Contents

[1. URS](#_Toc380007225) 1

[1.1 List URS 15](#_Toc380007226)

[1.2 user Requirement Specification 16](#_Toc380007227)

[**URS-01**: Customer can registration themselves as customer. 16](#_Toc380007228)

[**URS-02**: Customer can login to the system. 18](#_Toc380007280)

[**URS-03**:Customer can view all the product category. 19](#_Toc380007302)

[**URS-04:** Customer can search the product. 20](#_Toc380007309)

[**URS-05**: Customer can add the product as many products as customer want to the shopping cart. 21](#_Toc380007326)

[**URS-06:** Customer can edit the product in the shopping cart. 22](#_Toc380007342)

[**URS-07:** Customer can save the product in the shopping cart. 23](#_Toc380007364)

[**URS-09:** Customer can view the shopping history. 25](#_Toc380007378)

[**URS-10:** Customer can logout. 26](#_Toc380007385)

[**URS-11**: Administrator can login to the system. 27](#_Toc380007392)

[**URS-12:** Administrator can view the shopping history of all customer. 28](#_Toc380007414)

[**URS-13:** Administrator can provide detail of the product. 29](#_Toc380007420)

[**URS-14:** Administrator can update all information of registration. 30](#_Toc380007451)

[**URS-15:** Administrator can logout……………………………………………………………………………………………………31](#_Toc380007503)

[2. SRS 34](#_Toc380008193)

[2.1 List SRS 34](#_Toc380008194)

[3. Use case Diagram 36](#_Toc380008195)

[List use cases 36](#_Toc380008196)

[3.1 Use case 37](#_Toc380008197)

[4. Class Diagram 37](#_Toc380008198)

[4.1 List Class Diagram 37](#_Toc380008199)

[4.2 Class Diagram 38](#_Toc380008200)

[4.2.1 Entity 38](#_Toc380008201)

[4.2.2 MVC of User Controller 39](#_Toc380008202)

[4.2.3 MVC of Product Controller 40](#_Toc380008203)

[4.2.4 MVC of Shopping Cart Controller 40](#_Toc380008204)

[4.2.5 MVC of History Controller 41](#_Toc380008205)

[5. Sequence Diagram 41](#_Toc380008206)

[5.1 List Sequence Diagram 41](#_Toc380008207)

[5.2 Sequence Diagram 42](#_Toc380008208)

[5.2.1 SQD-01: Add Product Shopping Cart 42](#_Toc380008209)

[5.2.2 SQD-02: Check Out 42](#_Toc380008210)

[5.2.3 SQD-03: Edit Product Shopping Cart 43](#_Toc380008211)

[5.2.4 SQD-04: Registration 45](#_Toc380008212)

[5.2.5 SQD-05: Save Product Shopping Cart 46](#_Toc380008213)

[5.2.6 SQD-06: View Product 46](#_Toc380008214)

[5.2.7 SQD-07: Search Product 47](#_Toc380008215)

[5.2.8 SQD-08: Add Product 48](#_Toc380008216)

[5.2.9 SQD-09: Update Registration 49](#_Toc380008217)

[5.2.10 SQD-10: Login 50](#_Toc380008218)

[5.2.11 SQD-11: Logout 50](#_Toc380008219)

[5.2.12 SQD-12: View History 51](#_Toc380008220)

[6. Method Design 51](#_Toc380008221)

[6.1 List method 51](#_Toc380008222)

[6.2 Proper method design 52](#_Toc380008223)

[6.2.1 Entity 52](#_Toc380008224)

[6.2.1.1 MTD-01: User 52](#_Toc380008225)

[6.2.1.2 MTD-02: Administrator 53](#_Toc380008226)

[6.2.1.3 MTD-03: Customer 54](#_Toc380008227)

[6.2.1.4 MTD-04: Product 55](#_Toc380008228)

[6.2.1.5 MTD-05: Shopping Cart 56](#_Toc380008229)

[6.2.1.6 MTD-06: History 57](#_Toc380008230)

[6.2.2 MVC Pattern of User Class 58](#_Toc380008231)

[6.2.2.1 MTD-07: UserDAO 58](#_Toc380008232)

[6.2.2.2 MTD-08: UserDAOImpl 59](#_Toc380008233)

[6.2.2.3 MTD-09: UserService 60](#_Toc380008234)

[6.2.2.4 MTD-10: UserServiceImpl 61](#_Toc380008235)

[6.2.2.5 MTD-11: UserController 62](#_Toc380008236)

[6.2.3 MVC Pattern of Product Class 63](#_Toc380008237)

[6.2.3.1 MTD-12: ProductDAO 63](#_Toc380008238)

[6.2.3.2 MTD-13: ProductDAOImpl 63](#_Toc380008239)

[6.2.3.3 MTD-14: ProductService 64](#_Toc380008240)

[6.2.3.4 MTD-15: ProductServiceImpl 64](#_Toc380008241)

[6.2.3.5 MTD-16: ProductController 65](#_Toc380008242)

[6.2.4 MVC Pattern of ShoppingCart Class 66](#_Toc380008243)

[6.2.4.1 MTD-17: ShoppingCartDAO 66](#_Toc380008244)

[6.2.4.2 MTD-18: ShoppingCartDAOImpl 66](#_Toc380008245)

[6.2.4.3 MTD-19: ShoppingCartService 67](#_Toc380008246)

[6.2.4.4 MTD-20: ShoppingCartServiceImpl 68](#_Toc380008247)

[6.2.4.5 MTD-21: ShoppingCartController 69](#_Toc380008248)

[6.2.5 MVC Pattern of History Class 70](#_Toc380008249)

[6.2.5.1 MTD-22: HistoryDAO 70](#_Toc380008250)

[6.2.5.2 MTD-23: HistoryDAOImpl 71](#_Toc380008251)

[6.2.5.3 MTD-24: HistoryService 72](#_Toc380008252)

[6.2.5.4 MTD-25: HistoryServiceImpl 73](#_Toc380008253)

[6.2.5. MTD-26: HistoryController 74](#_Toc380008254)

[Traceability 75](#_Toc380008255)

[Traceability matric between URS/SRS 75](#_Toc380008256)

[Traceability matric between URS/Class Diagram 75](#_Toc380008257)

[Traceability matric between URS/Use cases Diagram 75](#_Toc380008258)

[Traceability matric between URS/Sequence Diagram 75](#_Toc380008259)

[Traceability matric between URS/Method designs 75](#_Toc380008260)

[Traceability matric between SRS/Use cases Diagram 75](#_Toc380008261)

[Traceability matric between SRS/Class Diagram 75](#_Toc380008262)

[Traceability matric between SRS/Sequence Diagram 75](#_Toc380008263)

[Traceability matric between SRS/Method designs 75](#_Toc380008264)

[Traceability matric between Use cases Diagram / Class Diagram 75](#_Toc380008265)

[Traceability matric between Use cases Diagram /Sequence Diagram 75](#_Toc380008266)

[Traceability matric between Use cases Diagram /Method designs 75](#_Toc380008267)

[Traceability matric between Class Diagram/Sequence Diagram 75](#_Toc380008268)

[Traceability matric between Class Diagram/Method designs 75](#_Toc380008269)

[Traceability matric between Sequence Diagram/Method designs 75](#_Toc380008270)

# 1. URS

## 1.1 List URS

|  |  |
| --- | --- |
| **ID** | **Description** |
| **URS-01** | Customer can registration themselves as customer. |
| **URS-02** | Customer can login to the system. |
| **URS-03** | Customer can view all the product category. |
| **URS-04** | Customer can search the product. |
| **URS-05** | Customer can add the product as many products as customer want to the shopping cart. |
| **URS-06** | Customer can edit the product in the shopping cart. |
| **URS-07** | Customer can save the product in the shopping cart. |
| **URS-08** | Customer can check out for purchase the product. |
| **URS-09** | Customer can view the shopping history. |
| **URS-10** | Customer can logout. |
| **URS-11** | Administrator can login to the system. |
| **URS-12** | Administrator can view the shopping history of all customer. |
| **URS-13** | Administrator can provide detail of the product. |
| **URS-14** | Administrator can update all information of registration. |
| **URS-15** | Administrator can logout. |

## 1.2 User Requirement Specification

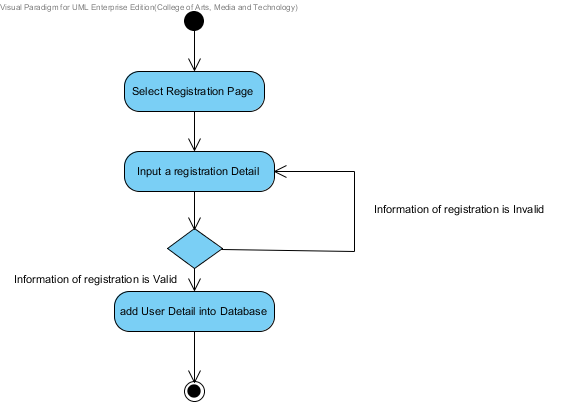
# **URS: CUSTOMER URS-01**: Customer can registration themselves as customer.

**Introduction:**  Customer register a customer’s information into the system. Customer need to provide the detail which consist of username, password, name, birth date, email, telephone Number, address and their own picture for register to the system.

**Input:** Customer request a registration Page.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Name** | **Type** | **Description** | **Example** | **Remarks** |
| username | String | A username of customer | “camt1234” | 1-50 characters |
| password | String | A password of customer. | “camt1234” | 1-50 characters |
| name | String | A Full-Name of customer | “Putsacha” | 1-40 characters |
| birthDate | DATE | A birth date of customer. | “1993-02-25” | Format : yyyy/mm/dd |
| email | String | An email of customer. | “se542115000@vr.camt.info” | - |
| telephoneNumber | String | A telephone number of customer. | “0888888888” | 0-9 digits |
| address | String | An address of customer. | “CAMT” | 1-255 characters |
| picture | BLOB | A picture of customer. | C:\Users\wuthipong\Desktop\User\542115037.jpg | - |

**Flow of Execution:**

****

**Output:**  The System add the registration of customer into database.

**System Requirement Specification**  
 **SRS-01:** The system provide an interface for registration which receive username, password,   
 name address and their picture for the registration.   
 **SRS-02:** The system can check the name format. The name format should be only a   
 character.  
 **SRS-03:** The system can check the address. The address format should be a characters   
 between 1-255 characters.  
 **SRS-04:** The system can check the picture that user uploading. The picture should be an   
 image format such as, .png, .jpg.  
 **SRS-05:** The system can check the username format. The username format should be   
 characters.  
 **SRS-06:** The system can check the password format. The password format should be   
 characters.  
 **SRS-07**: The system shall connect to the database.  
 **SRS-08**: The system retrieve information from the database.  
 **SRS-09:** The system shall add a new customer information to database using username,   
 password, name, address and their picture.  
 **SRS-10:** The system shall provide the successful registration page after registration is   
 complete.  
 **SRS-11:** The system shall display the error message “The username must be characters”.  
 **SRS-12:** The system shall display the error message “The password must be characters”.  
  **SRS-13:** The system shall display the error message “The name must be only characters”.  
 **SRS-14:** The system shall display the error message “The address must be a characters   
 between 1-255 characters”.  
 **SRS-15:** The system shall display the error message “The picture must be image format”.

# **URS-02**: Customer can login to the system.

**Introduction:**  Customer login into the system. Customer need to provide the detail which consist of username and password for validate the authentication of login into the system. The system shall validate the information of login which customer provided and session control of user duration into the system.

**Input:** Customer request a Login Page.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Name** | **Type** | **Description** | **Example** | **Remarks** |
| username | String | A username of customer. | “camt1234” | 1-50 characters |
| password | String | A password of customer. | “camt1234” | 1-50 characters |

**Flow of Execution:**

****

**Output:**  Customer login to the system.

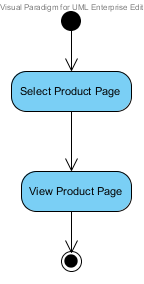
**System Requirement Specification**  
 **SRS-16:** The system provides the login interface which receive the user name and password.  
 **SRS-17:** The system can check the username format. The username format should be   
 characters.  
 **SRS-18:** The system can check the password format. The password must contain the   
 characters.  
 **SRS-19:** The system validates the username and password.  
 **SRS-20:** The system shall provide the successful login page.  
 **SRS-07**: The system shall connect to the database.  
 **SRS-08**: The system retrieve information from the database.  
 **SRS-21:** The system shall show the user status with a customer picture on the top-right of   
 the user interface to confirm his identity.

# **URS-03**:Customer can view all the product category.

**Introduction:**  Customer view all the product category in Product Page. Customer can view all the product which consist the information of product as name, description, price and image on the product page.

**Input:** Customer request a Product Page.

**Flow of Execution:**

****

**Output:**  The system show all the product detail in Product Page.

**System Requirement Specification**  
 **SRS-22:** The system provide an interface that show all the product category including, name,   
 description, price and image.   
 **SRS-07**: The system shall connect to the database.  
 **SRS-08**: The system retrieve information from the database.  
 **SRS-23:** The system show the product detail that consist of name, description, price and   
 image.

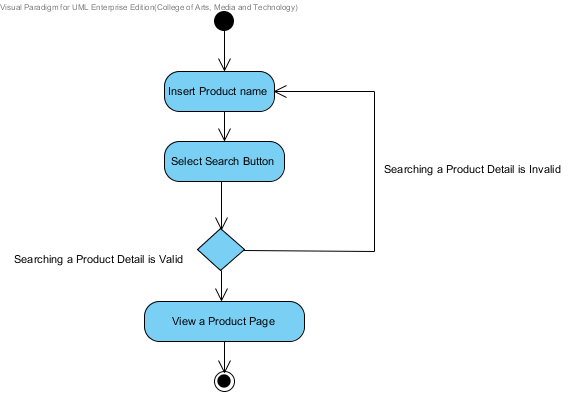
# **URS-04:** Customer can search the product.

**Introduction:**  Customer can search a product detail by using product’s name. Customer need to provide the name of the product for searching the elements. After searching the information of product which valid insertion, The system shall display the product page which user insertion corrective.

**Input:** Customer request a Product Page.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Name** | **Type** | **Description** | **Example** | **Remarks** |
| name | String | A product’s name. | “C” | 1-50 characters |

**Flow of Execution:**

****

**Output:**  The system show a product in Product Page.

**System Requirement Specification**  
 **SRS-24:** The system shall provide interface to search a product page which receive product   
 name for searching.  
 **SRS-07**: The system shall connect to the database.  
 **SRS-25**: The system retrieve information from the database by name.  
 **SRS-26:** The system shall provide detail page that consist with product name, product   
 description, and product picture.

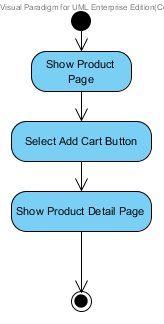
# **URS-05**: Customer can add the product as many products as customer want to the shopping cart.

**Introduction:**  Customer can add a product into shopping Cart by click “Add Cart” button on Product page. Customer need to select the product for adding into cart, After adding the product into the shopping cart, The system shall be adding, calculated total price and available for customer to view their shopping cart.

**Input:** Click “Add Cart” Button.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Name** | **Type** | **Description** | **Example** | **Remarks** |
| Add Cart | Button | User Interface | Add Cart | 1 time |

**Flow of Execution:**

****

**Output:**  The system show updating products on Shopping Cart Page.

**System Requirement Specification**   
 **SRS-27:** The system shall provide the interface to add new product which user selected.  
 **SRS-28:** The system add the product that user is selected into the shopping cart.

**SRS-07**: The system shall connect to the database.  
 **SRS-29:** The system shall update the product that user is selected into the database.  
 **SRS-30:** The system shall provide the successful add new product page after user add new   
 product into the shopping cart.

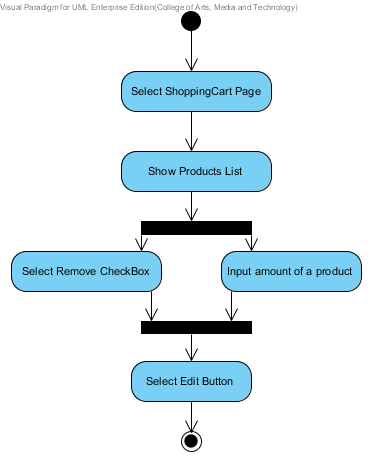
# **URS-06:** Customer can edit the product in the shopping cart.

**Introduction:**  Customer can delete or update product in Shopping Cart list. Customer need to select the product for deleting from their cart, after deleting the product from their cart, the system shall be deleted and the number of product completely compute as new total price.

**Input:** Customer select remove check box and input number of each product.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Name** | **Type** | **Description** | **Example** | **Remarks** |
| Check box | True/False | The check box UI for remove item. | download | Checked box is remove and Empty box is keep the item |
| amount | number | The number increase one step per | spinner_gallery | - |

**Flow of Execution:**

****

**Output:**  The system update Shopping Cart Page.

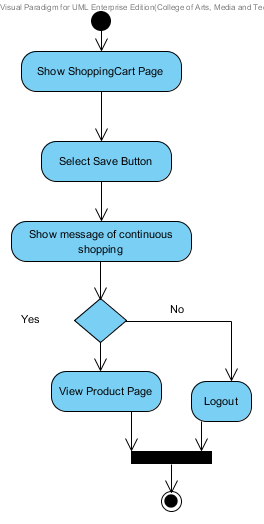
**System Requirement Specification**  
 **SRS-07**: The system shall connect to the database.  
 **SRS-31**: The system shall delete the products from the database.  
 **SRS-32:** The system shall update amount of the products into database.  
 **SRS-33**: The system shall display the error message “The system cannot edit the new   
 updating”.  
 **SRS-34**: The system shall display the message “Edit Completed” when updated in the   
 database.  
 **SRS-35**: The system provide an interface for edit function.

# **URS-07:** Customer can save the product in the shopping cart.

**Introduction:**  Customer save products into Shopping Cart system. Customer need to select the product for save a product into their cart, after save the product, the system shall save all of information and prepare the stage which customer continuous shopping or not.

**Input:** Customer select “save” button.

**Flow of Execution:**

****

**Output:**  Update product to Shopping Cart.

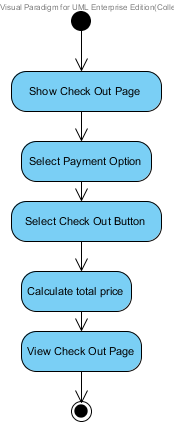
**System Requirement Specification**  
 **SRS-07**: The system shall connect to the database.  
 **SRS-36:** The system shall update information into the database.  
  **SRS-37:** The system shall display error message “The system cannot save the new product”.  
 **SRS-38:** The system shall display error message “The system cannot connect to the   
 database”.  
 **SRS-39:** The system shall display the message “Save Completed” when updated in the   
 database.  
 **SRS-40:** The system provide an interface for save function.

# **URS-08:** Customer can check out for purchase the product.

**Introduction:**  Customer can check out for purchase the product. Customer can check out for summary all the product that customer adding into the shopping cart.

**Input:** Customer request a Check out Page.

**Flow of Execution:**

****

**Output:**  The system show status of purchase.

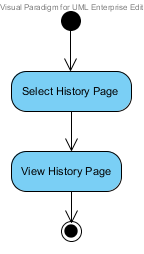
**System Requirement Specification**  
 **SRS-07**: The system shall connect to the database.  
 **SRS-08**: The system retrieve information from the database.  
 **SRS-36**: The system update product information in database.  
 **SRS-41**: The system shall provide an interface for Check out function.  
 **SRS-22:** The system provide an interface that show all the product category including, name,   
 description, price and image  
 **SRS-42:** The system shall provide an interface for select the payment option including,   
 Transfer, Credit card and PayPal.  
 **SRS-43:** The system shall update payment option into the database.  
 **SRS-44:** The system shall display message “Purchase successful” when the transect amount   
 from banking system.

# **URS-09:** Customer can view the shopping history.

**Introduction:**  Customer can view the shopping history. The system shall provide a list of shopping history which consist of name of product, amount of product that customer purchasing and date that customer purchase a product.

**Input:** Customer request History Page.

**Flow of Execution:**

****

**Output:**  The system show the history in History Page.

**System Requirement Specification**  
 **SRS-45:** The system provide an interface for the shopping history page.  
 **SRS-46**: The system retrieve shopping history form the database by reference from user.

# 

# **URS-10:** Customer can logout.

**Introduction:**  Customer logout off the system. The system shall close the session of customer for logout off the system.

**Input:** Customer select a Logout Button.

**Flow of Execution:**

****

**Output:**  Customer logout off the system.

**System Requirement Specification**  
 **SRS-47:** the system provide an interface for logout.  
 **SRS-48:** The system doesn’t show a user status on top right after logout successful.

# **URS: ADMINISTRATOR URS-11**: Administrator can login to the system.

**Introduction:**  Administrator login into the system. Administrator need to provide the detail which consist of username and password for validate the authentication of login into the system. The system shall validate the information of login which customer provided and session control of user duration into the system.

**Input:** Administrator request a Login Page.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Name** | **Type** | **Description** | **Example** | **Remarks** |
| username | String | A username of customer. | “camt1234” | 1-50 characters |
| password | String | A password of customer. | “camt1234” | 1-50 characters |

**Flow of Execution:**

****

**Output:**  Administrator login to the system.

**System Requirement Specification**  
  
 **SRS-16:** The system provides the login interface which receive the user name and password.  
 **SRS-17:** The system can check the username format. The username format should be   
 characters.  
 **SRS-18:** The system can check the password format. The password must contain the   
 characters.  
 **SRS-19:** The system validates the username and password.  
 **SRS-20:** The system shall provide the successful login page.  
 **SRS-07**: The system shall connect to the database.  
 **SRS-08**: The system retrieve information from the database.  
 **SRS-21:** The system shall show the user status with a customer picture on the top-right of   
 the user interface to confirm his identity

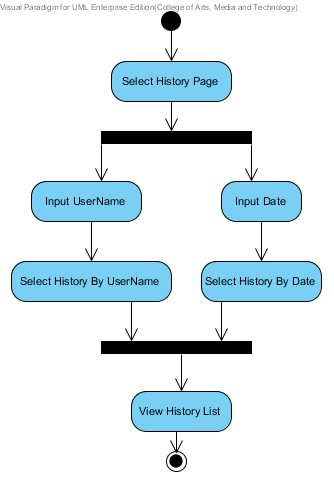
# **URS-12:** Administrator can view the shopping history of all customer.

**Introduction:**  Administrator view the shopping history of all customer. The system shall retrieve the information of customer’s history by using username of customer or Date. After searching, the list of history shall display on screen as, name of product, amount of product that customer purchasing and date that customer purchase a product.

**Input:** Administrator input Username or Date.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Name** | **Type** | **Description** | **Example** | **Remarks** |
| username | String | A username of customer. | “Putsacha” | 1-50 characters |
| date | Date | A date of history. | “2014-02-01” | Format: “yyyy/mm/dd” |

**Flow of Execution:**



**Output:**  The system show all the history list in History Page.

**System Requirement Specification**  
 **SRS-44:** The system provide an interface for the shopping history page.  
 **SRS-45**: The system retrieve shopping history form the database by reference from user.  
 **SRS-49**: The system retrieve shopping history form the database by reference from date.  
 **SRS-50:** The system provide an interface that show all the shopping history of all customer.

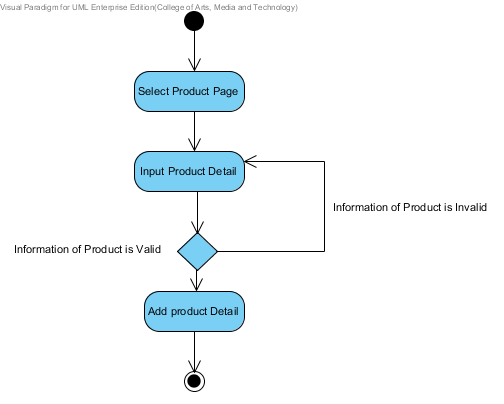
# **URS-13:** Administrator can provide detail of the product.

**Introduction:**  Administrator add a product’s information into the system. Administrator need to provide the detail of product which consist of name, description, price and image. After providing detail, the system shall add a new product into the system completely.

**Input:** Administrator request a Product Page.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Name** | **Type** | **Description** | **Example** | **Remarks** |
| name | String | A name of product. | “channel” | 1-50 characters |
| description | String | A description of product. | “flat” | 1-255 characters |
| price | Double | A price of product. | “110.25” | - |
| Image | BLOB | An image of product. |  | - |

**Flow of Execution:**

****

**Output:**  The System add the product into database.

**System Requirement Specification**  
 **SRS-51:** The system provide an interface for update a product which receive name,   
 description, price and image.  
 **SRS-07**: The system shall connect to the database.  
 **SRS-36**: The system update information into database.

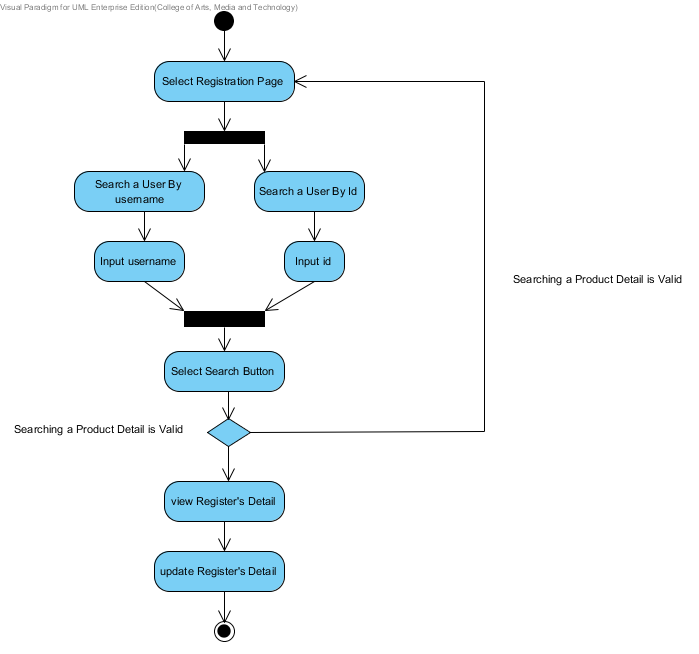
# **URS-14:** Administrator can update all information of registration.

**Introduction:**  Administrator can update a registration of customer into the system. Administrator need to provide a detail of customer which consist of username, password, name, birth date, email, telephone Number, address and their picture. After updating, the system shall display the customer list in Register Page.

**Input:** Administrator request a registration Page.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Name** | **Type** | **Description** | **Example** | **Remarks** |
| username | String | A username of customer | “camt1234” | 1-50 characters |
| password | String | A password of customer. | “camt1234” | 1-50 characters |
| name | String | A Full-Name of customer | “Putsacha” | 1-40 characters |
| birthDate | DATE | A birth date of customer. | “1993-02-25” | Format : yyyy/mm/dd |
| email | String | An email of customer. | “se542115000@vr.camt.info” | - |
| telephoneNumber | String | A telephone number of customer. | “0888888888” | 0-9 digits |
| address | String | An address of customer. | “CAMT” | 1-255 characters |
| picture | BLOB | A picture of customer. | C:\Users\wuthipong\Desktop\User\542115037.jpg | - |

**Flow of Execution:**

****

**Output:**  The System update the registration of customer into database.

**System Requirement Specification**  
 **SRS-52:** The system provide an interface for update the registration which receive   
 username, password, name, address and their picture for the registration.  
 **SRS-07**: The system shall connect to the database.  
 **SRS-36:** The system shall update all information into database.

# **URS-15:** Administrator can logout.

**Introduction:**  Administrator logout off the system. The system shall close the session of administrator for logout off the system.

**Input:** Administrator select a Logout Button.

**Flow of Execution:**

****

**Output:**  Administrator logout off the system.

**System Requirement Specification**  
 **SRS-47:** the system provide an interface for logout.  
 **SRS-48:** The system doesn’t show a user status on top right after logout successful.

# 2. SRS

## 2.1 List SRS

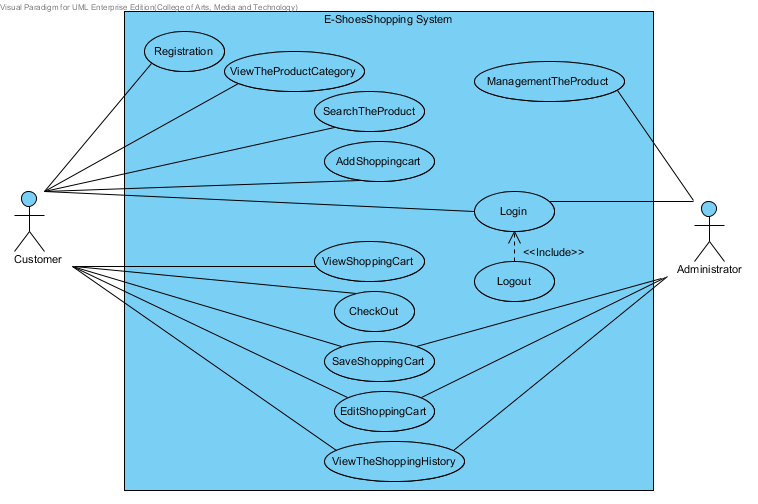
|  |  |
| --- | --- |
| **ID** | **Description** |
| **SRS-01** | The system provide an interface for registration which receive username, password, name address and their picture for the registration. |
| **SRS-02** | The system can check the name format. The name format should be only a character. |
| **SRS-03** | The system can check the address. The address format should be a characters between 1-255 characters. |
| **SRS-04** | The system can check the picture that user uploading. The picture should be an image format such as, .png, .jpg. |
| **SRS-05** | The system can check the username format. The username format should be characters. |
| **SRS-06** | The system can check the password format. The password format should be characters. |
| **SRS-07** | The system shall connect to the database. |
| **SRS-08** | The system retrieve information from the database. |
| **SRS-09** | The system shall add a new customer information to database using username, password, name, address and their picture. |
| **SRS-10** | The system shall provide the successful registration page after registration is complete. |
| **SRS-11** | The system shall display the error message “The username must be characters”. |
| **SRS-12** | The system shall display the error message “The password must be characters”. |
| **SRS-13** | The system shall display the error message “The name must be only characters”. |
| **SRS-14** | The system shall display the error message “The address must be a characters between 1-255 characters”. |
| **SRS-15** | The system shall display the error message “The picture must be image format”. |
| **SRS-16** | The system provides the login interface which receive the user name and password. |
| **SRS-17** | The system can check the username format. The username format should be characters. |
| **SRS-18** | The system can check the password format. The password must contain the characters. |
| **SRS-19** | The system validates the username and password. |
| **SRS-20** | The system shall provide the successful login page. |
| **SRS-21** | The system shall show the user status with a customer picture on the top-right of the user interface to confirm his identity. |
| **SRS-22** | The system provide an interface that show all the product category including, name, description, price and image. |
| **SRS-23** | The system show the product detail that consist of name, description, price and image. |
| **SRS-24** | The system shall provide interface to search a product page which receive product name for searching. |
| **SRS-25** | The system retrieve information from the database by name. |
| **SRS-26** | The system shall provide detail page that consist with product name, product description, and product picture. |
| **SRS-27** | The system shall provide the interface to add new product which user selected. |
| **SRS-28** | The system add the product that user is selected into the shopping cart. |
| **SRS-29** | The system shall update the product that user is selected into the database. |
| **SRS-30** | The system shall provide the successful add new product page after user add new product into the shopping cart. |
| **SRS-31** | The system shall delete the products from the database. |
| **SRS-32** | The system shall update amount of the products into database. |
| **SRS-33** | The system shall display the error message “The system cannot edit the new updating”. |
| **SRS-34** | The system shall display the message “Edit Completed” when updated in the database. |
| **SRS-35** | The system provide an interface for edit function. |
| ID | Decription |
| **SRS-36** | The system shall update information into the database. |
| **SRS-37** | The system shall display error message “The system cannot save the new product”. |
| **SRS-38** | The system shall display error message “The system cannot connect to the database”. |
| **SRS-39** | The system shall display the message “Save Completed” when updated in the database. |
| **SRS-40** | The system provide an interface for save function. |
| **SRS-41** | The system shall provide an interface for Check out function. |
| **SRS-42** | The system shall provide an interface for select the payment option including, Transfer, Credit card and PayPal. |
| **SRS-43** | The system shall update payment option into the database. |
| **SRS-44** | The system shall display message “Purchase successful” when the transect amount from banking system. |
| **SRS-45** | The system provide an interface for the shopping history page. |
| **SRS-46** | The system retrieve shopping history form the database by reference from user. |
| **SRS-47** | The system provide an interface for logout. |
| **SRS-48** | The system doesn’t show a user status on top right after logout successful. |
| **SRS-49** | The system retrieve shopping history form the database by reference from date. |
| **SRS-50** | The system provide an interface that show all the shopping history of all customer. |
| **SRS-51** | The system provide an interface for update a product which receive name, description, price and image. |
| **SRS-52** | The system provide an interface for update the registration which receive username, password, name, address and their picture for the registration. |

# 3. Use case Diagram

## List use cases

|  |  |
| --- | --- |
| **ID** | **Description** |
| **UCD-01** | Registration. |
| **UCD-02** | ViewTheProductCategory. |
| **UCD-03** | SearchTheProduct. |
| **UCD-04** | AddShoppingcart. |
| **UCD-05** | ViewShoppingCart. |
| **UCD-06** | CheckOut. |
| **UCD-07** | SaveShoppingCart. |
| **UCD-08** | EditShoppingCart. |
| **UCD-09** | ViewTheShoppingHistory. |
| **UCD-10** | ManagementTheProduct. |
| **UCD-11** | Login. |
| **UCD-12** | Logout. |

## 3.1 Use case



# 4. Class Diagram

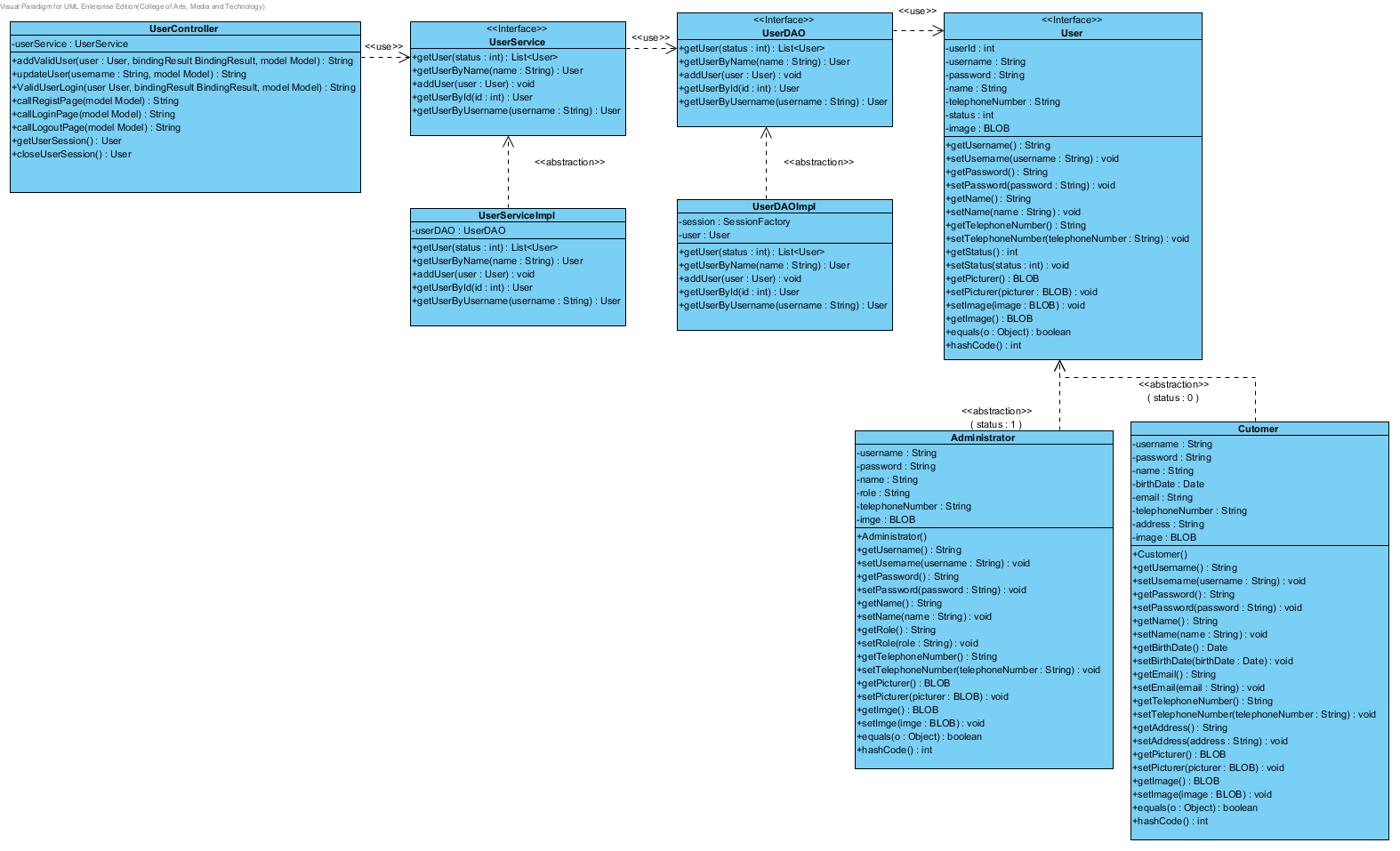
## 4.1 List Class Diagram

|  |  |
| --- | --- |
| **ID** | **Description** |
| **CD-01** | User. |
| **CD-02** | Administrator. |
| **CD-03** | Customer. |
| **CD-04** | Product. |
| **CD-05** | ShoppingCart. |
| **CD-06** | History. |
| **CD-07** | UserDAO. |
| **CD-08** | UserDAOImpl. |
| **CD-09** | UserService. |
| **CD-10** | UserServiceImpl. |
| **CD-11** | UserController. |
| **CD-12** | ProductDAO. |
| **CD-13** | ProductDAOImpl. |
| **CD-14** | ProductService. |
| **CD-15** | ProductServiceImpl. |
| **CD-16** | ProductController. |
| **CD-17** | ShoppingCartDAO. |
| **CD-18** | ShoppingCartDAOImpl. |
| **CD-19** | ShoppingCartService. |
| **CD-20** | ShoppingCartServiceImpl. |
| **CD-21** | ShoppingCartController. |
| **CD-22** | HistoryDAO. |
| **CD-23** | HistoryDAOImpl. |
| **CD-24** | HistoryService. |
| **CD-25** | HistoryServiceImpl. |
| **CD-26** | HistoryController. |

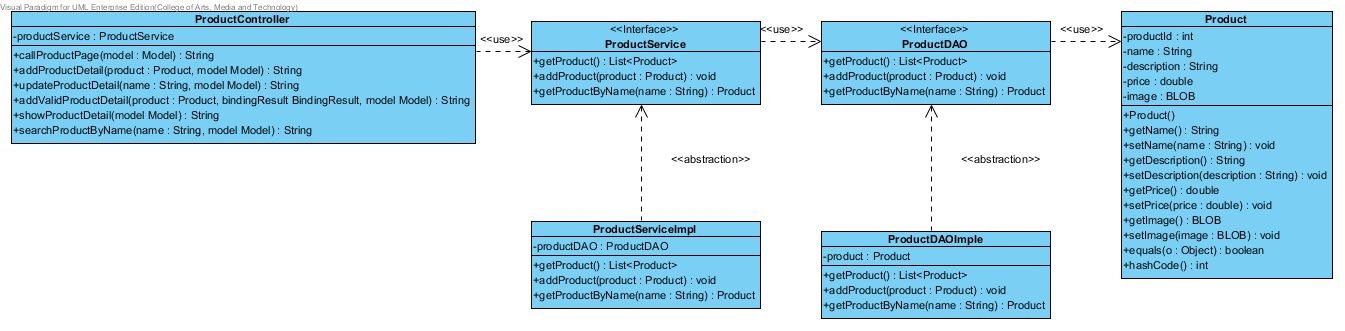
## 4.2 Class Diagram

# 4.2.1 Entity

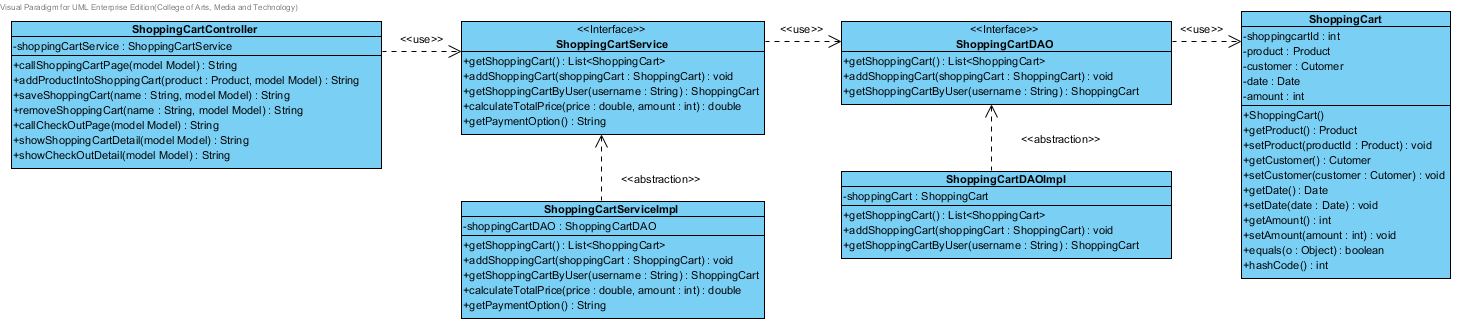
# 4.2.2 MVC of User Controller

******

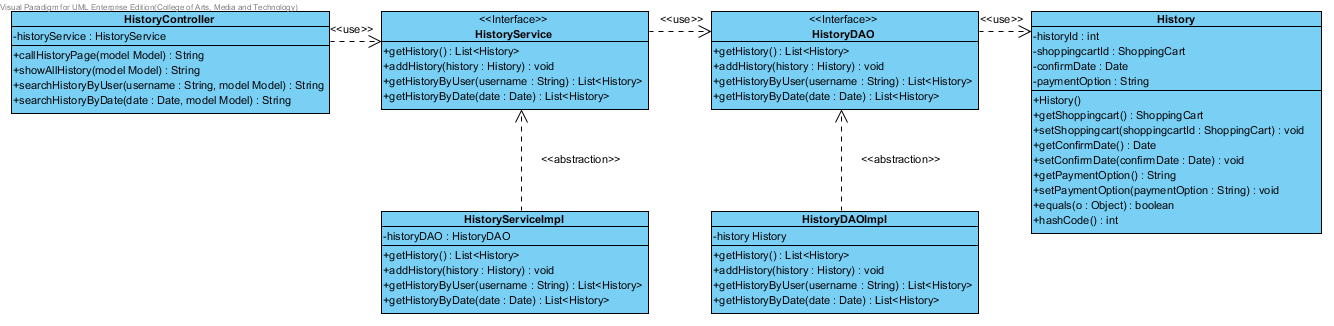
# 4.2.3 MVC of Product Controller

****

# 4.2.4 MVC of Shopping Cart Controller

****

# 4.2.5 MVC of History Controller

****

# 5. Sequence Diagram

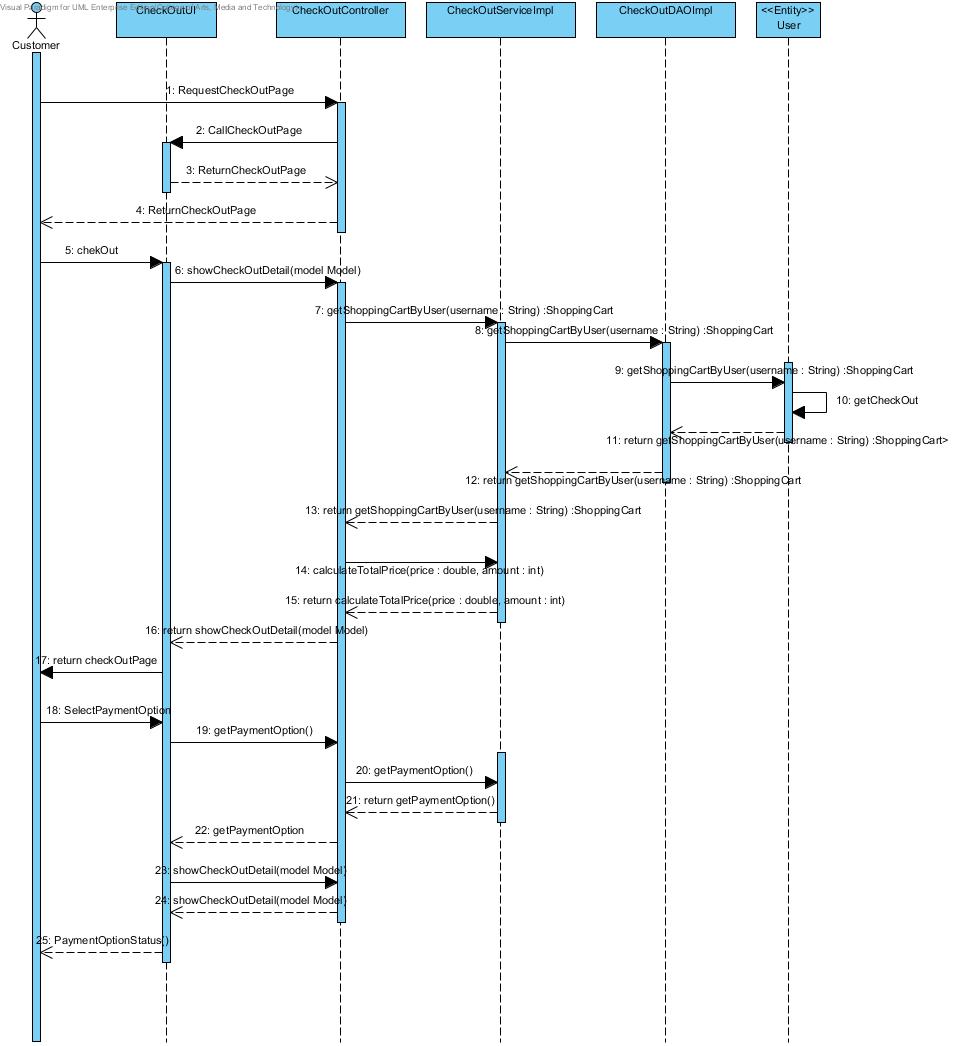
## 5.1 List Sequence Diagram

|  |  |
| --- | --- |
| **ID** | **Description** |
| **SQD-01** | AddProductShoppingCart. |
| **SQD-02** | CheckOut. |
| **SQD-03** | EditProductShoppingCart. |
| **SQD-04** | Registration. |
| **SQD-05** | SaveProductShopping. |
| **SQD-06** | ViewProduct. |
| **SQD-07** | SearchProduct. |
| **SQD-08** | AddProduct. |
| **SQD-09** | UpdateRegisteration. |
| **SQD-10** | Login. |
| **SQD-11** | Logout. |
| **SQD-12** | ViewHistory. |

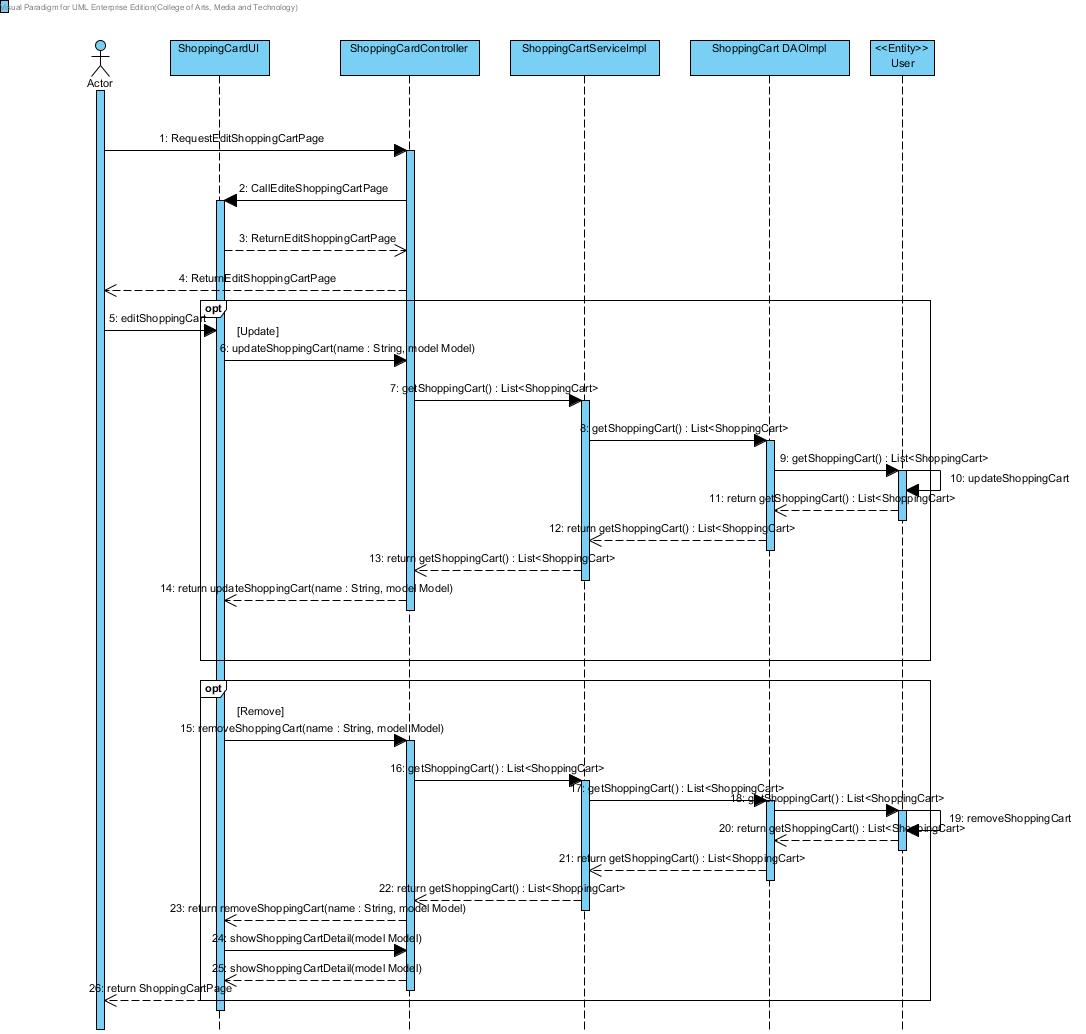
## 5.2 Sequence Diagram

# AddProductShoppingCart5.2.1 SQD-01: Add Product Shopping Cart

# 5.2.2 SQD-02: Check Out

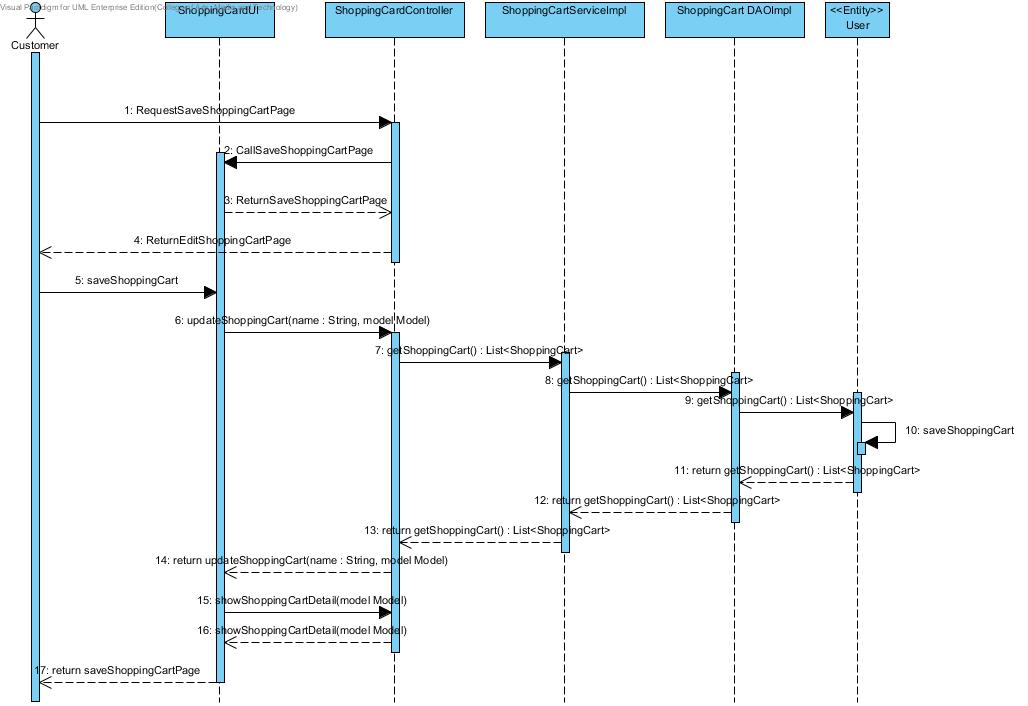
******

# 5.2.3 SQD-03: Edit Product Shopping Cart

******

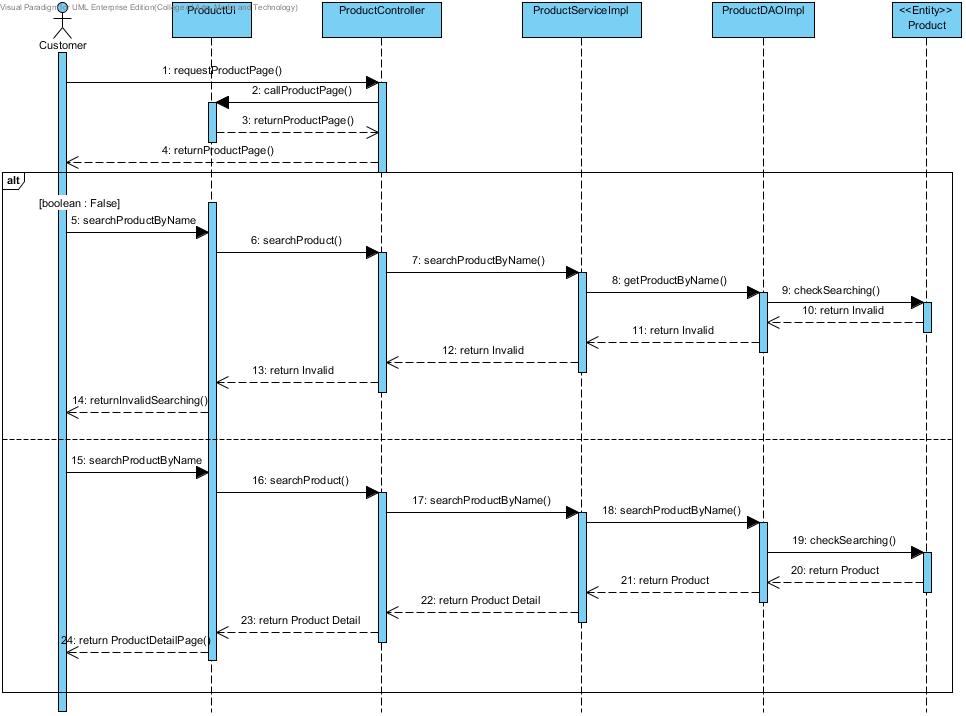
# 5.2.4 SQD-04: RegistrationC:\Users\wuthipong\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Registration.jpg

# 5.2.5 SQD-05: Save Product Shopping Cart

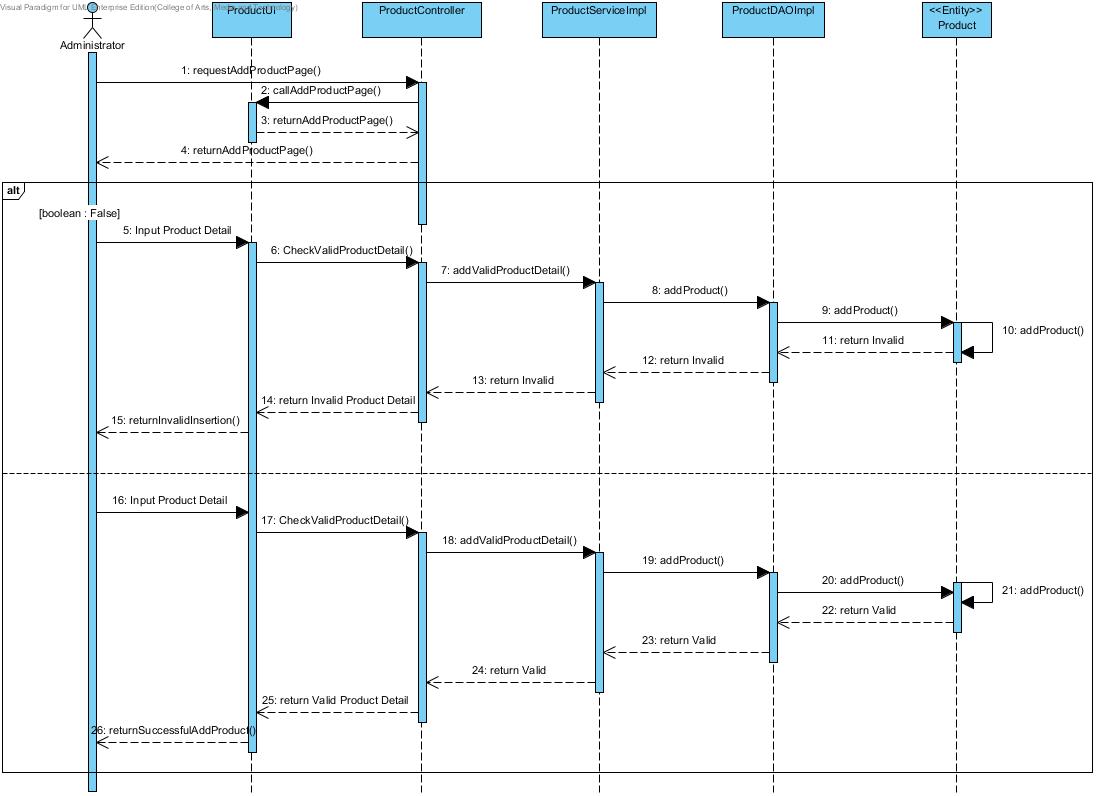
******

# 5.2.6 SQD-06: View ProductViewProduct

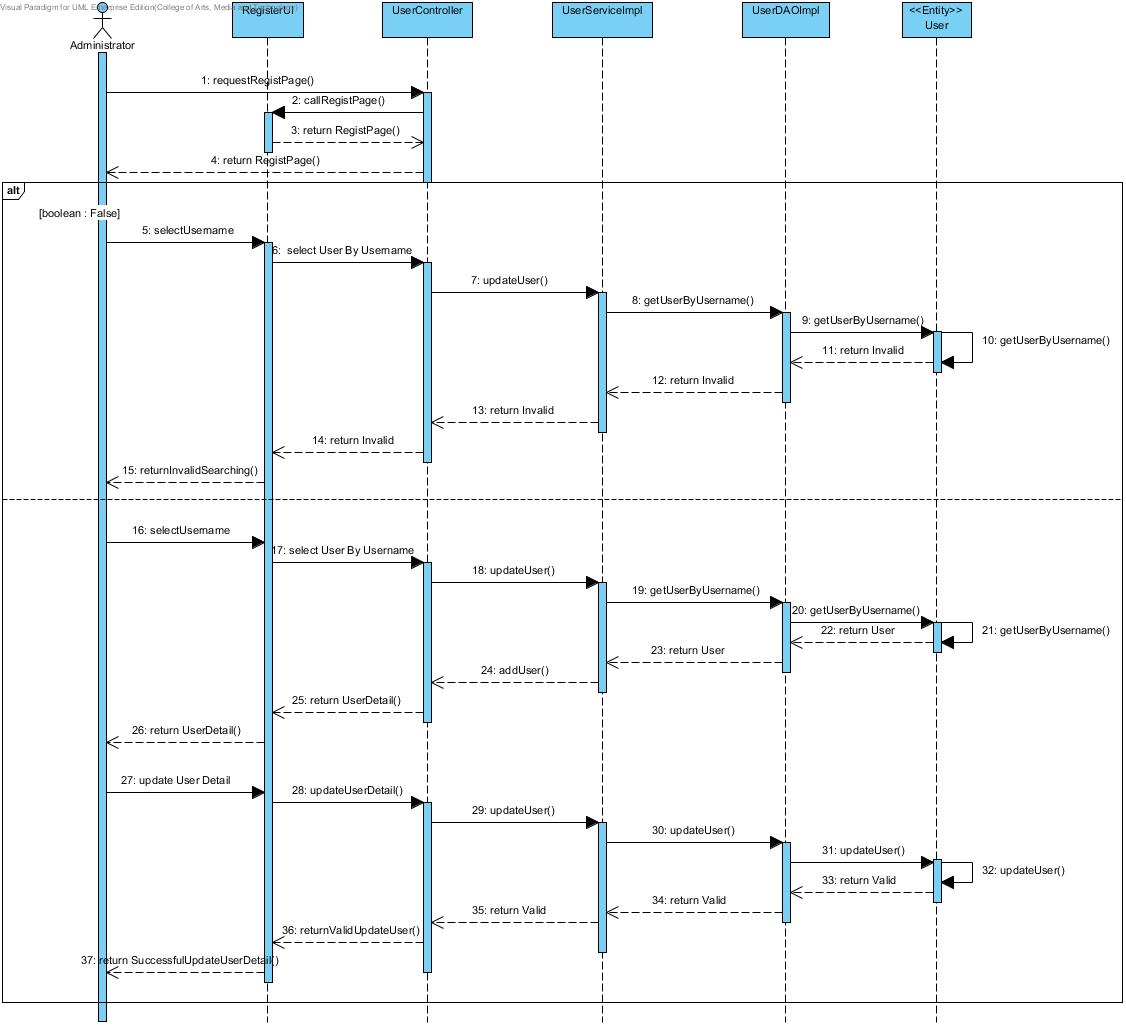
# 5.2.7 SQD-07: Search Product

******

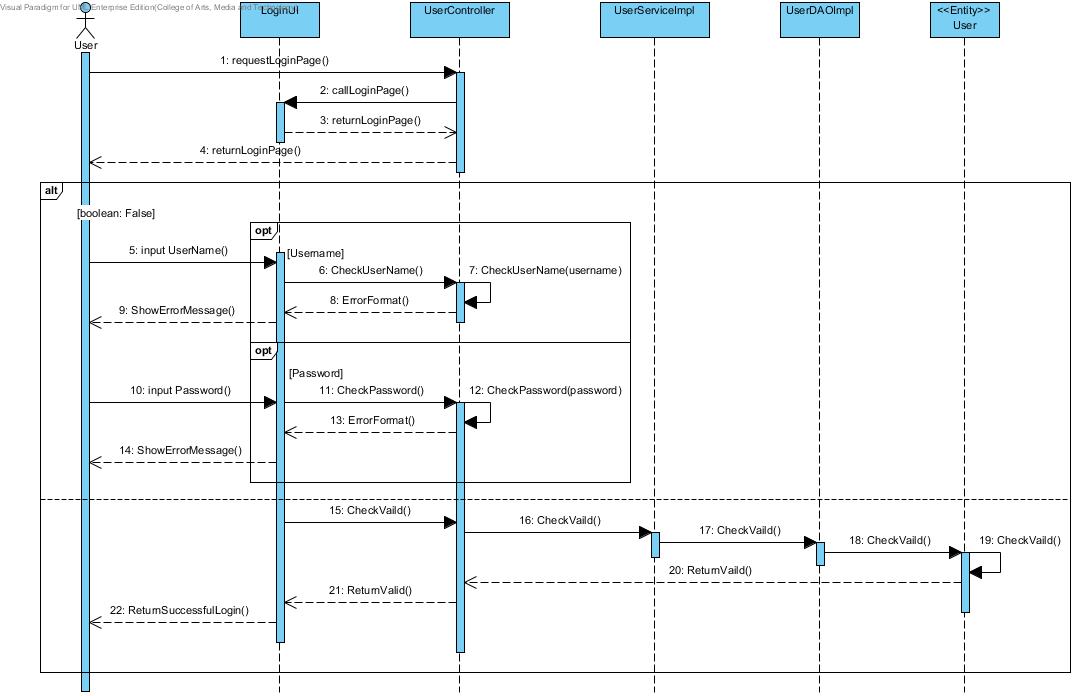
# 5.2.8 SQD-08: Add Product

******

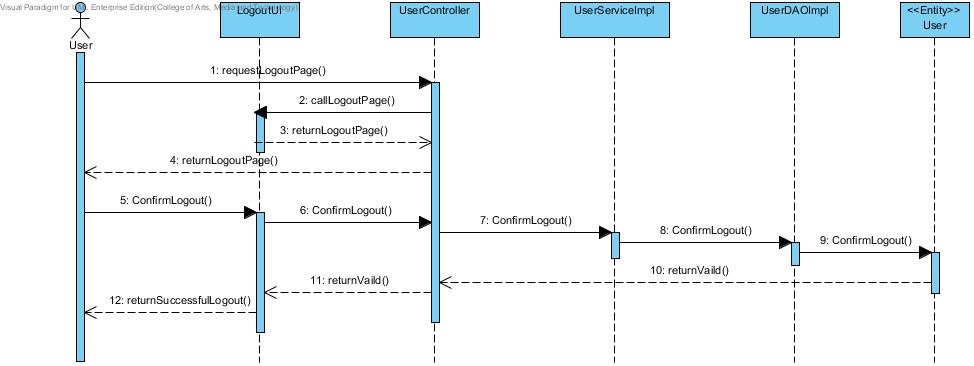
# 5.2.9 SQD-09: Update Registration

******

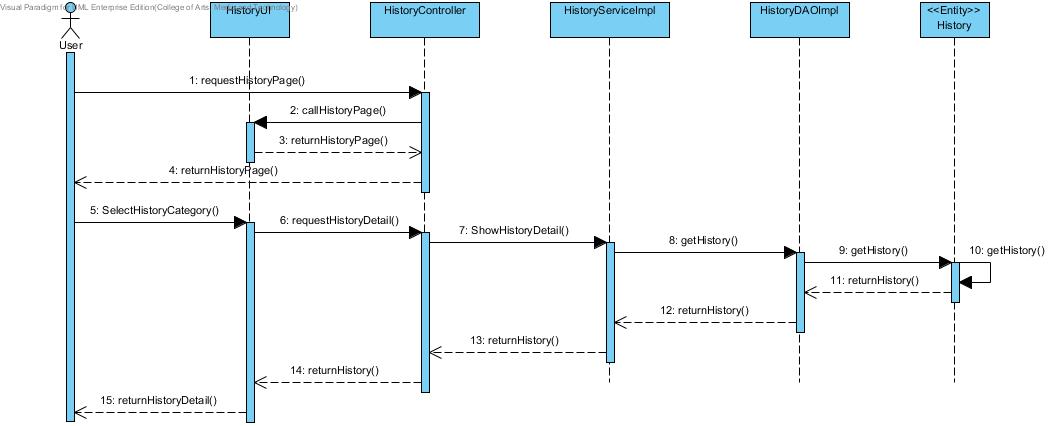
# 5.2.10 SQD-10: Login

******

# 5.2.11 SQD-11: Logout

******

# 5.2.12 SQD-12: View History

******

# 6. Method Design

## 6.1 List method

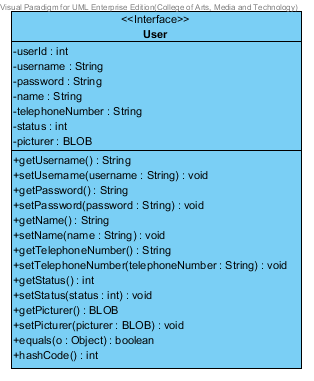
|  |  |
| --- | --- |
| **ID** | **Description** |
| **MTD-01** | User. |
| **MTD-02** | Administrator. |
| **MTD-03** | Customer. |
| **MTD-04** | Product. |
| **MTD-05** | ShoppingCart. |
| **MTD-06** | History. |
| **MTD-07** | UserDAO. |
| **MTD-08** | UserDAOImpl. |
| **MTD-09** | UserService. |
| **MTD-10** | UserServiceImpl. |
| **MTD-11** | UserController. |
| **MTD-12** | ProductDAO. |
| **MTD-13** | ProductDAOImpl. |
| **MTD-14** | ProductService. |
| **MTD-15** | ProductServiceImpl. |
| **MTD-16** | ProductController. |
| **MTD-17** | ShoppingCartDAO. |
| **MTD-18** | ShoppingCartDAOImpl. |
| **MTD-19** | ShoppingCartService. |
| **MTD-20** | ShoppingCartServiceImpl. |
| **MTD-21** | ShoppingCartController. |
| **MTD-22** | HistoryDAO. |
| **MTD-23** | HistoryDAOImpl. |
| **MTD-24** | HistoryService. |
| **MTD-25** | HistoryServiceImpl. |
| **MTD-26** | HistoryController. |

### 6.2 Proper method design

# 6.2.1 Entity

# 6.2.1.1 MTD-01: User

**Class Name: User**

****

**Entities**

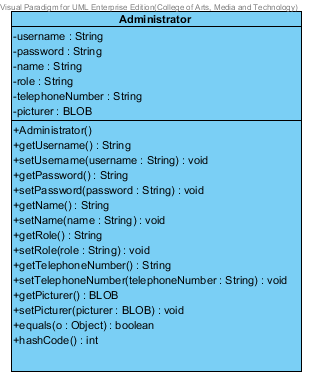
|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **NAME** | **DESCRIPTION** | **REMARK** |
| 1 | userId | This is an id for identify the object of User Class. | **TYPE :** Integer |
| 2 | username | This is a username of user. | **TYPE :** String |
| 3 | password | This is a password of user. | **TYPE :** String |
| 4 | name | This is a Full-name of user. | **TYPE :** String |
| 5 | telephoneNumber | This is a telephone number of user. | **TYPE :** String |
| 6 | status | This is a status to separate the type of User Object. | **TYPE :** Integer |
| 7 | Picture | This is a picture of user. | **TYPE :** BLOB |

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **NAME** | **DESCRIPTION** | **REMARK** |
| 1 | equals() | This method uses to equal the values. | **Return Type :** Boolean  **Parameters :** Object |
| 2 | hashCode() | This method uses to generate the hash value of variables. | **Return Type :** Integer |

**Method**

# 6.2.1.2 MTD-02: Administrator

**Class Name: Administrator**

****

**Entities**

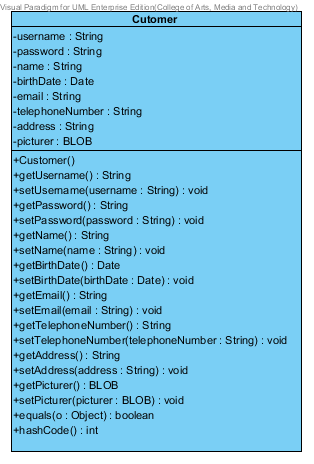
|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **NAME** | **DESCRIPTION** | **REMARK** |
| 1 | username | This is a username of Administrator. | **TYPE :** String |
| 2 | password | This is a password of Administrator. | **TYPE :** String |
| 3 | name | This is a Full-name of Administrator. | **TYPE :** String |
| 4 | role | This is a role that Administrator responsibility. | **TYPE :** String |
| 5 | telephoneNumber | This is a telephone number of Administrator. | **TYPE :** String |
| 6 | Picture | This is a picture of Administrator. | **TYPE :** BLOB |

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **NAME** | **DESCRIPTION** | **REMARK** |
| 1 | equals() | This method uses to equal the values. | **Return Type :** Boolean  **Parameters :** Object |
| 2 | hashCode() | This method uses to generate the hash value of variables. | **Return Type :** Integer |

**Method**

# 6.2.1.3 MTD-03: Customer

**Class Name: Customer**

****

**Entities**

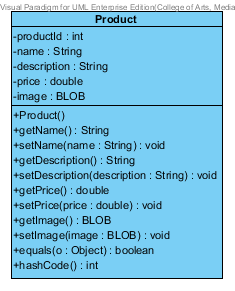
|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **NAME** | **DESCRIPTION** | **REMARK** |
| 1 | username | This is a username of customer. | **TYPE :** String |
| 2 | password | This is a password of customer. | **TYPE :** String |
| 3 | name | This is a Full-name of customer. | **TYPE :** String |
| 4 | birthDate | This is a birth date of customer | **TYPE :** String |
| 5 | email | This is an email of customer. | **TYPE :** String |
| 6 | telephoneNumber | This is a telephone number of customer. | **TYPE :** String |
| 7 | address | This is an address of customer. | **TYPE :** String |
| 8 | Picture | This is a picture of customer. | **TYPE :** BLOB |

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **NAME** | **DESCRIPTION** | **REMARK** |
| 1 | equals() | This method uses to equal the values. | **Return Type :** Boolean  **Parameters :** Object |
| 2 | hashCode() | This method uses to generate the hash value of variables. | **Return Type :** Integer |

**Method**

# 6.2.1.4 MTD-04: Product

**Class Name: Product**

****

**Entities**

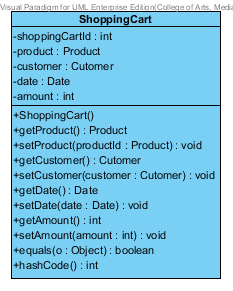
|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **NAME** | **DESCRIPTION** | **REMARK** |
| 1 | productId | This is an id for identify the object of Product Class. | **TYPE :** Integer |
| 2 | Name | This is a name of product. | **TYPE :** String |
| 3 | description | This is a description of product. | **TYPE :** String |
| 4 | Price | This is a price of product | **TYPE :** Double |
| 5 | Image | This is an image of product. | **TYPE :** BLOB |

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **NAME** | **DESCRIPTION** | **REMARK** |
| 1 | equals() | This method uses to equal the values. | **Return Type :** Boolean  **Parameters :** Object |
| 2 | hashCode() | This method uses to generate the hash value of variables. | **Return Type :** Integer |

**Method**

# 6.2.1.5 MTD-05: Shopping Cart

**Class Name: Shopping Cart**

****

**Entities**

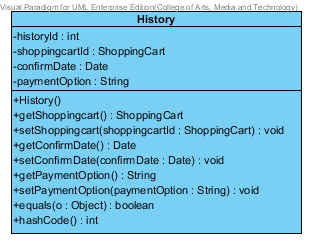
|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **NAME** | **DESCRIPTION** | **REMARK** |
| 1 | shoppingCartId | This is an id for identify the object of shopping Cart Class. | **TYPE :** Integer |
| 2 | product | This is an object to hold product information. | **TYPE :** Product |
| 3 | customer | This is an object to hold customer information. | **TYPE :** Customer |
| 4 | date | This is a date when customer is adding a product into The shopping Cart. | **TYPE :** Date |
| 5 | amount | This is an amount of product which is customer adding into the shopping Cart. | **TYPE :** Integer |

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **NAME** | **DESCRIPTION** | **REMARK** |
| 1 | equals() | This method uses to equal the values. | **Return Type :** Boolean  **Parameters :** Object |
| 2 | hashCode() | This method uses to generate the hash value of variables. | **Return Type :** Integer |

**Method**

# 6.2.1.6 MTD-06: History

**Class Name: History**

****

**Entities**

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **NAME** | **DESCRIPTION** | **REMARK** |
| 1 | historyId | This is an id for identify the object of History Class. | **TYPE :** Integer |
| 2 | shoppingCartId | This is an object to hold shopping Cart information. | **TYPE :** ShoppingCart |
| 3 | confirmDate | This is a date when customer confirm to purchase a product. | **TYPE :** Date |
| 4 | paymentOption | This is a payment Option which consist of transfer, credit card and Paypal. | **TYPE :** String |

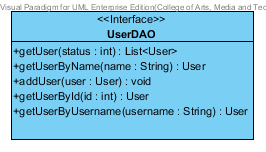
|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **NAME** | **DESCRIPTION** | **REMARK** |
| 1 | equals() | This method uses to equal the values. | **Return Type :** Boolean  **Parameters :** Object |
| 2 | hashCode() | This method uses to generate the hash value of variables. | **Return Type :** Integer |

**Method**

# 6.2.2 MVC Pattern of User Class

# 6.2.2.1 MTD-07: UserDAO

**Class Name: UserDAO**

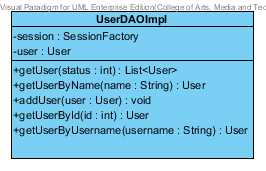
****

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **NAME** | **DESCRIPTION** | **REMARK** |
| 1 | getUser() | This method uses for getting a user. | **Return Type :** List<User>  **Parameters :** status |
| 2 | getUserByName() | This method uses for getting a user by using user’s name from database. | **Return Type :** User  **Parameters :** name |
| 3 | addUser() | This method uses for adding a user into a database. | **Return Type :** void  **Parameters :** user |
| 4 | getUserById() | This method uses for getting a user by using user’s id from database. | **Return Type :** User  **Parameters :** id |
| 5 | getUserByUsername() | This method uses for getting a user by using user’s username from database. | **Return Type :** User  **Parameters :** username |

**Method**

# 6.2.2.2 MTD-08: UserDAOImpl

**Class Name: UserDAOImpl**

****

**Entities**

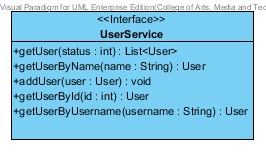
|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **NAME** | **DESCRIPTION** | **REMARK** |
| 1 | session | This is an object to hold sessionFactory. | **TYPE :** SessionFactory |
| 2 | user | This is an object to hold user’s information. | **TYPE :** User |

**Method**

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **NAME** | **DESCRIPTION** | **REMARK** |
| 1 | getUser() | This method uses for getting a user. | **Return Type :** List<User>  **Parameters :** status |
| 2 | getUserByName() | This method uses for getting a user by using user’s name from database. | **Return Type :** User  **Parameters :** name |
| 3 | addUser() | This method uses for adding a user into a database. | **Return Type :** void  **Parameters :** user |
| 4 | getUserById() | This method uses for getting a user by using user’s id from database. | **Return Type :** User  **Parameters :** id |
| 5 | getUserByUsername() | This method uses for getting a user by using user’s username from database. | **Return Type :** User  **Parameters :** username |

# 6.2.2.3 MTD-09: UserService

**Class Name: UserService**

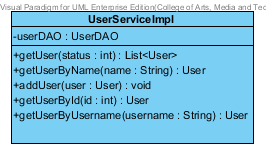
****

**Method**

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **NAME** | **DESCRIPTION** | **REMARK** |
| 1 | getUser() | This method uses for getting a user. | **Return Type :** List<User>  **Parameters :** status |
| 2 | getUserByName() | This method uses for getting a user by using user’s name from database. | **Return Type :** User  **Parameters :** name |
| 3 | addUser() | This method uses for adding a user into a database. | **Return Type :** void  **Parameters :** user |
| 4 | getUserById() | This method uses for getting a user by using user’s id from database. | **Return Type :** User  **Parameters :** id |
| 5 | getUserByUsername() | This method uses for getting a user by using user’s username from database. | **Return Type :** User  **Parameters :** username |

# 6.2.2.4 MTD-10: UserServiceImpl

**Class Name: UserServiceImpl**

****

**Entities**

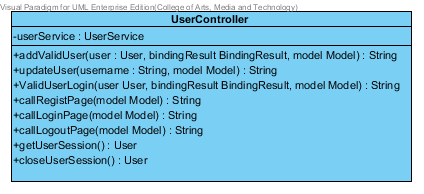
|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **NAME** | **DESCRIPTION** | **REMARK** |
| 1 | userDAO | This is an object to hold userDAO Class. | **TYPE :** UserDAO |

**Method**

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **NAME** | **DESCRIPTION** | **REMARK** |
| 1 | getUser() | This method uses for getting a user. | **Return Type :** List<User>  **Parameters :** status |
| 2 | getUserByName() | This method uses for getting a user by using user’s name from database. | **Return Type :** User  **Parameters :** name |
| 3 | addUser() | This method uses for adding a user into a database. | **Return Type :** void  **Parameters :** user |
| 4 | getUserById() | This method uses for getting a user by using user’s id from database. | **Return Type :** User  **Parameters :** id |
| 5 | getUserByUsername() | This method uses for getting a user by using user’s username from database. | **Return Type :** User  **Parameters :** username |

# 6.2.2.5 MTD-11: UserController

**Class Name: UserController**

****

**Entities**

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **NAME** | **DESCRIPTION** | **REMARK** |
| 1 | userService | This is an object to hold userService Class. | **TYPE :** UserService |

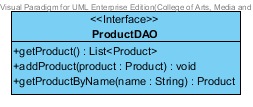
**Method**

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **NAME** | **DESCRIPTION** | **REMARK** |
| 1 | addValidUser() | This method uses for adding a valid user into database. | **Return Type :** String  **Parameters :** user, bindingResult, model |
| 2 | updateUser() | This method uses for update a user into a database. | **Return Type :** String  **Parameters :** username, model |
| 3 | validUserLogin() | This method uses for validate a user when user is login. | **Return Type :** String  **Parameters :** user, bindingResult, model |
| 4 | callRegistPage() | This method uses for connecting with Register Page . | **Return Type :** String  **Parameters :** model |
| 5 | callLoginPage() | This method uses for connecting with Login Page . | **Return Type :** String  **Parameters :** model |
| 6 | callLogoutPage() | This method uses for connecting with Logout Page. | **Return Type :** String  **Parameters :** model |
| 7 | getUserSession() | This method uses to get a user’s session. | **Return Type :** User |
| 8 | closeUserSession() | This method uses to close a user’s session. | **Return Type :** User |

# 6.2.3 MVC Pattern of Product Class

# 6.2.3.1 MTD-12: ProductDAO

**Class Name: ProductDAO**

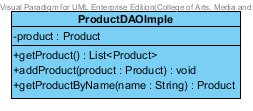
****

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **NAME** | **DESCRIPTION** | **REMARK** |
| 1 | getProduct() | This method uses for getting a product. | **Return Type :** List<Product> |
| 2 | addProduct() | This method uses for adding a product into a database. | **Return Type :** void  **Parameters :** product |
| 3 | getProductByName() | This method uses for getting a product by using product’s name from database. | **Return Type :** User  **Parameters :** name |

**Method**

# 6.2.3.2 MTD-13: ProductDAOImpl

**Class Name: ProductDAOImpl**

****

**Entities**

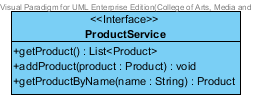
|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **NAME** | **DESCRIPTION** | **REMARK** |
| 1 | product | This is an object to hold product’s information. | **TYPE :** Product |

**Method**

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **NAME** | **DESCRIPTION** | **REMARK** |
| 1 | getProduct() | This method uses for getting a product. | **Return Type :** List<Product> |
| 2 | addProduct() | This method uses for adding a product into a database. | **Return Type :** void  **Parameters :** product |
| 3 | getProductByName() | This method uses for getting a product by using product’s name from database. | **Return Type :** User  **Parameters :** name |

# 6.2.3.3 MTD-14: ProductService

**Class Name: ProductService**

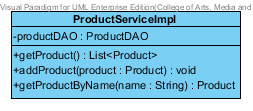
****

**Method**

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **NAME** | **DESCRIPTION** | **REMARK** |
| 1 | getProduct() | This method uses for getting a product. | **Return Type :** List<Product> |
| 2 | addProduct() | This method uses for adding a product into a database. | **Return Type :** void  **Parameters :** product |
| 3 | getProductByName() | This method uses for getting a product by using product’s name from database. | **Return Type :** User  **Parameters :** name |

# 6.2.3.4 MTD-15: ProductServiceImpl

**Class Name: ProductServiceImpl**

****

**Entities**

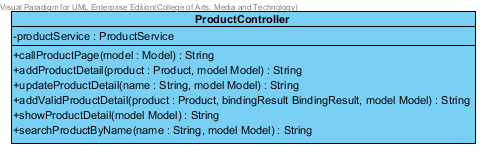
|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **NAME** | **DESCRIPTION** | **REMARK** |
| 1 | productDAO | This is an object to hold ProductDAO Class. | **TYPE :** ProductDAO |

**Method**

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **NAME** | **DESCRIPTION** | **REMARK** |
| 1 | getProduct() | This method uses for getting a product. | **Return Type :** List<Product> |
| 2 | addProduct() | This method uses for adding a product into a database. | **Return Type :** void  **Parameters :** product |
| 3 | getProductByName() | This method uses for getting a product by using product’s name from database. | **Return Type :** User  **Parameters :** name |

# 6.2.3.5 MTD-16: ProductController

**Class Name: ProductController**

****

**Entities**

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **NAME** | **DESCRIPTION** | **REMARK** |
| 1 | productService | This is an object to hold Product Service Class. | **TYPE :** ProductService |

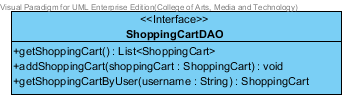
**Method**

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **NAME** | **DESCRIPTION** | **REMARK** |
| 1 | callProductPage() | This method uses for connecting with Product Page. | **Return Type :** String  **Parameters :** model |
| 2 | addProductDetail() | This method uses for adding a product detail into a database. | **Return Type :** String  **Parameters :** product, model |
| 3 | updateProductDetail() | This method uses for update a product when user is login. | **Return Type :** String  **Parameters :** name, model |
| 4 | addValidProductDetail() | This method uses for adding a valid product into database. | **Return Type :** String  **Parameters :** product, bindingResult, model |
| 5 | showProductDetail() | This method uses for showing a product detail into Product Page. | **Return Type :** String  **Parameters :** model |
| 6 | searchProductByName() | This method uses for searching a product by using product’s name from database. | **Return Type :** String  **Parameters :** name, model |

# 6.2.4 MVC Pattern of ShoppingCart Class

# 6.2.4.1 MTD-17: ShoppingCartDAO

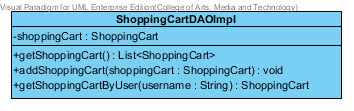
**Class Name: ShoppingCartDAO**

****

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **NAME** | **DESCRIPTION** | **REMARK** |
| 1 | getShoppingCart () | This method uses for getting a Shopping Cart. | **Return Type :** -  **Parameters :** List<ShoppingCart> |
| 2 | addShoppingCart () | This method uses for adding a shopping cart by using shoppingCart. | **Return Type :** void  **Parameters :**shoppingCart |
| 3 | getShoppingCartByUser() | This method uses for getting a ShoppingCart by username. | **Return Type :** ShoppingCart  **Parameters :** username |

# 6.2.4.2 MTD-18: ShoppingCartDAOImpl

**Class Name: ShoppingCartDAOImpl**

****

**Entities**

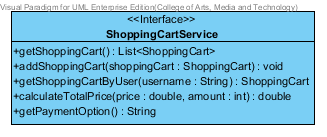
|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **NAME** | **DESCRIPTION** | **REMARK** |
| 1 | shoppingCart | This is an object to hold ShoppingCart. | **TYPE :** ShoppingCart |

**Method**

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **NAME** | **DESCRIPTION** | **REMARK** |
| 1 | getShoppingCart () | This method uses for getting a ShoppingCart. | **Return Type :** List< ShoppingCart >  **Parameters :** status |
| 2 | addShoppingCart() | This method uses for adding ShoppingCart into database. | **Return Type :** Void  **Parameters :** ShoppingCart |
| 3 | getShoppingCartByUser() | This method uses for getting a ShoppingCart by username from a database. | **Return Type :** ShoppingCart  **Parameters :** username |

# 6.2.4.3 MTD-19: ShoppingCartService

**Class Name: ShoppingCartService**

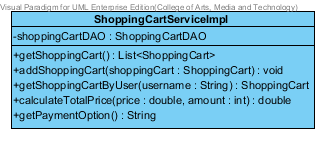
****

**Method**

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **NAME** | **DESCRIPTION** | **REMARK** |
| 1 | getShoppingCart () | This method uses for getting a Shopping Cart. | **Return Type :** List<ShoppingCart>  **Parameters :** status |
| 2 | addShoppingCart () | This method uses for adding a shoppingCart in to a database. | **Return Type :** void  **Parameters :** name |
| 3 | getShoppingCart () | This method uses for adding a user into a database. | **Return Type :** ShoppingCart  **Parameters :** username |
| 4 | calculateTotalPrice() | This method uses for calculate the total price by using a price and amount. | **Return Type :** double  **Parameters :** price, amount |
| 5 | getPaymentOption() | This method uses for getting a payment option detail. | **Return Type :** String  **Parameters :** - |

# 6.2.4.4 MTD-20: ShoppingCartServiceImpl

**Class Name: ShoppingCartServiceImpl**

****

**Entities**

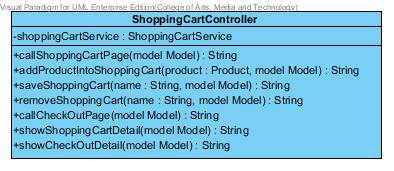
|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **NAME** | **DESCRIPTION** | **REMARK** |
| 1 | shoppingCartDAO | This is an object to hold shoppingCartDAO Class. | **TYPE :** shoppingCartDAO |

**Method**

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **NAME** | **DESCRIPTION** | **REMARK** |
| 1 | getShoppingCart () | This method uses for getting a Shopping Cart. | **Return Type :** List<ShoppingCart>  **Parameters :** status |
| 2 | addShoppingCart () | This method uses for adding a shoppingCart in to a database. | **Return Type :** void  **Parameters :** name |
| 3 | getShoppingCart () | This method uses for adding a user into a database. | **Return Type :** ShoppingCart  **Parameters :** username |
| 4 | calculateTotalPrice() | This method uses for calculate the total price by using a price and amount. | **Return Type :** double  **Parameters :** price, amount |
| 5 | getPaymentOption() | This method uses for getting a payment option detail. | **Return Type :** String  **Parameters :** - |

# 6.2.4.5 MTD-21: ShoppingCartController

**Class Name: ShoppingCartController**

****

**Entities**

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **NAME** | **DESCRIPTION** | **REMARK** |
| 1 | shoppingCartService | This is an object to hold shoppingCartService Class. | **TYPE :** ShoppingCartService |

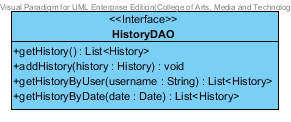
**Method**

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **NAME** | **DESCRIPTION** | **REMARK** |
| 1 | callShoppingCartPage() | This method uses for connecting with shoppingCart page from ShoppingCartUI. | **Return Type :** String  **Parameters :** model |
| 2 | addProductIntoShoppingCart() | This method uses for adding product into a database. | **Return Type :** String  **Parameters :** product, model |
| 3 | saveShoppingCart () | This method uses for adding shoppingCart by using name into database. | **Return Type :** String  **Parameters :** name, model |
| 4 | removeShoppingCart() | This method uses for delete shoppingCart by using name from database. | **Return Type :** String  **Parameters :** name, model |
| 5 | callCheckOutPage() | This method uses for connecting with check Out Page. | **Return Type :** String  **Parameters :** model |
| 6 | showShoppingCartDetail () | This method uses for showing with shopping Cart Page. | **Return Type :** String  **Parameters :** model |
| 7 | showCheckOutDetai() | This method uses for showing with check out page | **Return Type :** String  **Parameters :** model |

# 6.2.5 MVC Pattern of History Class

# 6.2.5.1 MTD-22: HistoryDAO

**Class Name: HistoryDAO**

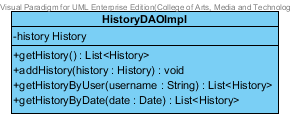
****

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **NAME** | **DESCRIPTION** | **REMARK** |
| 1 | getHistory() | This method uses for getting a history. | **Return Type :** List<History> |
| 2 | addHistory() | This method uses for adding a history into a database. | **Return Type :** void  **Parameters :** history |
| 3 | getHistoryByName() | This method uses for getting a history list by using user’s name from database. | **Return Type :** List<History>  **Parameters :** username |
| 4 | getHistoryByDate() | This method uses for getting a history list by using date from database. | **Return Type :** List<History>  **Parameters :** date |

**Method**

# 6.2.5.2 MTD-23: HistoryDAOImpl

**Class Name: HistoryDAOImpl**

****

**Entities**

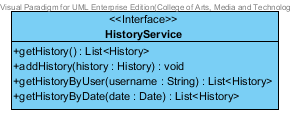
|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **NAME** | **DESCRIPTION** | **REMARK** |
| 1 | history | This is an object to hold history’s information. | **TYPE :** History |

**Method**

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **NAME** | **DESCRIPTION** | **REMARK** |
| 1 | getHistory() | This method uses for getting a history. | **Return Type :** List<History> |
| 2 | addHistory() | This method uses for adding a history into a database. | **Return Type :** void  **Parameters :** history |
| 3 | getHistoryByName() | This method uses for getting a history list by using user’s name from database. | **Return Type :** List<History>  **Parameters :** username |
| 4 | getHistoryByDate() | This method uses for getting a history list by using date from database. | **Return Type :** List<History>  **Parameters :** date |

# 6.2.5.3 MTD-24: HistoryService

**Class Name: HistoryService**

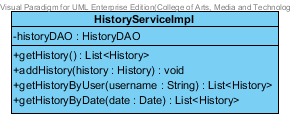
****

**Method**

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **NAME** | **DESCRIPTION** | **REMARK** |
| 1 | getHistory() | This method uses for getting a history. | **Return Type :** List<History> |
| 2 | addHistory() | This method uses for adding a history into a database. | **Return Type :** void  **Parameters :** history |
| 3 | getHistoryByName() | This method uses for getting a history list by using user’s name from database. | **Return Type :** List<History>  **Parameters :** username |
| 4 | getHistoryByDate() | This method uses for getting a history list by using date from database. | **Return Type :** List<History>  **Parameters :** date |

# 6.2.5.4 MTD-25: HistoryServiceImpl

**Class Name: HistoryServiceImpl**

****

**Entities**

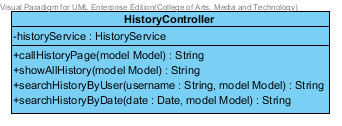
|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **NAME** | **DESCRIPTION** | **REMARK** |
| 1 | historyDAO | This is an object to hold HistoryDAO Class. | **TYPE :** HistoryDAO |

**Method**

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **NAME** | **DESCRIPTION** | **REMARK** |
| 1 | getHistory() | This method uses for getting a history. | **Return Type :** List<History> |
| 2 | addHistory() | This method uses for adding a history into a database. | **Return Type :** void  **Parameters :** history |
| 3 | getHistoryByName() | This method uses for getting a history list by using user’s name from database. | **Return Type :** List<History>  **Parameters :** username |
| 4 | getHistoryByDate() | This method uses for getting a history list by using date from database. | **Return Type :** List<History>  **Parameters :** date |

# 6.2.5. MTD-26: HistoryController

**Class Name: HistoryController**

****

**Entities**

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **NAME** | **DESCRIPTION** | **REMARK** |
| 1 | historyService | This is an object to hold History Service Class. | **TYPE :** HistoryService |

**Method**

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **NAME** | **DESCRIPTION** | **REMARK** |
| 1 | callHistoryPage() | This method uses for connecting with History Page. | **Return Type :** String  **Parameters :** model |
| 2 | showAllHistory() | This method uses for showing all history list into History Page. | **Return Type :** String  **Parameters :** model |
| 3 | searchHistoryByUser() | This method uses for searching a history list by using user’s name from database. | **Return Type :** String  **Parameters :** name, model |
| 4 | searchHistoryByDate() | This method uses for searching a product by using date from database. | **Return Type :** String  **Parameters :** date, model |

# Traceability

## Traceability matric between URS/SRS

## Traceability matric between URS/Class Diagram

## Traceability matric between URS/Use cases Diagram

## Traceability matric between URS/Sequence Diagram

## Traceability matric between URS/Method designs

## Traceability matric between SRS/Use cases Diagram

## Traceability matric between SRS/Class Diagram

## Traceability matric between SRS/Sequence Diagram

## Traceability matric between SRS/Method designs

## Traceability matric between Use cases Diagram / Class Diagram

## Traceability matric between Use cases Diagram /Sequence Diagram

## Traceability matric between Use cases Diagram /Method designs

## Traceability matric between Class Diagram/Sequence Diagram

## Traceability matric between Class Diagram/Method designs

## Traceability matric between Sequence Diagram/Method designs