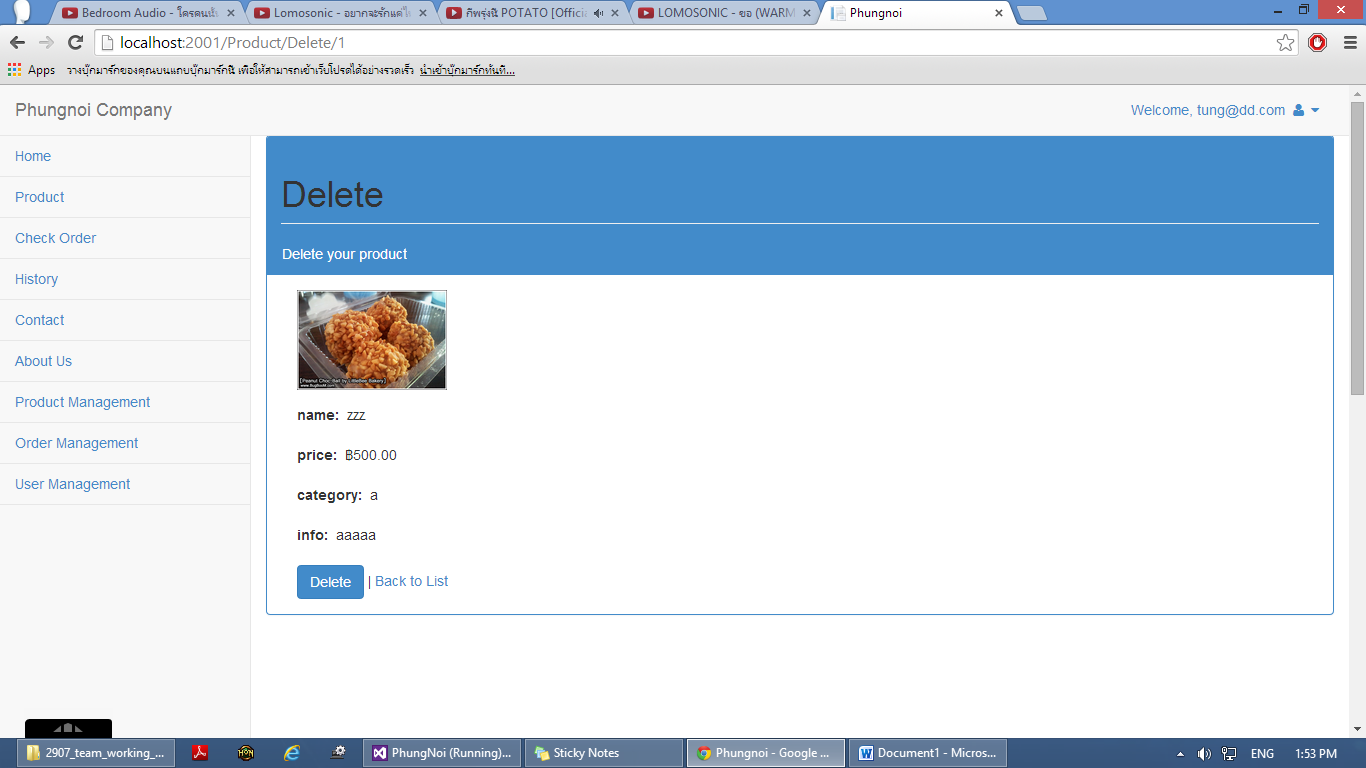
**Figure 24:** UI-09Edit own detail page



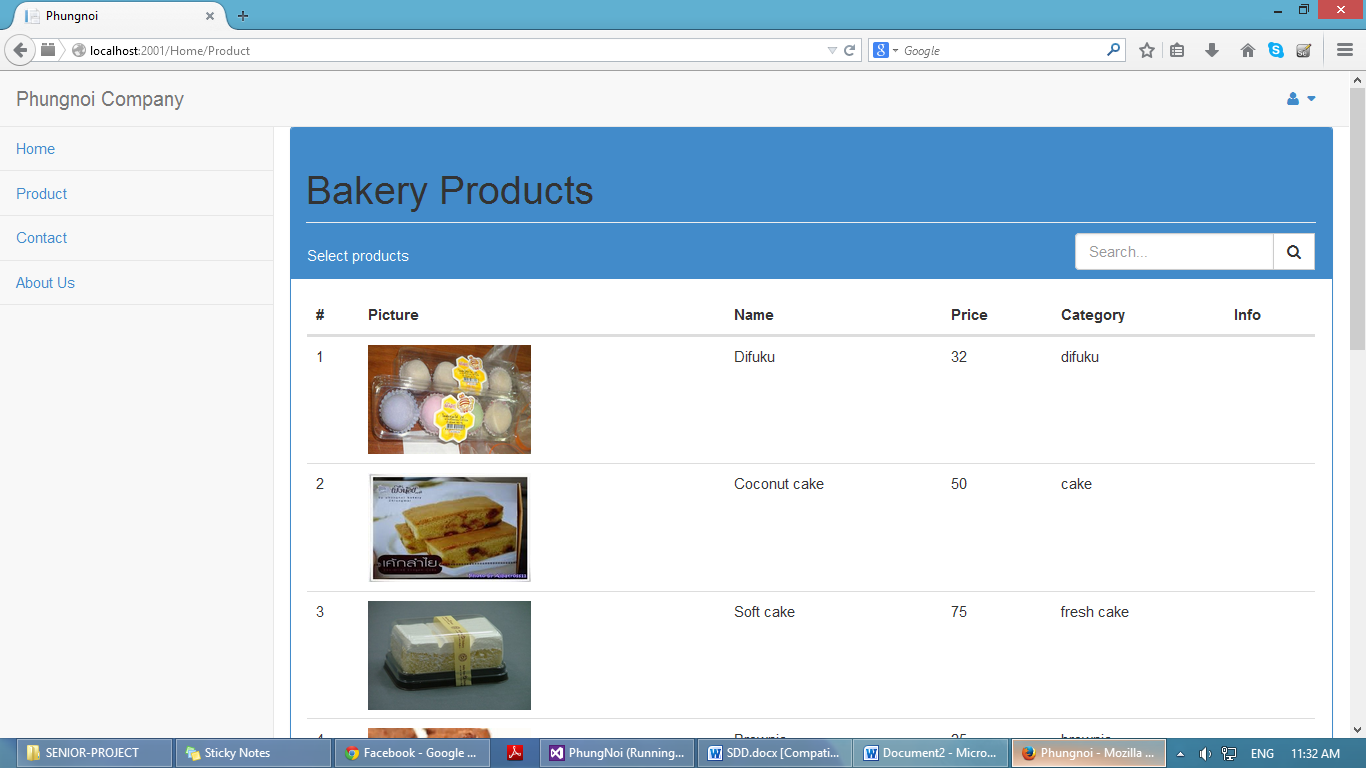
**Figure 25:** UI-10Create bakery product page

****

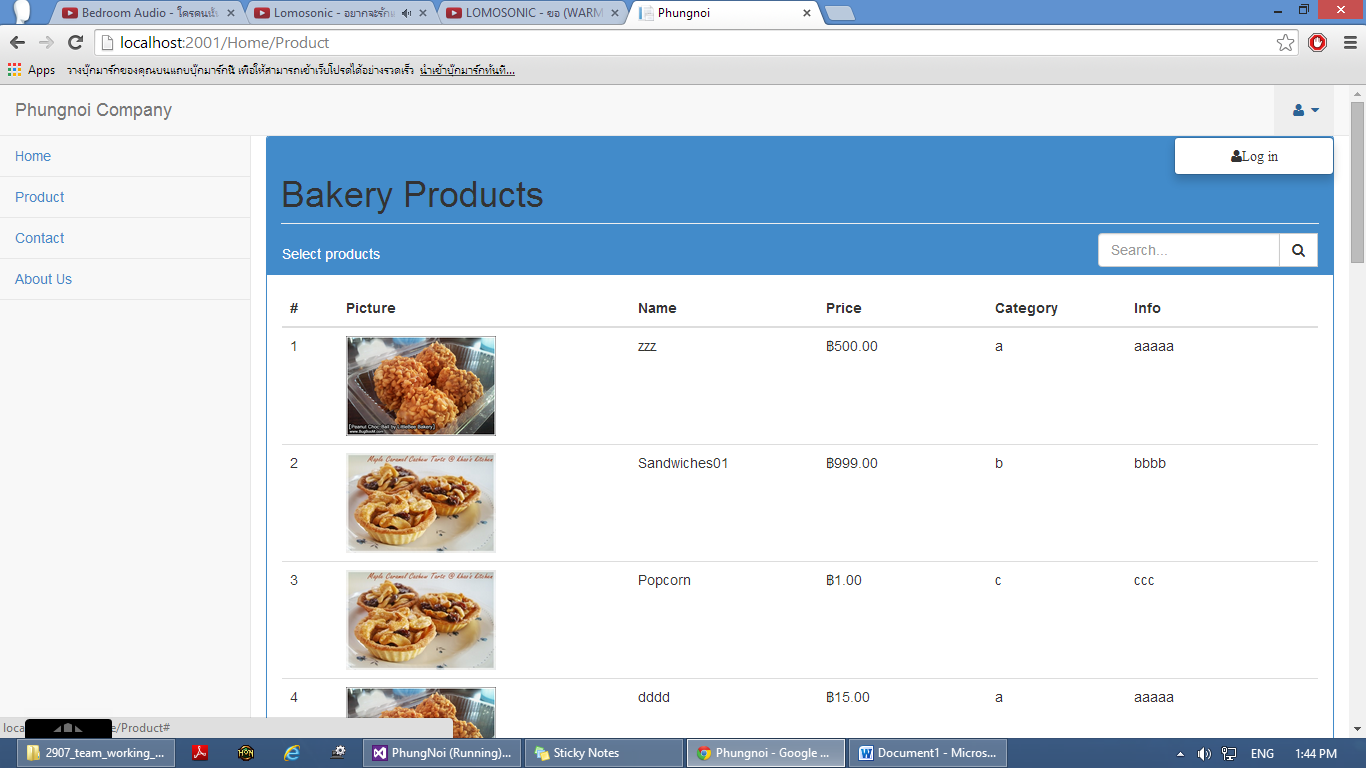
**Figure 26:** UI-11 Edit bakery product page



**Figure 27:** UI-12 Delete bakery product page

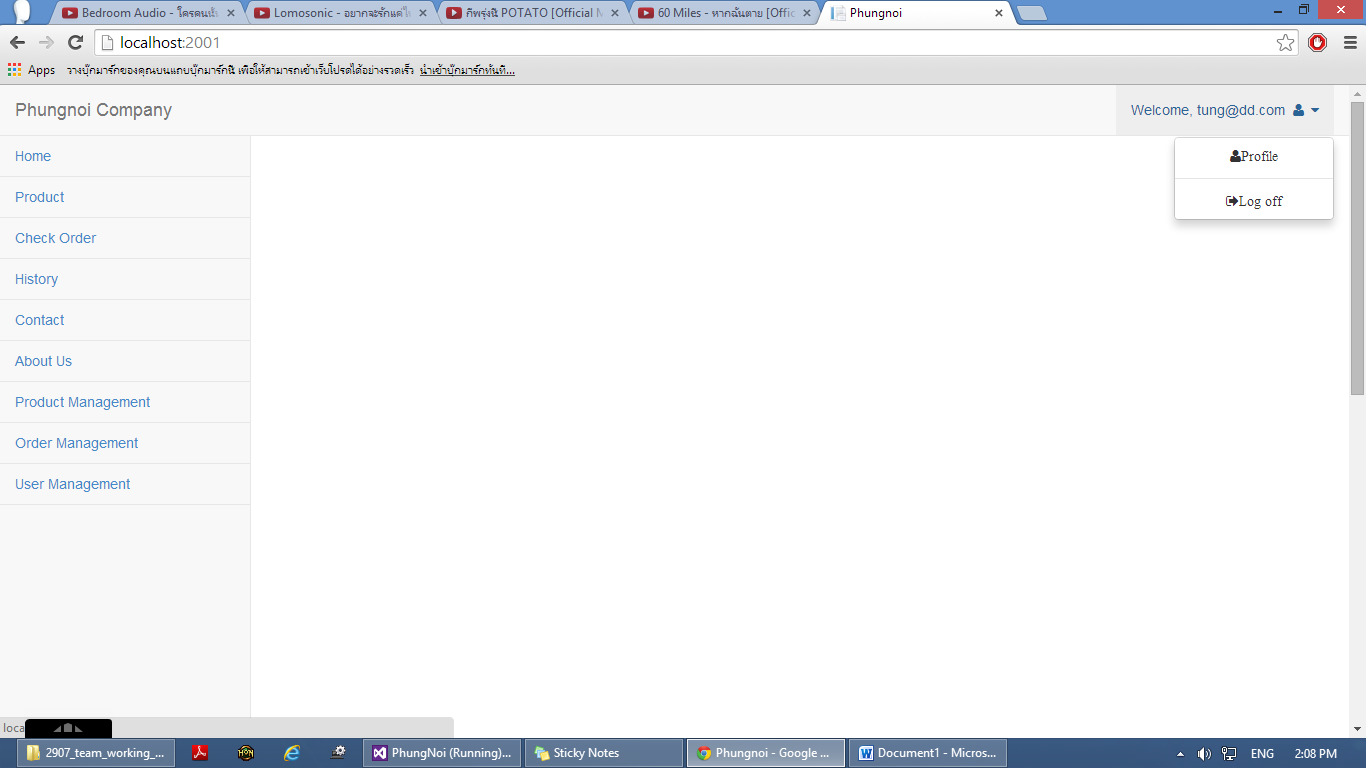


**Figure 28:** UI-13Bakery product management page

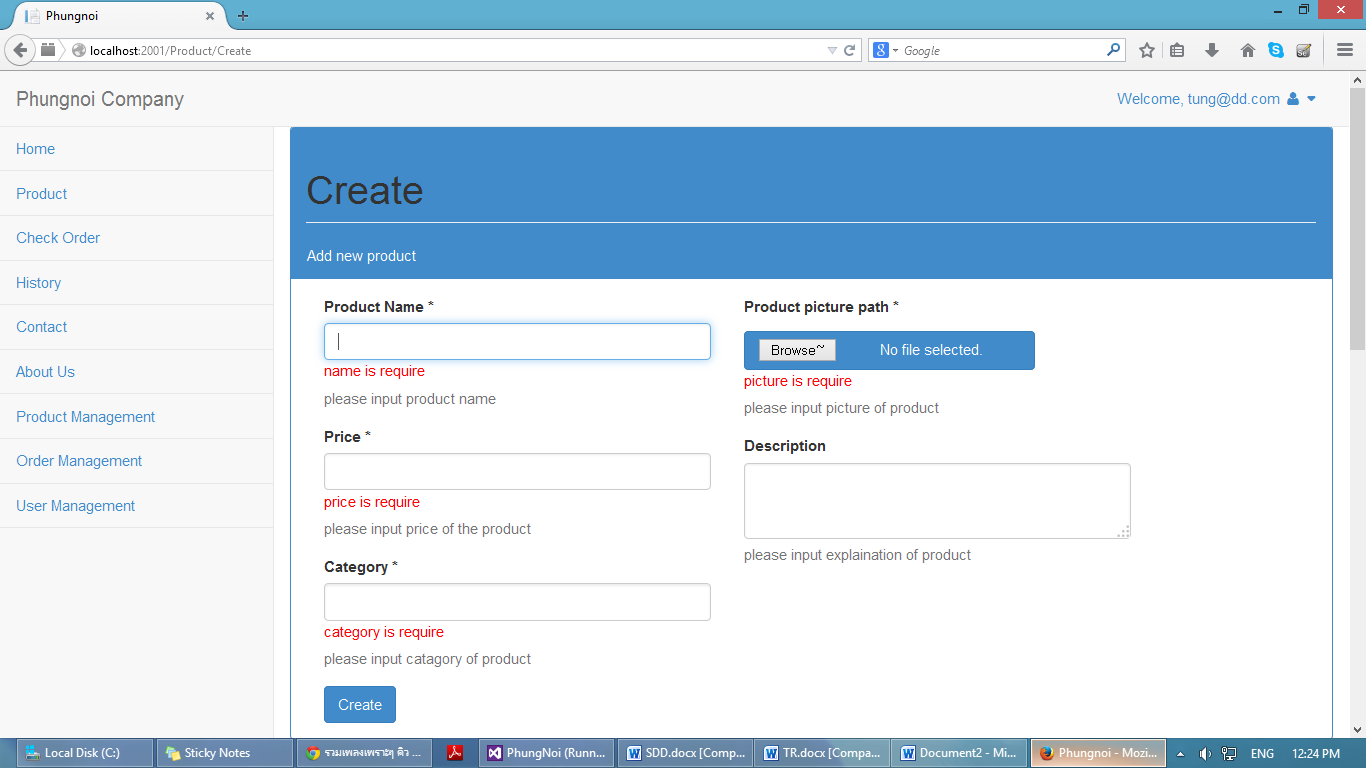


**Figure 25:** UI-16Wholesaler bakery product page

**Figure 29:** UI-14Bakery product detail page



**Figure 30:** UI-15 Logout page



**Figure 31:** UI-16 Validationform page