# *1. USER REQUIREMENT*

## 1.1 List

|  |  |
| --- | --- |
| **URS** | **Function Name** |
| URS-01 | The customer can register to the system. |
| URS-02 | The customer can login to the system. |
| URS-03 | The customer can update their account information. |
| URS-04 | The customer can search for the product name. |
| URS-05 | The customer can browse the product from product catalog. |
| URS-06 | The customer can add the product to cart. |
| URS-07 | The customer can identify amount of the product. |
| URS-08 | The customer can view the shopping cart. |
| URS-09 | The customer can save the shopping cart. |
| URS-10 | The customer can remove the product from shopping cart. |
| URS-11 | The customer can select payment option. |
| URS-12 | The customer can checkout the product. |
| URS-13 | The customer can view the shopping history. |
| URS-14 | The customer can logout from the system. |
| URS-15 | The administrator can login to the system. |
| URS-16 | The administrator can provide details of the product. |
| URS-17 | The administrator can view the shopping history of all customers. |
| URS-18 | The administrator can logout from the system. |

## 1.2 User Requirement Specification

**URS-01: The customer can register their information to the system**

***Description:***

The customer can register their information into the database.

***Precondition:***

The customer shall open the website and appear the index page.

***Input:***

|  |  |  |
| --- | --- | --- |
| ***Name*** | ***Type*** | ***Remarks*** |
| Name | String | Name must be only character and its length must less than 30 character. |
| Surname | String | Surname must be only character and its length must less than 30 character. |
| Address | String | - |
| Email | String | Name must be only character and its length must less than 30 character. |
| Username | String | Username must be at least 4 characters. |
| Password | String | Password length should be 4-20 character. |
| Picture | BLOB | Picture format must be only .jpg, .jpeg, and .png. |

***Execution flow:***

1. The customer type the website URL.

2. The system shall provide the index page on the customer.

3. The customer clicks on “register” menu at the index page.

4. The system shall provide the register page on the customer.

5. The customer type their information consist of name, surname, address, email, picture, username and password.

6. The system validates the format of the input data.

7. The customer clicks on “Submit” button on the register page to confirm the registration.

***Alternative flow:***

6a) If the user input data in the wrong format, the system should provide the error message as followed:

The wrong name format: “The name must be only characters” and “The name length must less than 30 characters”

The wrong surname format: “The surname must be only characters” and “The surname length must less than 30 characters”

The wrong username format: “The username must be at least 4 characters”

The wrong password format: “The password length should be 4-20 character” and “The password must contain the small letter, capital letter, and number.

The wrong picture format: “The picture must be the picture format”

***Output:***

The customers’ information is added into the users’ database.

***System Requirement Specification:***

SRS-01: The system shall provide the registration interface, which receives the name, surname, address, username, password, and picture for register to the system.

SRS-02: The system shall add customer information to the database using name, surname, address, username, password, and picture.

SRS-03: The system shall provide the successful registration page.

SRS-09: The system can check the name format. The name format should be only character.

SRS-10: The system can check the surname format. The surname format should be only character.

SRS-11: The system can check the username format. The username format should be at least 4 characters.

SRS-12: The system can check the password format. The password format must be 4-20 characters. The password must contain small letter, capital letter, and number.

SRS-13: The system can check the picture format. The picture format must be only .jpg, .jpeg, and .png.

SRS-14: The system shall display the error message. “The name must be only characters”

SRS-15: The system shall display the error message. “The surname must be only characters”

SRS-16: The system shall display the error message. “The name length must less than 30 characters”

SRS-17: The system shall display the error message. “The surname length must less than 30 characters”

SRS-18: The system shall display the error message. “The username must be at least 4 characters”

SRS-19: The system shall display the error message. “The password length should be 4-20 character”

SRS-20: The system shall display the error message. “The password must contain the small letter, capital letter, and number.

SRS-23: The system shall display the error message. “The picture must be the picture format”

**URS-02: The customer can login to the system.**

***Description:***

The customer can use their username and password in order to access to the system.

***Precondition:***

The customers have to register with their information already.

***Input:***

|  |  |  |
| --- | --- | --- |
| ***Name*** | ***Type*** | ***Remarks*** |
| Username | String | Username must be at least 4 characters. |
| Password | String | Password length should be 4-20 character. |

***Execution flow:***

1). The customer type the website URL.

2). The system shall provide the index page to the customer.

3). The customer clicks on “Login” menu on the index page.

4). The system shall provide the login page to the customer.

5). The customer type username and password in the login page.

6). The customer clicks on “Login” button to submit.

7). The system validates the user name and password.

8). The system shall redirect to the index page.

***Alternative flow:***

7a) If the username or password are not matched, the system shows the error message “The username is incorrect” and “The password is incorrect”.

***Output:***

The customer views their picture at the top-right on the index page to confirm that customer has login already.

***System Requirement Specification:***

SRS-06: The system shall provide the login user interface, which receives the username and password.

SRS-07: The system shall provide the successful login page.

SRS-08: The system shall validate the username and password.

SRS-21: The system shall display the error message. “The username is incorrect”

SRS-22: The system shall display the error message. “The password is incorrect”

**URS-03: The customer can update their account information.**

***Description:***

The customer can edit their information and change the picture on the edit account page.

***Precondition:***

The customers have to register and login already.

***Input:***

|  |  |  |
| --- | --- | --- |
| ***Name*** | ***Type*** | ***Remarks*** |
| Name | String | Name must be only character and its length must less than 30 character. |
| Surname | String | Surname must be only character and its length must less than 30 character. |
| Address | String | - |
| Email | String | Name must be only character and its length must less than 30 character. |
| Username | String | Username must be at least 4 characters. |
| Password | String | Password length should be 4-20 character. |
| Picture | BLOB | Picture format must be only .jpg, .jpeg, and .png. |

***Execution flow:***

1. The customer type the website URL.

2. The customer login successful.

3. The system shall redirect to the index page to the customer.

4. The customer clicks on “My account” menu on the index page.

5. The system shall provide the customers’ information on the account page.

6. The customer clicks on “Edit account” button on the account page.

7. The system shall provide the edit account page to the customer.

8. The customer edit their information consist of name, surname, address, email, picture, username and password.

9. The system validates the format of the input data.

***Alternative flow:***

9a) If the user input data in the wrong format, the system should provide the error message as followed:

The wrong name format: “The name must be only characters” and “The name length must less than 30 characters”

The wrong surname format: “The surname must be only characters” and “The surname length must less than 30 characters”

The wrong username format: “The username must be at least 4 characters”

The wrong password format: “The password length should be 4-20 character” and “The password must contain the small letter, capital letter, and number.

The wrong picture format: “The picture must be the picture format”

***Output:***

The system shall update the customers’ account into the database.

***System Requirement Specification***

SRS-09: The system can check the name format. The name format should be only character.

SRS-10: The system can check the surname format. The surname format should be only character.

SRS-11: The system can check the username format. The username format should be at least 4 characters.

SRS-12: The system can check the password format. The password format must be 4-20 characters. The password must contain small letter, capital letter, and number.

SRS-13: The system can check the picture format. The picture format must be only .jpg, .jpeg, and .png.

SRS-14: The system shall display the error message. “The name must be only characters”

SRS-15: The system shall display the error message. “The surname must be only characters”

SRS-16: The system shall display the error message. “The name length must less than 30 characters”

SRS-17: The system shall display the error message. “The surname length must less than 30 characters”

SRS-18: The system shall display the error message. “The username must be at least 4 characters”

SRS-19: The system shall display the error message. “The password length should be 4-20 character”

SRS-20: The system shall display the error message. “The password must contain the small letter, capital letter, and number.

SRS-23: The system shall display the error message. “The picture must be the picture format”

SRS-24: The system shall provide the Edit customer information interface for edit their own information after customer has been logged in to the system.

SRS-25: The system shall display the picture of customer on top-right of the user interface.

SRS-51: The system shall update information of customer to database.

**URS-04: The customer can search for the product name.**

***Description:***

The customer can search the product from search box on the website.

***Precondition:***

The customers have to register and login already.

***Input:***

|  |  |  |
| --- | --- | --- |
| ***Name*** | ***Type*** | ***Remarks*** |
| Product name | String | Product name must be at least 1 character. |

***Execution flow:***

1). The customer type the website URL.

2). The customer login successful.

3). The system shall redirect to the index page to the customer.

4). The customer types a product name in search box at the top-right on the index page.

5). The customer clicks on “Search” button to search the products result.

***Alternative flow:***

4a) If the search empty, the system will shows no product.

***Output:***

The customers view the products’ information follow the products name that customer want to search on the product page.

***System Requirement Specification:***

SRS-25: The system shall display the picture of customer on top-right of the user interface.

SRS-27: The system shall provide the search box for search the product name.

SRS-28: The system shall display the result of search the product name.

SRS-49: The system shall retrieve the product information from the product name.

**URS-05: The customer can browse the product from products catalogs.**

***Description:***

The customer can select the product from products catalogs on the index page.

***Precondition:***

The customers have to register and login already.

***Input:***

|  |  |  |
| --- | --- | --- |
| ***Name*** | ***Type*** | ***Remarks*** |
| catalogs | button | Click left on button |

***Execution flow:***

1). The customer type the website URL.

2). The system shall provide the index page to the customer.

3). The customer login successful.

4). The system shall redirect to the index page to the customer.

5). The customer selects a product from product catalogs.

6). The system shall add the products’ information in the shopping carts’ database.

7). The system shall provide the product description page to the customer.

***Alternative flow:***

5a) If the administrator don’t provide the product, In product catalogs don’t have the product.

***Output:***

The system shall display the product name, picture, and price in the product catalogs.

***System Requirement Specification:***

SRS-25: The system shall display the picture of customer on top-right of the user interface.

SRS-26: The system shall display the product catalogs.

**URS-06: The customer can add the productsto cart.**

***Description:***

The customer can select a product and add to cart.

***Precondition:***

The customers have to register and login already.

***Input:***

|  |  |  |
| --- | --- | --- |
| ***Name*** | ***Type*** | ***Remarks*** |
| Add to cart | button | Click left on button. |

***Execution flow:***

1). The customer type the website URL.

2). The system shall provide the index page to the customer.

3). The customer login successful.

4). The system shall redirect to the index page to the customer.

5). The customer selects a product.

6). The system shall provide the product description to customer.

7). The customer clicks on “Add to cart” button to submit the product.

***Alternative flow:***

5a) If the administrator don’t provide the product, the customer can’t selects a product.

***Output:***

The customer can view the product list on the shopping cart page.

***System Requirement Specification:***

SRS-04: The system shall provide the user interface, which receives the name of product, description, and picture of product.

SRS-05: The system shall add the product information to the database using name, description, and picture of product

SRS-13: The system can check the picture format. The picture format must be only .jpg, .jpeg, and .png.

SRS-23: The system shall display the error message. “ The picture must be the picture format”

SRS-25: The system shall display the picture of customer on top-right of the user interface.

SRS-31: The system shall provide the shopping cart interface.

SRS-36: The system shall provide the user interface for confirm of buying.

**URS-07: The customer can identify amount of the product at least one.**

***Description:***

The customer can select the amount of each product in the product description page.

***Precondition:***

The customers have to register and login already.

***Input:***

|  |  |  |
| --- | --- | --- |
| ***Name*** | ***Type*** | ***Remarks*** |
| Amount | int | Amount number must be at least 1. |

***Execution flow:***

1). The customer type the website URL.

2). The system shall provide the index page to the customer.

3). The customer login successful.

4). The system shall redirect to the index page to the customer.

5). The customer selects a product.

6). The system shall provide the product description page to the customer.

7). The customer identifies the amount of products.

8). The customer clicks on “Add to cart” to confirm the amount.

***Alternative flow:***

5a) If the administrator don’t provide the product, the customer can’t selects a product.

7a) If the product not enough for amount that customer need, the customer can’t add the product to cart.

***Output:***

The customer views correctly amount of each product on the shopping cart page.

***System Requirement Specification:***

SRS-25: The system shall display the picture of customer on top-right of the user interface.

SRS-31: The system shall provide the shopping cart interface.

SRS-35: The system shall provide the user interface for receive amount of product.

SRS-36: The system shall provide the user interface for confirm of buying.

**URS-08: The customer can view the customers’ shopping cart.**

***Description:***

The customer can view their customers’ s shopping cart, which is consist of product name, product description, amount and price.

***Precondition:***

The customers have to select the product to the cart.

***Input:***

|  |  |  |
| --- | --- | --- |
| ***Name*** | ***Type*** | ***Remarks*** |
| Amount | int | Amount number must be at least 1. |

***Execution flow:***

1). The customer type the website URL.

2). The system shall provide the index page to the customer.

3). The customer login successful.

4). The system shall redirect to the index page to the customer.

5). The customer selects a product from product catalogs.

6). The system shall provide the product description page to the customer.

7). The customer identifies the amount of products.

8). The customer clicks on “Add to cart” to confirm the amount.

9). The system shall redirect to the shopping cart page to the customer.

***Alternative flow:***

5a) If the administrator don’t provide the product, the customer can’t selects a product.

8a) If the product not enough for amount that customer need, the customer can’t add the product to cart.

***Output:***

The customer views their shopping cart list on the shopping cart page.

***System Requirement Specification:***

SRS-25: The system shall display the picture of customer on top-right of the user interface.

SRS-31: The system shall provide the shopping cart interface.

SRS-35: The system shall provide the user interface for receive amount of product.

SRS-36: The system shall provide the user interface for confirm of buying.

**URS-09: The customer can save the shopping cart.**

***Description:***

After the customer logout from the website. The customer can continue their shopping when they login again.

***Precondition:***

The customer must have at least one product in their shopping cart list.

***Input:***

|  |  |  |
| --- | --- | --- |
| ***Name*** | ***Type*** | ***Remarks*** |
| save | button | Click left on button |

***Execution flow:***

1). The customer type the website URL.

2). The system shall provide the index page to the customer.

3). The customer login successful.

4). The system shall set the session from customers’ id in each login process.

5). The system shall redirect to the index page to the customer.

***Output:***

The customer can continue shopping with the old shopping list when they login again.

***System Requirement Specification:***

SRS-25: The system shall display the picture of customer on top-right of the user SRS-43: The system shall display the date that customer buy product.

SRS-44: The system shall provide the user interface for save the shopping cart.

SRS-45: The system shall save shopping cart history.

SRS-46: The system shall display the successful shopping cart history interface.

SRS-48: The system shall add history of customer to database using username, name of product, picture of product, amount of product, total price, payment option, and date.

SRS-49: The system shall retrieve the product information from the product name.

**URS-10: The customer can remove the product from shopping cart.**

***Description:***

The customer can remove a product from shopping cart list in the shopping cart page.

The customer must have at least one product in their shopping cart list.

***Input:***

|  |  |  |
| --- | --- | --- |
| ***Name*** | ***Type*** | ***Remarks*** |
| Product name | box | Click left on product box that the customer want to remove. |
| remove | button | Click left on button |

***Execution flow:***

1). The customer type the website URL.

2). The system shall provide the index page to the customer.

3). The customer login successful.

4). The system shall redirect to the index page to the customer.

5). The customer selects a product from product catalogs.

6). The system shall provide the product description page to the customer.

7). The customer identifies the amount of products.

8). The customer clicks on “Add to cart” to confirm the amount.

9). The system shall redirect to the shopping cart page to the customer

10). The customer selects the product that they want to remove from shopping cart list.

11). The customer clicks on “Remove from list” button to remove the products.

12). The system shall redirect to shopping cart page to the customer.

***Alternative flow:***

10a) If the shopping cart don’t have the product, the customer can't remove the product in the shopping cart.

***Output:***

The product that customer select to remove from list was deleted from shopping cart list in the shopping cart page.

***System Requirement Specification:***

SRS-25: The system shall display the picture of customer on top-right of the user interface.

SRS-32: The system shall remove the selected product from the shopping cart database.

SRS-52: The system shall provide the user interface for remove the shopping cart.

**URS-11: The customer can select payment option.**

***Description:***

The customer can select payment option after they want to check out the product from the shopping cart page.

***Precondition:***

The customer must select the product and add a product to cart at least 1 items.

***Input:***

|  |  |  |
| --- | --- | --- |
| ***Name*** | ***Type*** | ***Remarks*** |
| Payment | String | Payment must be only character format. |

***Execution flow:***

1). The customer type the website URL.

2). The system shall provide the index page to the customer.

3). The customer login successful.

4). The system shall redirect to the index page to the customer.

5). The customer selects a product from product catalogs.

6). The system shall provide the product description page to the customer.

7). The customer identifies the amount of products.

8). The customer clicks on “Add to cart” to confirm the amount.

9). The customer selects the payment option.

10). The system shall provide the shopping cart page to the customer.

***Alternative flow:***

9a) If the shopping cart don’t have product, the customer can’t selects the payment option.

***Output:***

The customer views the payment option of each purchase in their history page.

***System Requirement Specification:***

SRS-25: The system shall display the picture of customer on top-right of the user interface.

SRS-33: The system shall provide the payment user interface, which receive the payment option (money transfer, credit cart and PayPal) that the customer wants to use.

SRS-34: The system shall display the payment option that the customer wants to use.

**URS-12: The customer can checkout the products.**

***Description:***

The customer can check out the product after they add a product to cart finish.

***Precondition:***

The customer must have at least one product in their shopping cart list.

***Input:***

|  |  |  |
| --- | --- | --- |
| ***Name*** | ***Type*** | ***Remarks*** |
| Check out | button | Click left on button |

***Execution flow:***

1). The customer type the website URL.

2). The system shall provide the index page to the customer.

3). The customer login successful.

4). The system shall redirect to the index page to the customer.

5). The customer selects a product from product catalogs.

6). The system shall provide the product description page to the customer.

7). The customer identifies the amount of products.

8). The customer clicks on “Add to cart” to confirm the amount.

9). The system shall provide the checkout page to the customer.

10). The customer selects the payment option.

11). The customer clicks on “Check out” button to submit the purchase.

12). The system shall redirect to the history page.

***Alternative flow:***

11a) If the customer don’t selects the payment option, the customer can’t click on “Check out” to submit the purchase.

***Output:***

The customer views their purchase that consists of product name, product description, amount, total price and payment option in the history page.

***System Requirement Specification:***

SRS-25: The system shall display the picture of customer on top-right of the user interface.

SRS-29: The system shall provide the Checkout interface consists of the product name, picture, amount, total price, and payment option.

SRS-30: The system shall display the Checkout interface that contains product name, total price, and payment option for the customer.

SRS-31: The system shall provide the shopping cart interface.

SRS-35: The system shall provide the user interface for receive amount of product.

SRS-36: The system shall provide the user interface for confirm of buying.

SRS-37: The system shall provide the shopping history page.

SRS-38: The system shall display the name of product in shopping cart.

SRS-39: The system shall display the picture of product in shopping cart.

SRS-40: The system shall display the amount of product in shopping cart.

SRS-41: The system shall display the total price in shopping cart.

**URS-13: The customer can view the shopping history.**

***Description:***

After the customer purchase a product, they can view their shopping history in the history page.

***Precondition:***

The customer must purchase a product at least one items.

***Input:***

|  |  |  |
| --- | --- | --- |
| ***Name*** | ***Type*** | ***Remarks*** |
| History | menu | Click left on tab menu |

***Execution flow:***

1). The customer type the website URL.

2). The system shall provide the index page to the customer.

3). The customer login successful.

4). The system shall redirect to the index page to the customer.

5). The customer clicks on “History” menu on the index page.

6). The system shall provide the history page to the customer.

***Alternative flow:***

6a) If the customer never buy the product, the system will show no history.

***Output:***

The customer views their shopping history that consists of product name, product description, amount, total price and payment option.

***System Requirement Specification:***

SRS-37: The system shall provide the shopping history page.

SRS-47: The system shall retrieve shopping history from database when customer login and continue shopping.

**URS-14: The customer can logout from the system.**

***Introduction:***

The customer can logout from the website when they finish the purchase.

***Precondition:***

The customers have to login already.

***Input:***

|  |  |  |
| --- | --- | --- |
| ***Name*** | ***Type*** | ***Remarks*** |
| Logout | button | Click left on button |

***Execution flow:***

1). The customer type the website URL.

2). The system shall provide the index page to the customer.

3). The customer login successful.

4). The system shall redirect to the index page to the customer.

5). The customer clicks on “Logout” menu on the index page.

6). The system shall provide the alert message in order to confirm the customer.

7). The customer clicks on “Submit” button on alert message to confirm.

8). The system shall delete the customers’ session from site.

***Alternative flow:***

5a) If the customer don’t login, the customer can’t logout to system.

7a) If the customer click on “Cancel” button on alert message, the system will redirect to the index page to the customer.

***Output:***

The pictures’ customers do not appear on the top-right in the index page and show the login menu in order to login again.

***System Requirement Specification:***

SRS-50: The system shall provide the logout interface.

**URS-15: The administrator can login to the system.**

***Description:***

The administrator can login in order to manage the website.

***Precondition:***

The administrator must have their account in the database.

***Input:***

|  |  |  |
| --- | --- | --- |
| ***Name*** | ***Type*** | ***Remarks*** |
| Username | String | Username must be at least 4 characters. |
| Password | String | Password length should be 4-20 character. |

***Execution flow:***

1). The administrator type the website URL.

2). The system shall provide the index page to the administrator.

3). The administrator clicks on “Login” menu on the index page.

4). The system shall provide the login page to the administrator.

5). The administrator type username and password in the login page.

6). The administrator clicks on “Login” button to submit.

7). The system validates the user name and password.

8). The system shall redirect to the admin index page.

***Alternative flow:***

7a) If the username or password are not matched, the system shows the error message “The username is incorrect” and “The password is incorrect”.

***Output:***

The administrator views their picture at the top-right on the index page to confirm that administrator has login already.

***System Requirement Specification:***

SRS-06: The system shall provide the login user interface, which receives the username and password.

SRS-07: The system shall provide the successful login page.

SRS-08: The system shall validate the username and password.

SRS-11: The system can check the username format. The username format should be at least 4 characters.

SRS-12: The system can check the password format. The password format must be 4-20 characters. The password must contain small letter, capital letter, and number.

SRS-21: The system shall display the error message. “The username is incorrect”

SRS-22: The system shall display the error message. “The password is incorrect”

**URS-16: The administrator can provide details of the products.**

***Description:***

The administrator can add the product and provide products’ description into the database.

***Precondition:***

The administrators have to login already.

***Input:***

|  |  |  |
| --- | --- | --- |
| ***Name*** | ***Type*** | ***Remarks*** |
| Name | String | Name must be only a character. |
| Description | String | Description must be only a character |
| Price | double | Price must be Thai Bath only. |
| Picture | BLOB | Picture format must be only .jpg, .jpeg, and .png. |

***Execution flow:***

1). The administrator type the website URL.

2). The system shall provide the index page to the administrator.

3). The customer login successful.

4). The system shall redirect to the admin index page.

5). The administrator clicks on “Add product” button to provide the product.

6). The system shall provide the add product page to the administrator.

7). The administrator types the products’ information that consists of name, description, price and picture.

8). The administrator clicks on “Submit” to confirm the product.

9). The system shall redirect to the product page to the administrator.

***Alternative flow:***

7a) If the administration input data incomplete, the system will show message “Input the data”.

***Output:***

The new product appears on the product page after administrator added a product.

***System Requirement Specification:***

SRS-04: The system shall provide the user interface, which receives the name of product, description, and picture of product.

SRS-05: The system shall add the product information to the database using name, description, and picture of product.

**URS-17: The administrator can see the shopping history of all customers.**

***Description:***

The administrator can view the all customers’ history in the history page.

***Precondition:***

In the history database must contain at least one customers’ history.

***Input:***

|  |  |  |
| --- | --- | --- |
| ***Name*** | ***Type*** | ***Remarks*** |
| History | menu | Click left on button |

***Execution flow:***

1). The administrator type the website URL.

2). The system shall provide the index page to the administrator.

3). The customer login successful.

4). The system shall redirect to the admin index page.

5). The administrator clicks on “History” menu in the admin index page.

***Output:***

The all of customers’ history appear on the history page.

***System Requirement Specification:***

SRS-25: The system shall display the picture of customer on top-right of the user interface.

SRS-42: The system shall provide the shopping history of all customers page.

SRS-43: The system shall display the date that customer buy product.

**URS-18: The administrator can logout from the system.**

***Description:***

The administrator can logout from the website when they finish managing website.

***Precondition:***

The administrators have to login already.

***Input:***

|  |  |  |
| --- | --- | --- |
| ***Name*** | ***Type*** | ***Remarks*** |
| logout | button | Click left on button |

***Execution flow:***

1). The administrator type the website URL.

2). The system shall provide the index page to the administrator.

3). The customer login successful.

4). The system shall redirect to the admin index page.

5). The administrator clicks on “Logout” menu in the index page.

6). The system shall provide the alert message to the administrator.

7). The administrator clicks on “Submit” button on the alert message to confirm.

8). The system shall delete the administrators’ session from site.

***Alternative flow:***

6a) If the customer don’t login, the customer can’t logout to system.

8a) If the customer click on “Cancel” button on alert message, the system will redirect to the index page to the customer.

***Output:***

The pictures’ administrators do not appear on the top-right in the index page and show the login menu in order to login again.

***System Requirement Specification:***

SRS-50: The system shall provide the logout interface.

# *2. SYSTEM REQUIREMENT SPECIFICATION*

## 2.1 List SRS

|  |  |  |  |
| --- | --- | --- | --- |
| **SRS** | | **Function Name** | |
| SRS-01 | | The system shall provide the registration interface, which receives the name, surname, address, username, password, and picture for register to the system. | |
| SRS-02 | | The system shall add customer information to the database using name, surname, address, username, password, and picture. | |
| SRS-03 | | The system shall provide the successful registration page. | |
| SRS-04 | | The system shall provide the user interface, which receives the name of product, description, and picture of product. | |
| SRS-05 | | The system shall add the product information to the database using name, description, and picture of product. | |
| SRS-06 | | The system shall provide the login user interface, which receives the username and password. | |
| SRS-07 | | The system shall provide the successful login page. | |
| SRS-08 | | The system shall validate the username and password. | |
| SRS-09 | | The system can check the name format. The name format should be only character. | |
| SRS-10 | | The system can check the surname format. The surname format should be only character. | |
| SRS-11 | | The system can check the username format. The username format should be at least 4 characters. | |
| SRS-12 | | The system can check the password format. The password format must be 4-20 characters. The password must contain small letter, capital letter, and number. | |
| SRS-13 | | The system can check the picture format. The picture format must be only .jpg, .jpeg, and .png. | |
| SRS-14 | | The system shall display the error message. “The name must be only characters” | |
| SRS-15 | | The system shall display the error message. “The surname must be only characters” | |
| SRS-16 | | The system shall display the error message. “The name length must less than 30 characters” | |
| SRS-17 | | The system shall display the error message. “The surname length must less than 30 characters” | |
| SRS-18 | | The system shall display the error message. “The username must be at least 4 characters” | |
| SRS-19 | | The system shall display the error message. “The password length should be 4-20 character” | |
| SRS-20 | | The system shall display the error message. “ The password must contain the small letter, capital letter, and number. | |
| SRS-21 | | The system shall display the error message. “The username is incorrect” | |
| SRS-22 | | The system shall display the error message. “The password is incorrect” | |
| SRS-23 | | The system shall display the error message. “ The picture must be the picture format” | |
| SRS-24 | | The system shall provide the Edit customer information interface for edit their own information after customer has been logged in to the system. | |
| SRS-25 | | The system shall display the picture of customer on top-right of the user interface. | |
| SRS-26 | | The system shall display the product catalogs. | |
| SRS-27 | | The system shall provide the search box for search the product name. | |
| SRS-28 | | The system shall display the result of search the product name. | |
| **SRS** | | **Function Name** | |
| SRS-29 | | The system shall provide the Checkout interface consists of the product name, picture, amount, total price, and payment option. | |
| SRS-30 | | The system shall display the Checkout interface that contains product name, total price, and payment option for the customer. | |
| SRS-31 | | The system shall provide the shopping cart interface. | |
| SRS-32 | | The system shall remove the selected product from the shopping cart database. | |
| SRS-33 | | The system shall provide the payment user interface, which receive the payment option (money transfer, credit cart and PayPal) that the customer wants to use. | |
| SRS-34 | | The system shall display the payment option interface that the customer wants to use. | |
| SRS-35 | | The system shall provide the user interface for receive amount of product. | |
| SRS-36 | | The system shall provide the user interface for confirm of buying. | |
| SRS-37 | | The system shall provide the shopping history page. | |
| SRS-38 | | The system shall display the name of product in shopping cart. | |
| SRS-39 | | The system shall display the picture of product in shopping cart. | |
| SRS-40 | | The system shall display the amount of product in shopping cart. | |
| SRS-41 | | The system shall display the total price in shopping cart. | |
| SRS-42 | | The system shall provide the shopping history of all customers interface. | |
| SRS-43 | | The system shall display the date that customer buy product. | |
| SRS-44 | | The system shall provide the user interface for save the shopping cart. | |
| SRS-45 | | The system shall save shopping cart history. | |
| SRS-46 | | The system shall display the successful shopping cart history interface. | |
| SRS-47 | | The system shall retrieve shopping history from database when customer login and continue shopping. | |
| SRS-48 | | The system shall add history of customer to database using username, name of product, and picture of product, amount of product, total price, payment option, and date. | |
| SRS-49 | | The system shall retrieve the product information from the product name. | |
| SRS-50 | | The system shall provide the logout interface. | |
| SRS-51 | | The system shall update information of customer to database. | |
| SRS-52 | | The system shall provide the user interface for remove the shopping cart. | |
|  | |  | |
|  | |  | |
|  | |  | |
|  | |  | |
|  | |  | |
|  | |  | |
|  | |  | |

# *3. Use Case Diagram*



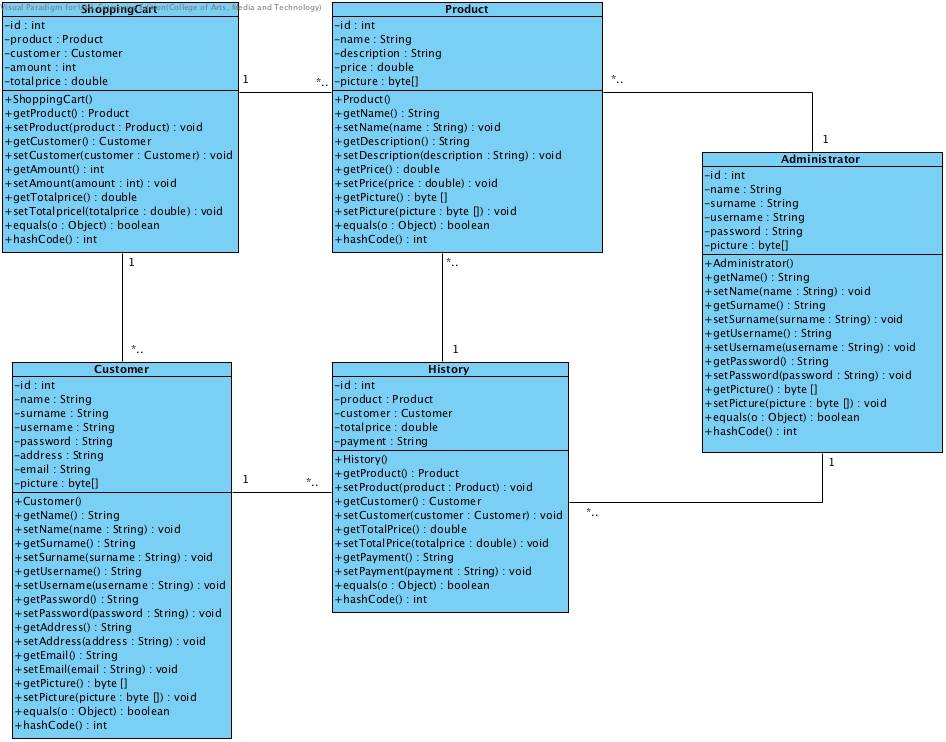
## 3.1 List Use Case

|  |  |
| --- | --- |
| **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. |

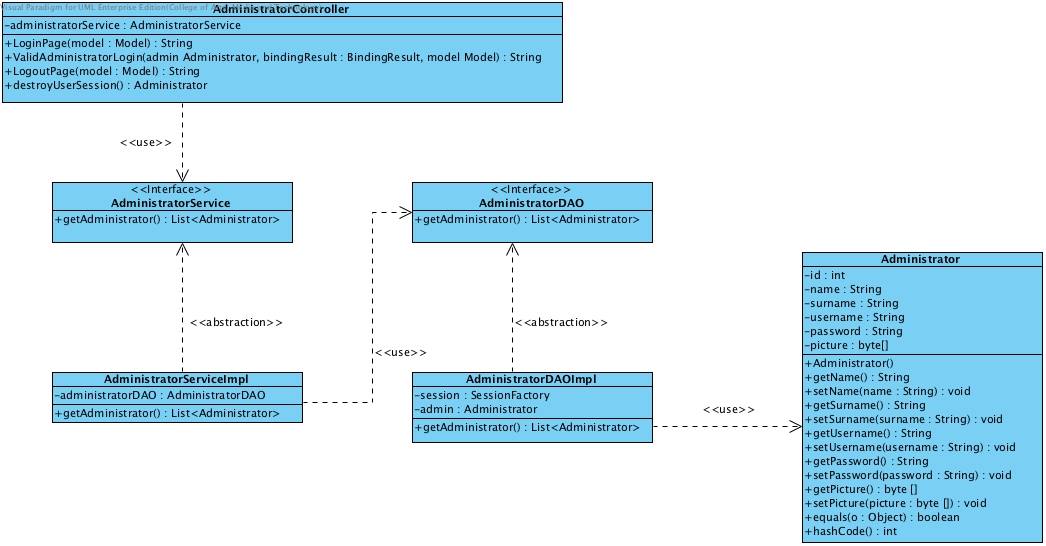
# *4. Class Diagram*

|  |  |
| --- | --- |
| ID | Description |
| CD-01 | Administrator |
| CD-02 | Login |
| CD-03 | Product |
| CD-04 | History |
| CD-05 | ShoppingCart |
| CD-06 | AdministratorController |
| CD-07 | AdministratorDAO |
| CD-08 | AdministratorDAOImpl |
| CD-09 | AdministratorService |
| CD-10 | AdministratorServiceImpl |
| CD-11 | CustomerController |
| CD-12 | CustomerDAO |
| CD-13 | CustomerDAOImpl |
| CD-14 | CustomerService |
| CD-15 | CustomerServiceImpl |
| CD-16 | ProductController |
| CD-17 | ProductDAO |
| CD-18 | ProductDAOImpl |
| CD-19 | ProductService |
| CD-20 | ProductServiceImpl |
| CD-21 | ShoppingCartController |
| CD-22 | ShoppingCartDAO |
| CD-23 | ShoppingCartDAOImpl |
| CD-24 | ShoppingCartService |
| CD-25 | ShoppingCartServiceImpl |
| CD-26 | HistoryController |
| CD-27 | HistoryDAO |
| CD-28 | HistoryDAOImpl |
| CD-29 | HistoryService |
| CD-30 | HistoryServiceImpl |

**Package: Entity**



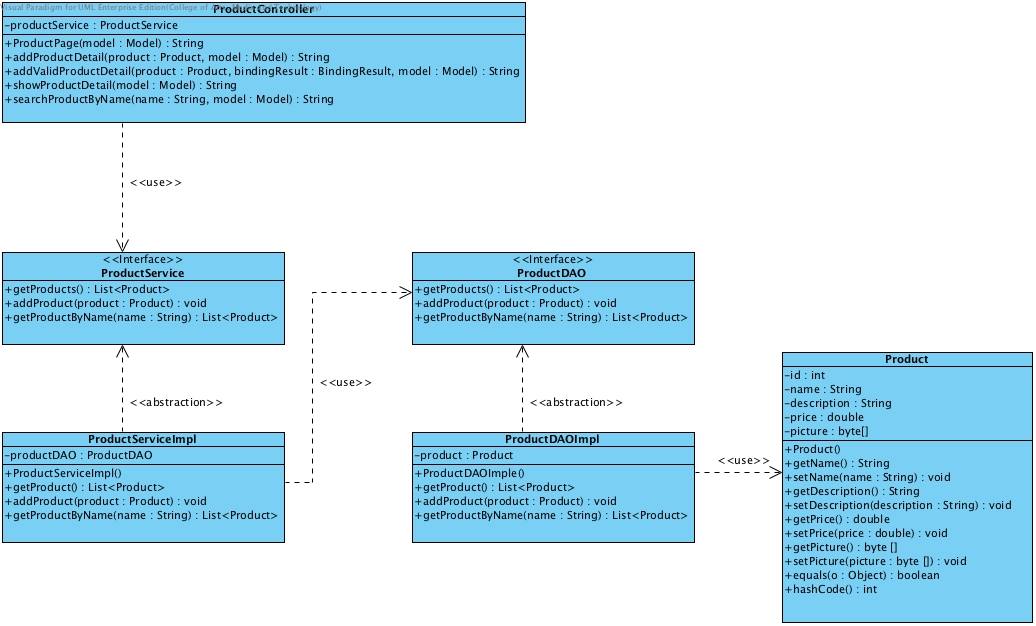
**Package: MVC of Administrator**



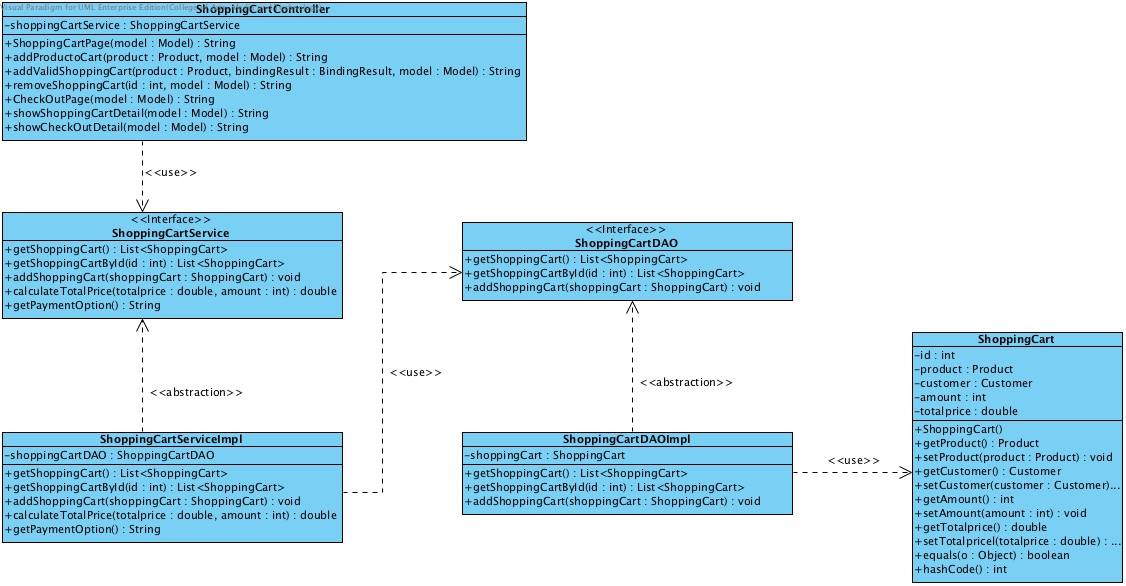
**Package: MVC of Customer**

****

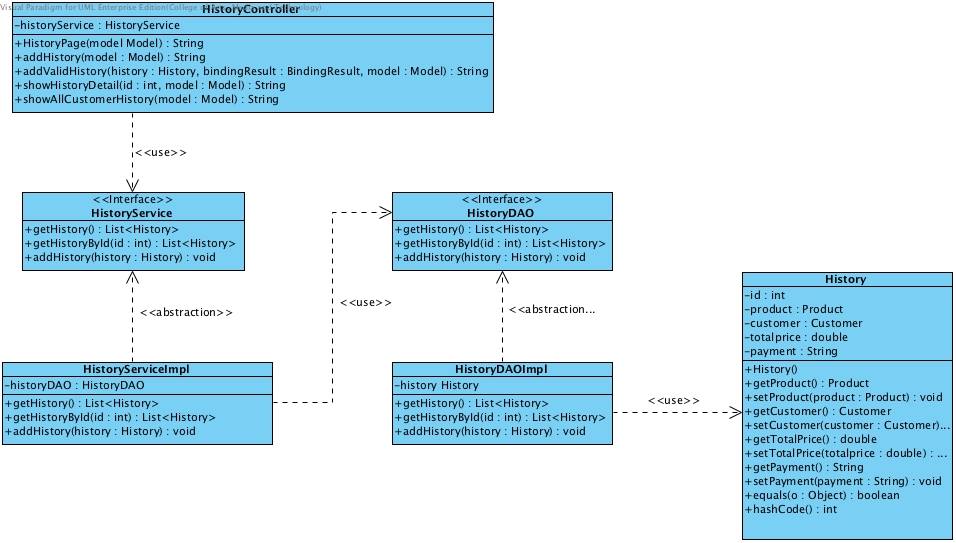
**Package: MVC of Product**

****

**Package: MVC of ShoppingCart**

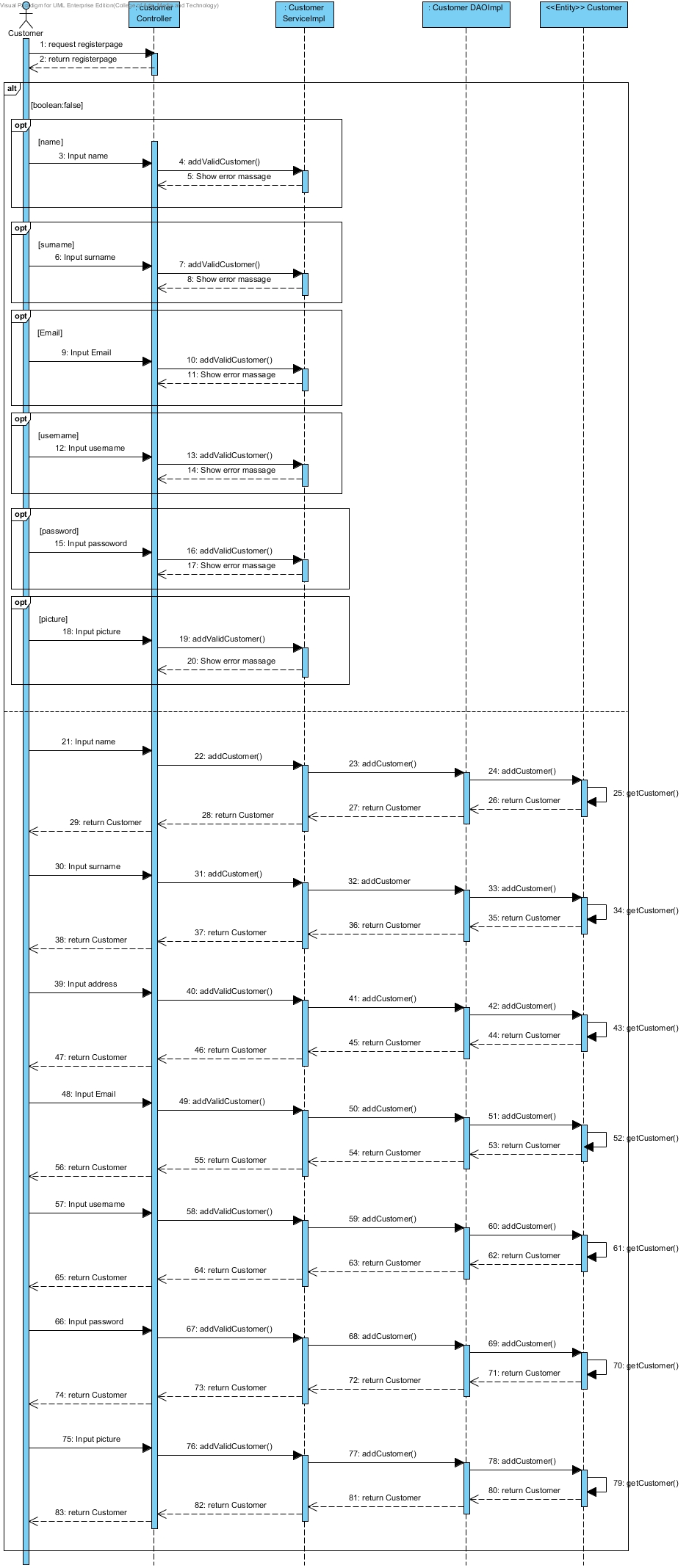
****

**Package: MVC of History**

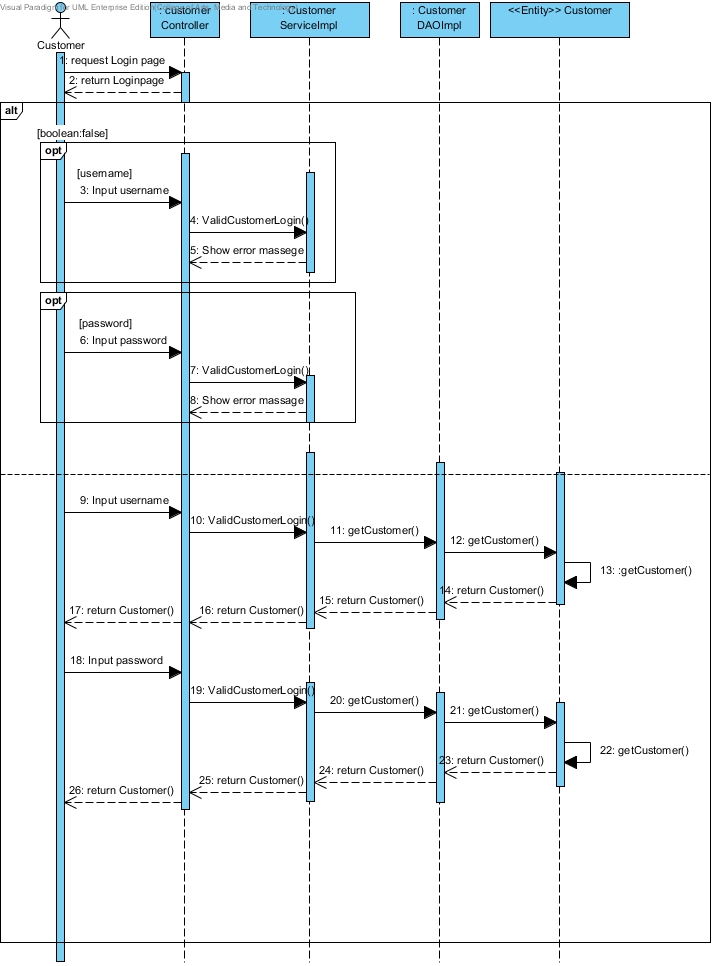
****

# *5. SEQUENCE DIAGRAM*

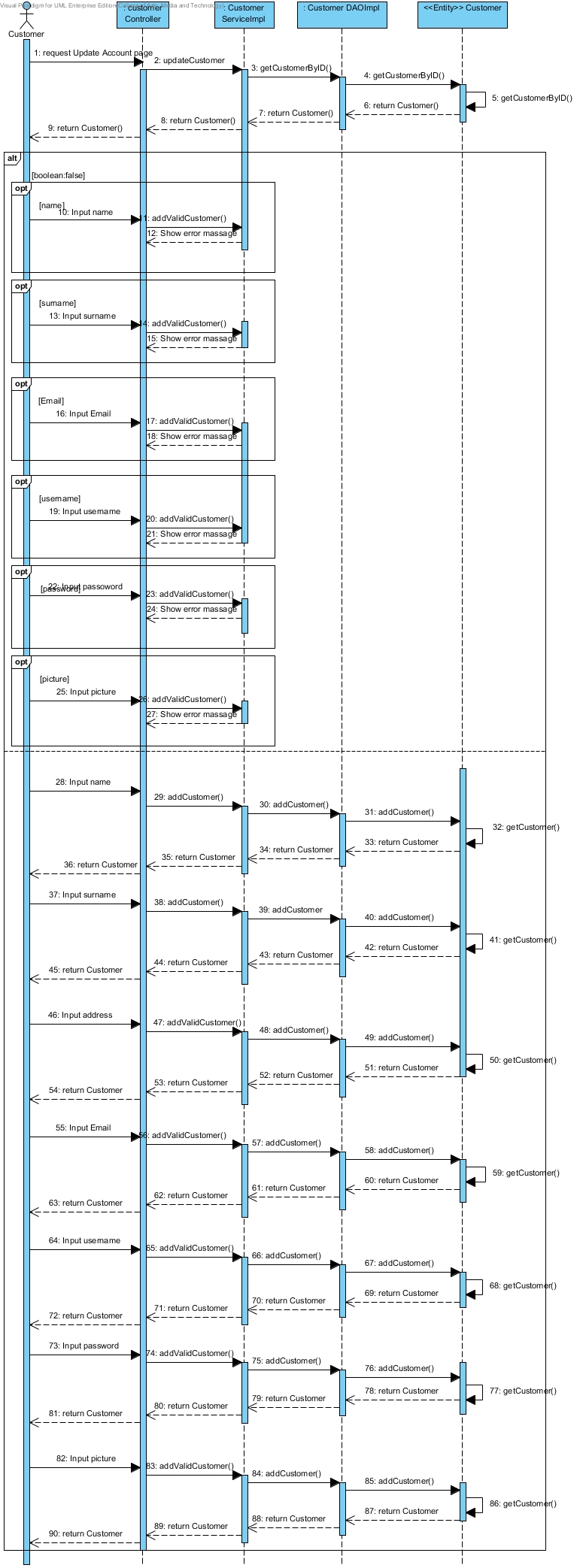
**SD-01: Customer register to system**



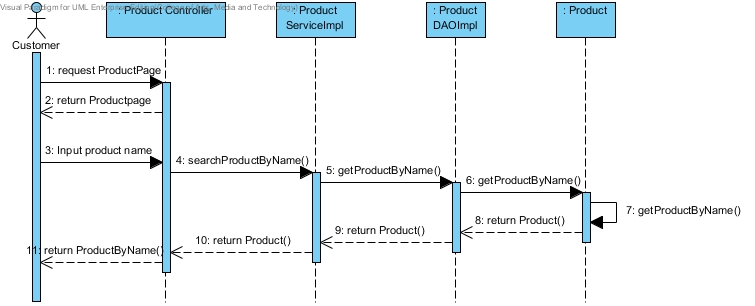
**SD-02: Customer login to system**



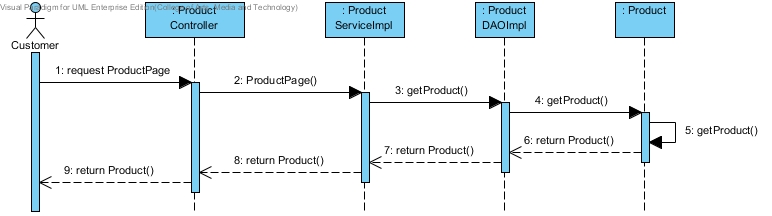
**SD-03: Customer update information**



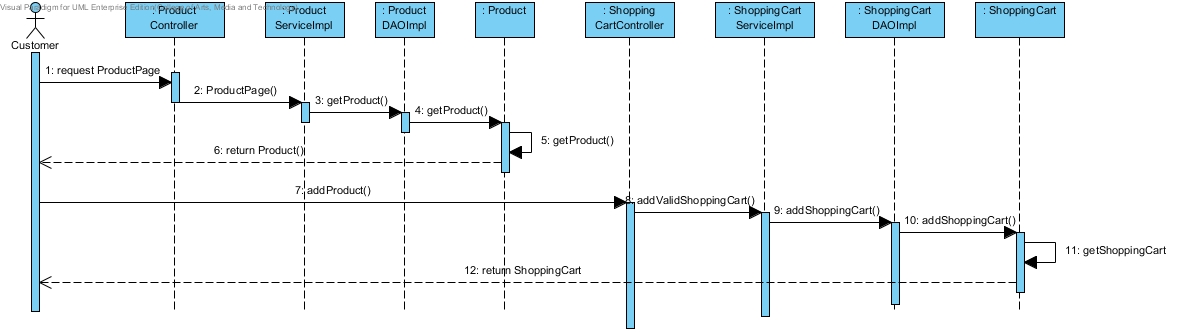
**SD-04: Customer search product by name**



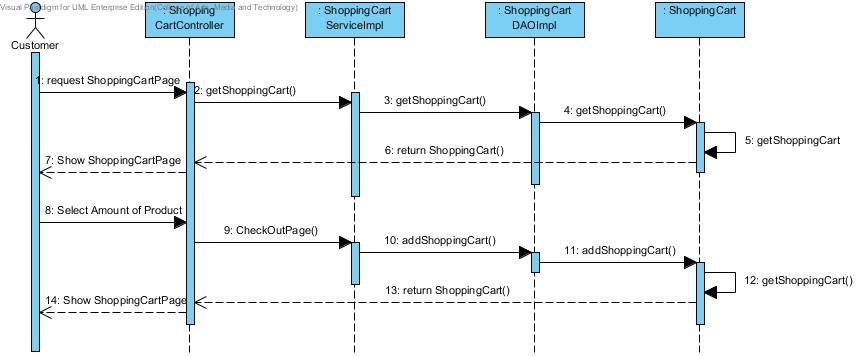
**SD-05: Customer browse product from catalog**



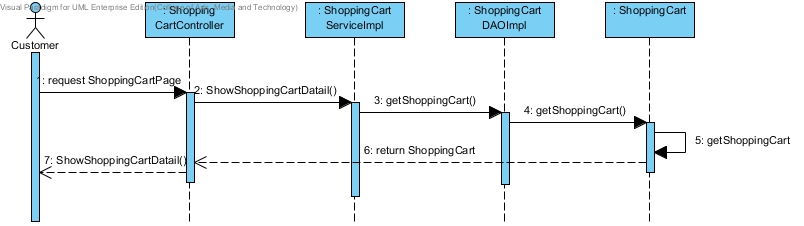
**SD-06: Customer add product to cart**



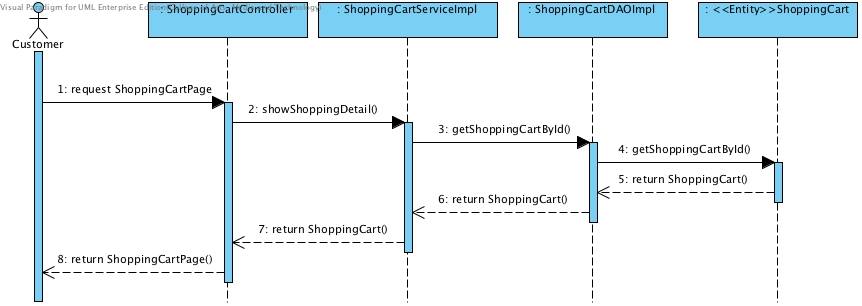
**SD-07: Customer identify amount of product**



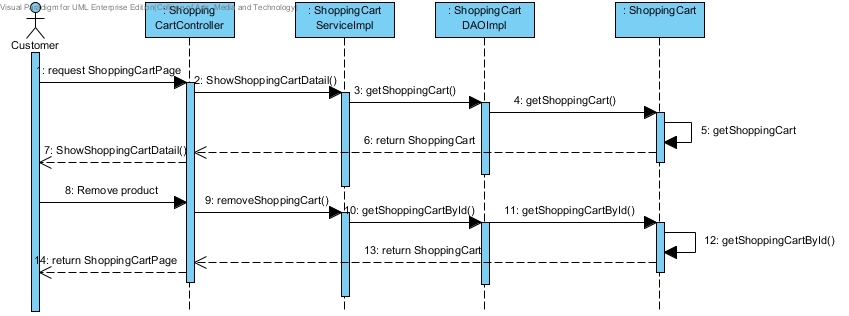
**SD-08: Customer can view shopping cart**



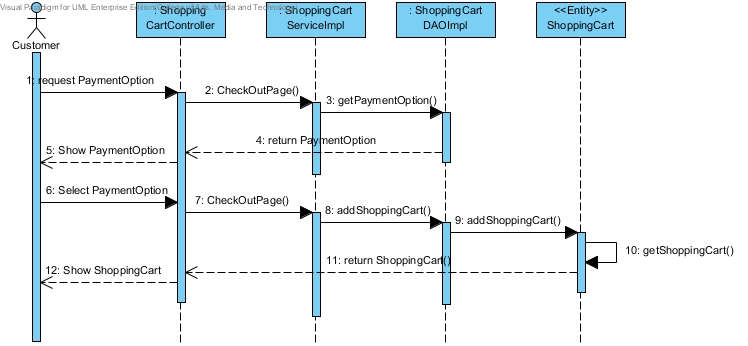
**SD-09: Customer can save shopping cart**



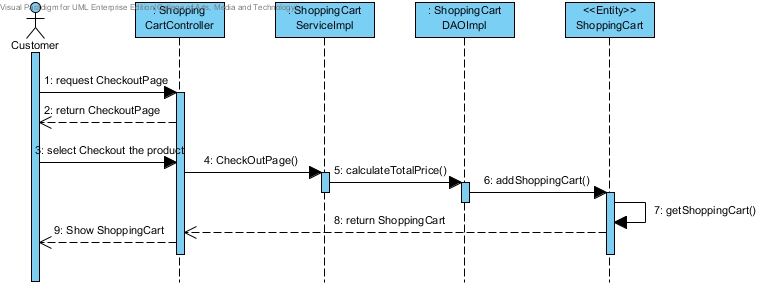
**SD-10: Customer remove product from shopping cart**



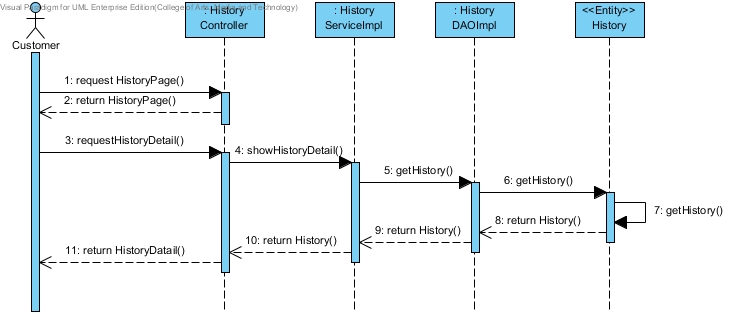
**SD-11: Customer select payment option**



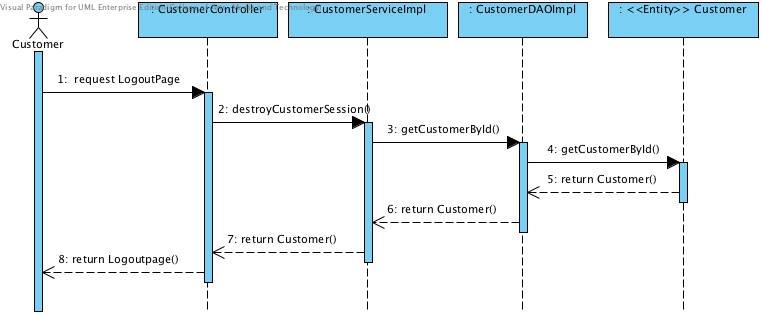
**SD-12: Customer checkout product**



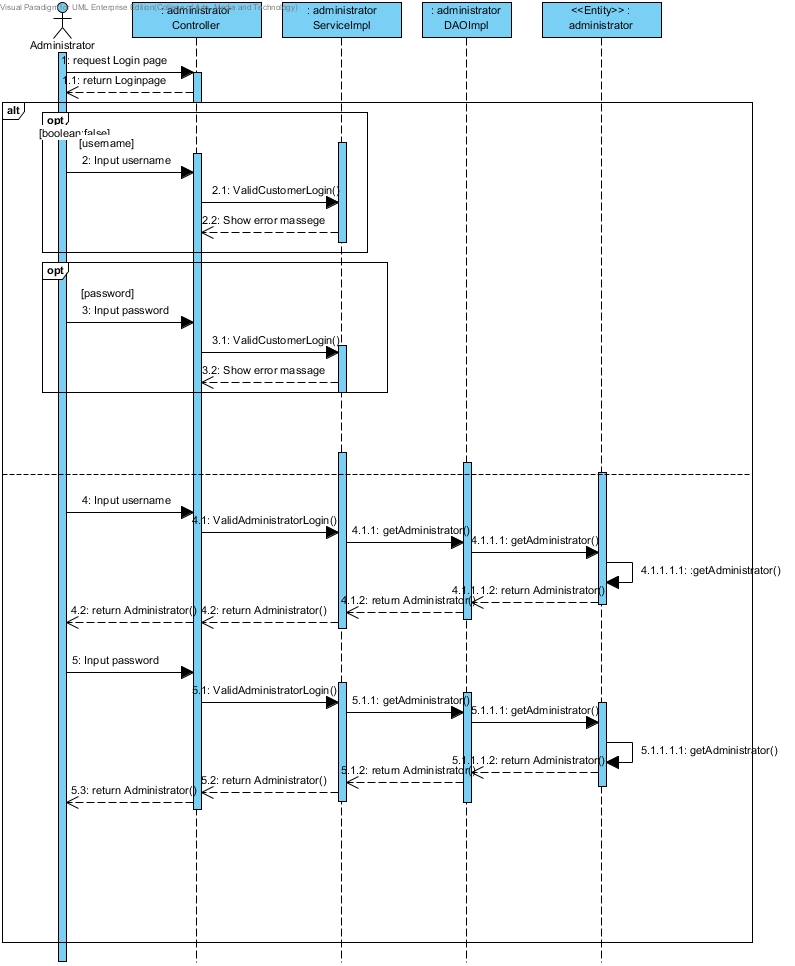
**SD-13: Customer view shopping history**



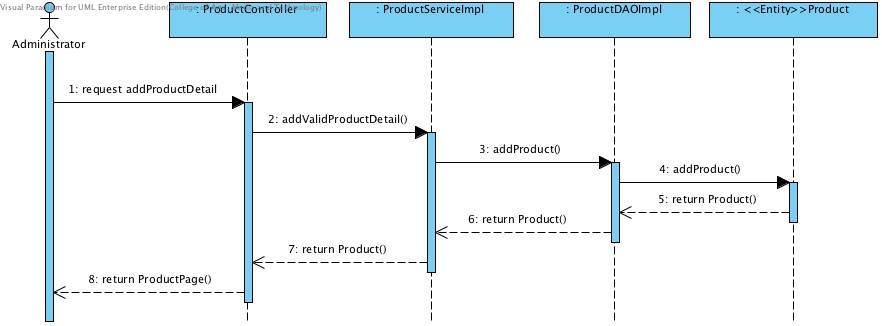
**SD-14: Customer logout from system**



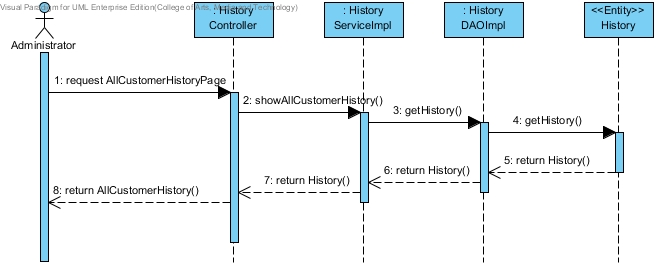
**SD-15: Administrator login to the system**



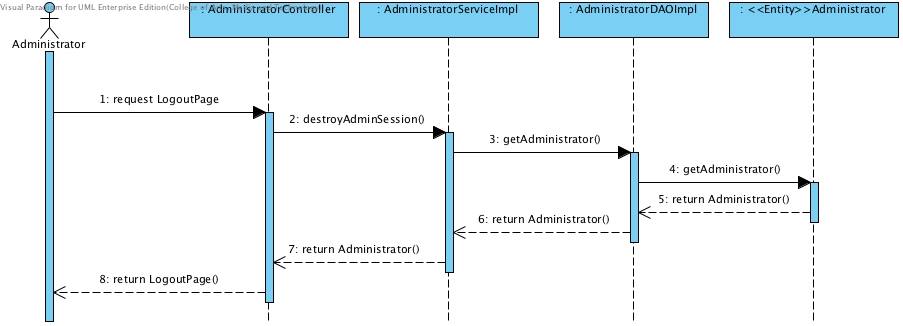
**SD-16: Administrator provide details of product**



**SD-17: Administrator view shopping history of all customers**

****

**SD-18: Administrator logout from system**

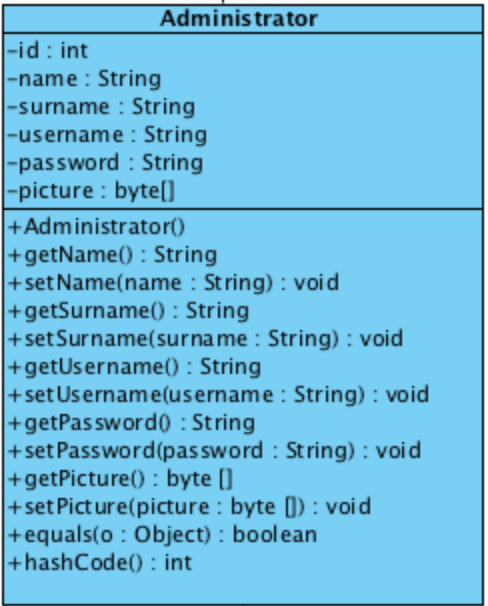


# *6. Method Design*

## 6.1 List Method Design

|  |  |
| --- | --- |
| ID | Description |
| MTD-01 | Administrator |
| MTD-02 | Login |
| MTD-03 | Product |
| MTD-04 | History |
| MTD-05 | ShoppingCart |
| MTD-06 | AdministratorController |
| MTD-07 | AdministratorDAO |
| MTD-08 | AdministratorDAOImpl |
| MTD-09 | AdministratorService |
| MTD-10 | AdministratorServiceImpl |
| MTD-11 | CustomerController |
| MTD-12 | CustomerDAO |
| MTD-13 | CustomerDAOImpl |
| MTD-14 | CustomerService |
| MTD-15 | CustomerServiceImpl |
| MTD-16 | ProductController |
| MTD-17 | ProductDAO |
| MTD-18 | ProductDAOImpl |
| MTD-19 | ProductService |
| MTD-20 | ProductServiceImpl |
| MTD-21 | ShoppingCartController |
| MTD-22 | ShoppingCartDAO |
| MTD-23 | ShoppingCartDAOImpl |
| MTD-24 | ShoppingCartService |
| MTD-25 | ShoppingCartServiceImpl |
| MTD-26 | HistoryController |
| MTD-27 | HistoryDAO |
| MTD-28 | HistoryDAOImpl |
| MTD-29 | HistoryService |
| MTD-30 | HistoryServiceImpl |

**MTD-01: Administrator**



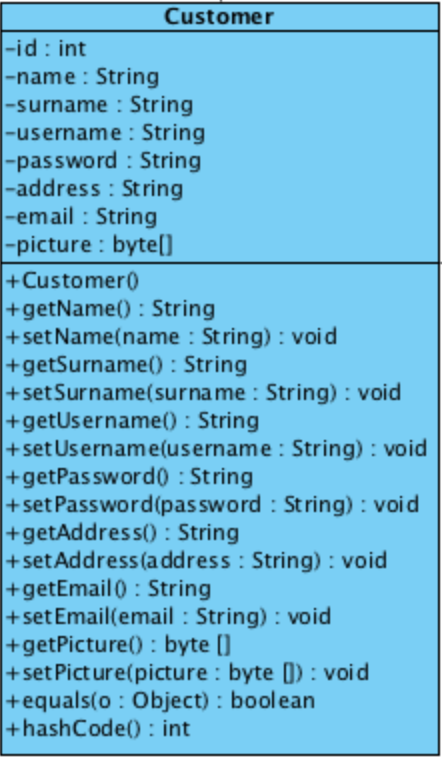
**Attribute**

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Name** | **Description** | **Remarks** |
| 1 | id | The attribute is used to identify id of the administrator. | **Return type:** int |
| 2 | name | The attribute is used to identify name of the administrator. | **Return type:** String |
| 3 | surname | The attribute is used to identify surname of the administrator. | **Return type:** String |
| 4 | username | The attribute is used to identify username of the administrator. | **Return type:** String |
| 5 | password | The attribute is used to identify password of the administrator. | **Return type:** String |
| 6 | picture | The attribute is used to identify picture of the administrator. | **Return type:** byte[] |

**Methods**

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Name** | **Description** | **Remarks** |
| 1 | equals() | The method is used for comparing between two objects. | **Return type:** Boolean  **Parameter:** o |
| 2 | hashCode() | The method is used for checking instance attributes value from override methods. | **Return type:** int |

**MTD-02: Customer**



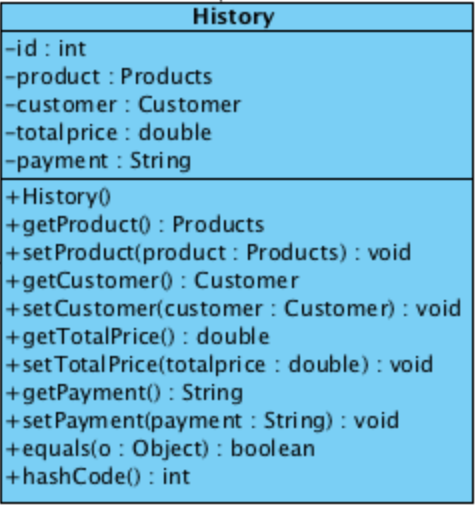
Attributes

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Name** | **Description** | **Remarks** |
| 1 | id | The attribute is used to identify id of the customer. | **Return type:** int |
| 2 | name | The attribute is used to identify name of the customer. | **Return type:** String |
| 3 | surname | The attribute is used to identify surname of the customer. | **Return type:** String |
| 4 | username | The attribute is used to identify username of the customer. | **Return type:** String |
| 5 | password | The attribute is used to identify password of the customer. | **Return type:** String |
| 6 | address | The attribute is used to identify address of the customer. | **Return type:** String |
| 7 | email | The attribute is used to identify email of the customer. | **Return type:** String |
| 8 | picture | The attribute is used to identify a picture input of the customer. | **Return type:** byte[] |

**Methods**

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Name** | **Description** | **Remarks** |
| 1 | equals() | The method is used for comparing between two objects. | **Return type:** Boolean  **Parameter:** o |
| 2 | hashCode() | The method is used for checking instance attributes value from override methods. | **Return type:** int |

**MTD-03: History**



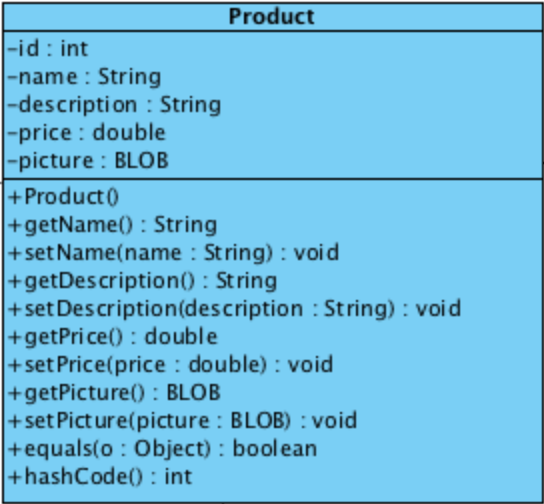
**Attributes**

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Name** | **Description** | **Remarks** |
| 1 | id | The attribute is used to identify id of the history. | **Return type:** int |
| 2 | product | The product object that identify a product in the history. | **Return type:** Products |
| 3 | customer | The customer object that identify a product in the history. | **Return type:** Customer |
| 4 | totalprice | The attribute is used to identify total price in the history. | **Type value:** double |
| 5 | payment | The attribute is used to identify payment option in the history. | **Type value:** String |

**Methods**

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Name** | **Description** | **Remarks** |
| 1 | equals() | The method is used for comparing between two objects. | **Return type:** Boolean  Parameter: o |
| 2 | hashCode() | The method is used for checking instance attributes value from override methods. | **Return type:** int |

**MTD-04: Product**



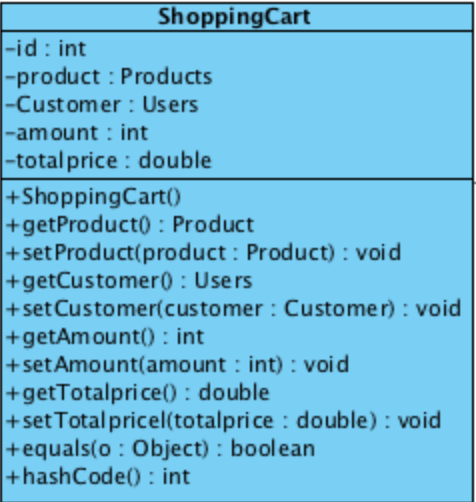
**Attributes**

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Name** | **Description** | **Remarks** |
| 1 | id | The attribute is used to identify id of the product. | **Return type:** int |
| 2 | name | The attribute is used to identify name of the product. | **Return type:** String |
| 3 | description | The attribute is used to identify description of the product. | **Return type:** String |
| 4 | price | The attribute is used to identify price of the product. | **Return type:** String |
| 5 | picture | The attribute is used to identify a picture input of the product. | **Return type:** byte[] |

**Methods**

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Name** | **Description** | **Remarks** |
| 1 | equals() | The method is used for comparing between two objects. | **Return type:** Boolean  **Parameter:** o |
| 2 | hashCode() | The method is used for checking instance attributes value from override methods. | **Return type:** int |

**MTD-05: ShoppingCart**



**Attributes**

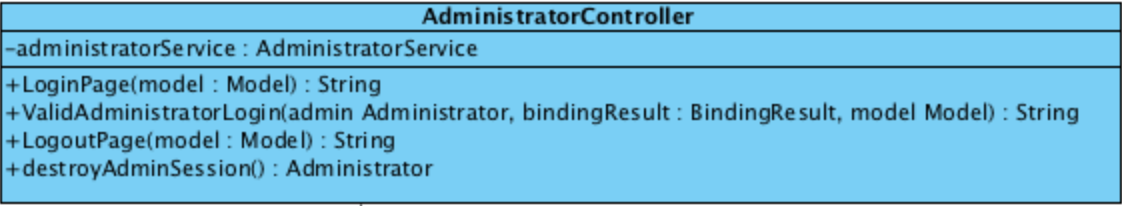
|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Name** | **Description** | **Remarks** |
| 1 | id | The attribute is used to identify id of the administrator | **Return type:** int |
| 2 | product | The product object that identify a product in shopping cart. | **Return type:** String |
| 3 | customer | The customer object that identify a product in shopping cart. | **Return type:** Customer |
| 4 | amount | The attribute is used to identify amount in the shopping cart. | **Return type:** int |
| 5 | totalprice | The attribute is used to identify total price in the shopping cart. | **Return type:** double |

**Methods**

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Name** | **Description** | **Remarks** |
| 1 | equals() | The method is used for compare between two objects. | **Return type:** Boolean  **Parameter:** o |
| 2 | hashCode() | The method is used for check instance attributes value from override methods. | **Return type:** int |

**Method design of Administrators’ MVC**

**MTD-06: AdministratorController**



**Attributes**

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Name** | **Description** | **Remarks** |
| 1 | administratorService | The object that related to AdministratorService class. | **Return type:** AdministratorService |

**Methods**

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Name** | **Description** | **Remarks** |
| 1 | LoginPage() | The method is used for providing the login page. | **Return type:** String  **Parameter:** model |
| 2 | ValidAdministratorLogin() | The method is used for checking valid the username and password that match with the database. | **Return type:** String  **Parameter:** admin, bindingResult, model |
| 3 | LogoutPage() | The method is used for providing the logout page. | **Return type:** String  **Parameter:** model |
| 4 | destroyAdminSession() | The method is used for destroying session when admin logout from website. | **Return type:** Admininstrator |

**MTD-07: AdministratorDAO**



**Methods**

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Name** | **Description** | **Remarks** |
| 1 | getAdministrator() | The method is used for getting administrator list from database. | **Return type:** List<Admininstrator> |

**MTD-08: AdministratorDAOImpl**



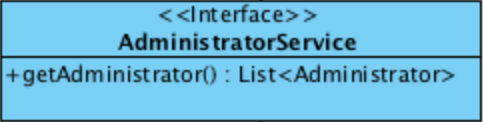
**Attributes**

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Name** | **Description** | **Remarks** |
| 1 | session | The object is used for collect a session after admin and customer login successful. | **Return type:** SessionFactory |
| 2 | admin | The object that related to Administrator class. | **Return type:** Administrator |

**Methods**

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Name** | **Description** | **Remarks** |
| 1 | getAdministrator() | The method is used for getting administrator list from database. | **Return type:** List<Admininstrator> |

**MTD-09: AdministratorService**



**Methods**

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Name** | **Description** | **Remarks** |
| 1 | getAdministrator() | The method is used for getting administrator list from database. | **Return type:** List<Admininstrator> |

**MTD-10: AdministratorServiceImpl**



**Attributes**

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Name** | **Description** | **Remarks** |
| 1 | administratorDAO | The object that related to AdministratorDAO class. | **Return type:** AdministratorDAO |

**Methods**

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Name** | **Description** | **Remarks** |
| 1 | getAdministrator() | The method is used for getting administrator list from database. | **Return type:** List<Admininstrator> |

**Method design of Customers’ MVC**

**MTD-11: CustomerController**

****

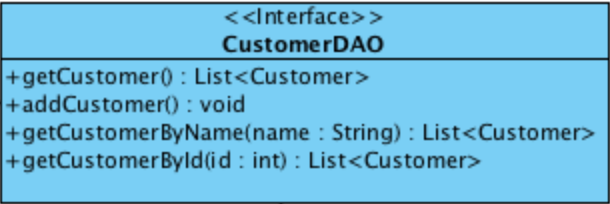
**Attributes**

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Name** | **Description** | **Remarks** |
| 1 | customerService | The object that related to CustomerService class. | **Return type:** CustomerService |

**Methods**

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Name** | **Description** | **Remarks** |
| 1 | RegisterPage() | The method is used for providing the register page to the customer. | **Return type:** String  **Parameter:** model |
| 2 | addCustomer() | The method is used for adding customer into the database when customer finishes registration. | **Return type:** String  **Parameter:** customer, model |
| 3 | addValidCustomer() | The method is used for checking valid customers’ information before adding into database. | **Return type:** String  **Parameter:** customer, model, bindingResult |
| 4 | LoginPage() | The method is used for providing the login page to the customer. | **Return type:** String  **Parameter:** model |
| 5 | ValidCustomerLogin() | The method is used for checking valid the username and password that match with the database. | **Return type:** String  **Parameter:** customer, model, bindingResult |
| 6 | updateCustomer() | The method is used for updating the customers’ information into the database | **Return type:** String  **Parameter:** id, model |
| 7 | LogoutPage() | The method is used for providing the logout page to the customer. | **Return type:** String  **Parameter:** model |
| 8 | destroyCustomerSession() | The method is used for destroying session when customer logout from website. | **Return type:** Customer |

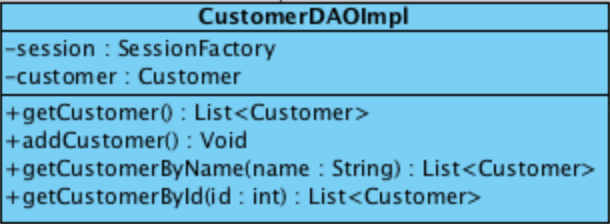
**MTD-12: CustomerDAO**



**Methods**

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Name** | **Description** | **Remarks** |
| 1 | getCustomer() | The method is used for getting a customer list from database. | **Return type:** List<Customer> |
| 2 | addCustomer() | The method is used for adding customer into the database when customer finishes registration. | **Return type:** void |
| 3 | getCustomerByName() | The method is used for using name character to find the customer from the list. | **Return type:** List<Customer>  **Parameter:** name |
| 4 | getCustomerById() | The method is used for using id character to find the customer from the list. | **Return type:** List<Customer>  **Parameter:** id |

**MTD-13: CustomerDAOImpl**



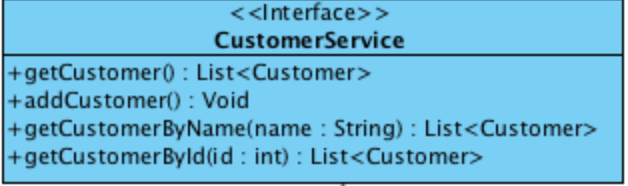
**Attributes**

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Name** | **Description** | **Remarks** |
| 1 | session | The object is used for collect a session after admin and customer login successful. | **Return type:** SessionFactory |
| 2 | customer | The object that related to Customer class. | **Return type:** Customer |

**Methods**

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Name** | **Description** | **Remarks** |
| 1 | getCustomer() | The method is used for getting a customer list from database. | **Return type:** List<Customer> |
| 2 | addCustomer() | The method is used for adding customer into the database when customer finishes registration. | **Return type:** void |
| 3 | getCustomerByName() | The method is used for using name character to find the customer from the list. | **Return type:** List<Customer>  **Parameter:** name |
| 4 | getCustomerById() | The method is used for using id character to find the customer from the list. | **Return type:** List<Customer>  **Parameter:** id |

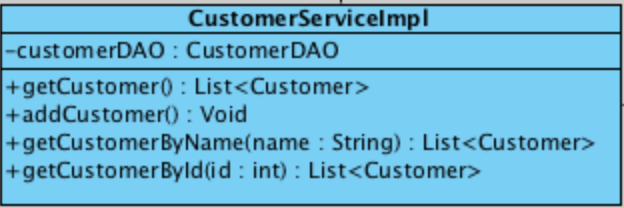
**MTD-14: CustomerService**



**Methods**

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Name** | **Description** | **Remarks** |
| 1 | getCustomer() | The method is used for getting a customer list from database. | **Return type:** List<Customer> |
| 2 | addCustomer() | The method is used for adding customer into the database when customer finishes registration. | **Return type:** void |
| 3 | getCustomerByName() | The method is used for using name character to find the customer from the list. | **Return type:** List<Customer>  **Parameter:** name |
| 4 | getCustomerById() | The method is used for using id character to find the customer from the list. | **Return type:** List<Customer>  **Parameter:** id |

**MTD-15: CustomerServiceImpl**



**Attributes**

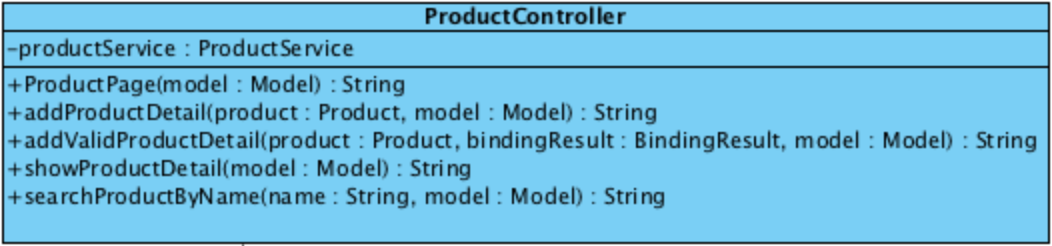
|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Name** | **Description** | **Remarks** |
| 1 | customerDAO | The object that related to CustomerDAO class. | **Return type:** CustomerDAO |

**Methods**

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Name** | **Description** | **Remarks** |
| 1 | getCustomer() | The method is used for getting a customer list from database. | **Return type:** List<Customer> |
| 2 | addCustomer() | The method is used for adding customer into the database when customer finishes the registration. | **Return type:** void |
| 3 | getCustomerByName() | The method is used for using name character to get the customer from the list. | **Return type:** List<Customer>  **Parameter:** name |
| 4 | getCustomerById() | The method is used for using id character to get the customer from the list. | **Return type:** List<Customer>  **Parameter:** id |

**Method design of Products’ MVC**

**MTD-16: ProductController**



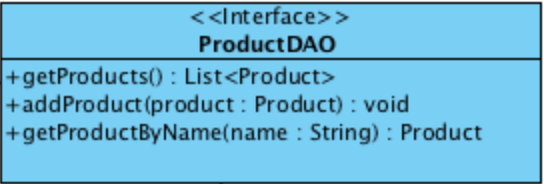
**Attributes**

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Name** | **Description** | **Remarks** |
| 1 | productService | The object that related to ProductService class. | **Return type:** ProductService |

**Methods**

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Name** | **Description** | **Remarks** |
| 1 | ProductPage() | The method is used for providing the product page to the website. | **Return type:** String  **Parameter:** model |
| 2 | addProductDetail() | The method is used for adding a products’ description into the database. | **Return type:** String  **Parameter:** product, model |
| 3 | addValidProductDetail() | The method is used for checking valid of products’ description. | **Return type:** String  **Parameter:** product, bindingResult, model |
| 4 | showProductDetail() | The method is used for providing the products’ description. | **Return type:** String  **Parameter:** model |
| 5 | searchProductByName() | The method is used for getting the products’ description from the name character. | **Return type:** String  **Parameter:** name, model |

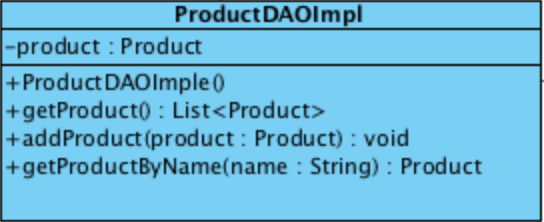
**MTD-17: ProductDAO**



**Methods**

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Name** | **Description** | **Remarks** |
| 1 | getProducts() | The method is used for getting a product list from the database. | **Return type:** List<Product> |
| 2 | addProduct () | The method is used for adding a products’ description into the database. | **Return type:** void  **Parameter:** product |
| 3 | getProductByName() | The method is used for getting the products’ description from the database by name character. | **Return type:** Product  **Parameter:** name |

**MTD-18: ProductDAOImpl**



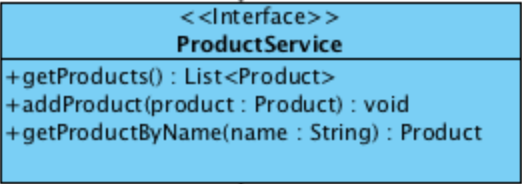
**Attributes**

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Name** | **Description** | **Remarks** |
| 1 | product | The object that related to Product class. | **Return type:** Product |

**Methods**

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Name** | **Description** | **Remarks** |
| 1 | getProducts() | The method is used for getting a product list from the database. | **Return type:** List<Product> |
| 2 | addProduct () | The method is used for adding a products’ description into the database. | **Return type:** void  **Parameter:** product |
| 3 | getProductByName() | The method is used for getting the products’ description from the database by name character. | **Return type:** Product  **Parameter:** name |

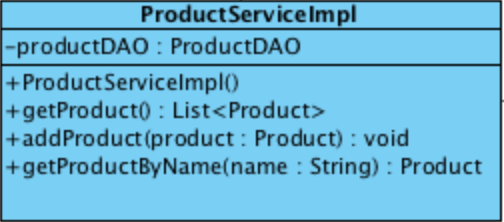
**MTD-19: ProductService**



**Methods**

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Name** | **Description** | **Remarks** |
| 1 | getProducts() | The method is used for getting a product list from the database. | **Return type:** List<Product> |
| 2 | addProduct () | The method is used for adding a products’ description into the database. | **Return type:** void  **Parameter:** product |
| 3 | getProductByName() | The method is used for getting the products’ description from the database by name character. | **Return type:** Product  **Parameter:** name |

**MTD-20: ProductServiceImpl**



**Attributes**

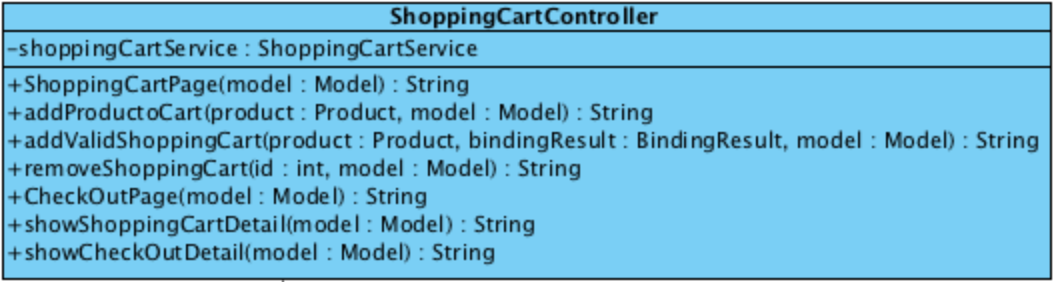
|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Name** | **Description** | **Remarks** |
| 1 | productDAO | The object that related to ProductDAO class. | **Return type:** ProductDAO |

**Methods**

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Name** | **Description** | **Remarks** |
| 1 | getProducts() | The method is used for getting a product list from the database. | **Return type:** List<Product> |
| 2 | addProduct () | The method is used for adding a products’ description into the database. | **Return type:** void  **Parameter:** product |
| 3 | getProductByName() | The method is used for getting the products’ description from the database by name character. | **Return type:** Product  **Parameter:** name |

**Method design of Shopping Carts’ MVC**

**MTD-21: ShoppingCartController**

****

**Attributes**

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Name** | **Description** | **Remarks** |
| 1 | shoppingCartService | The object that related to ShoppingCartService service class. | **Return type:** ShoppingCartService |

**Methods**

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Name** | **Description** | **Remarks** |
| 1 | ShoppingCartPage() | The method is used for providing the product page to the website. | **Return type:** String  **Parameter:** model |
| 2 | addProductCart() | The method is used for adding a products’ description into the database. | **Return type:** String  **Parameter:** product, model |
| 3 | addValidShoppingCart() | The method is used for checking valid in order to add a products’ description before add into the database. | **Return type:** String  **Parameter:** product, bindingResult, model |
| 4 | removeShoppingCart() | The method is used for checking valid a products’ description. | **Return type:** int  **Parameter:** product, model |
| 5 | CheckOutPage() | The method is used for providing the products’ description. | **Return type:** String  **Parameter:** model |
| 6 | showShoppingCart() | The method is used for getting the shopping car list of each customer. | **Return type:** String  **Parameter:** model |
| 7 | showCheckOutDetail() | The method is used for providing the shopping carts’ description and payment option in the check out page. | **Return type:** String  **Parameter:** model |

**MTD-22: ShoppingCartDAO**

****

**Methods**

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Name** | **Description** | **Remarks** |
| 1 | getShoppingCart() | The method is used for getting a shopping cart list from the database. | **Return type:** List<ShoppingCart> |
| 2 | getShoppingCartById() | The method is used for getting a shopping cart list by shopping carts’ id. | **Return type:** List<ShoppingCart>  **Parameter:** id |
| 3 | addShoppingCart () | The method is used for adding a products’ description into the database. | **Return type:** void  **Parameter:** shoppingCart |

**MTD-23: ShoppingCartDAOImpl**



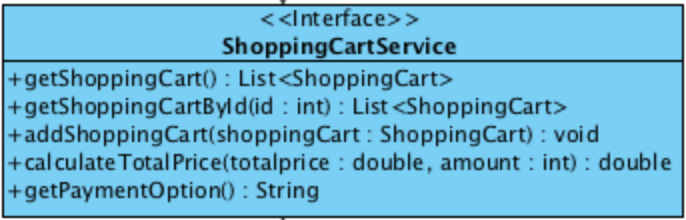
**Attributes**

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Name** | **Description** | **Remarks** |
| 1 | shoppingCart | The object that related to ShoppingCart class. | **Return type:** ShoppingCart |

**Methods**

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Name** | **Description** | **Remarks** |
| 1 | getShoppingCart() | The method is used for getting a shopping cart list from the database. | **Return type:** List<ShoppingCart> |
| 2 | getShoppingCartById() | The method is used for getting a shopping cart list by shopping carts’ id. | **Return type:** List<ShoppingCart>  **Parameter:** id |
| 3 | addShoppingCart () | The method is used for adding a products’ description into the database. | **Return type:** void  **Parameter:** shoppingCart |

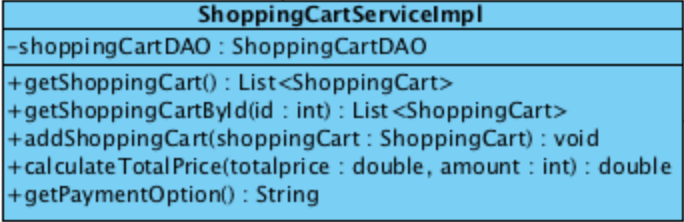
**MTD-24: ShoppingCartService**



**Methods**

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Name** | **Description** | **Remarks** |
| 1 | getShoppingCart() | The method is used for getting a shopping cart list from the database. | **Return type:** List<ShoppingCart> |
| 2 | getShoppingCartById() | The method is used for getting a shopping cart list by shopping carts’ id. | **Return type:** List<ShoppingCart>  **Parameter:** id |
| 3 | addShoppingCart () | The method is used for adding a products’ description into the database. | **Return type:** void  **Parameter:** shoppingCart |
| 4 | calculateTotalPrice() | The method is used for calculating total of product in the shopping cart page. | **Return type:** double  **Parameter:** totalprice, amount |
| 5 | getPaymentOption() | The method is used for providing a payment option in the shopping cart page. | **Return type:** String |

**MTD-25: ShoppingCartServiceImpl**



**Attributes**

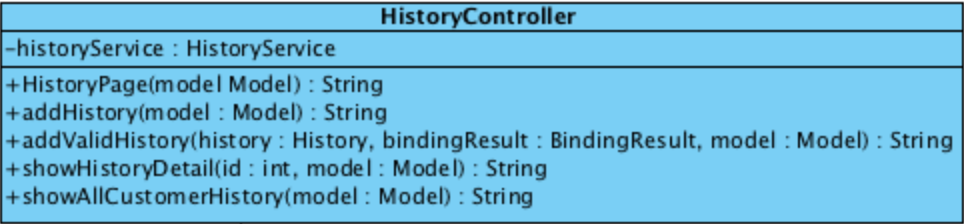
|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Name** | **Description** | **Remarks** |
| 1 | shoppingCartDAO | The object that related to ShoppingCartDAO class. | **Return type:** ShoppingCartDAO |

**Methods**

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Name** | **Description** | **Remarks** |
| 1 | getShoppingCart() | The method is used for getting a shopping cart list from the database. | **Return type:** List<ShoppingCart> |
| 2 | getShoppingCartById() | The method is used for getting a shopping cart list by shopping carts’ id. | **Return type:** List<ShoppingCart>  **Parameter:** id |
| 3 | addShoppingCart () | The method is used for adding a products’ description into the database. | **Return type:** void  **Parameter:** shoppingCart |
| 4 | calculateTotalPrice() | The method is used for calculating total of product in the shopping cart page. | **Return type:** double  **Parameter:** totalprice, amount |
| 5 | getPaymentOption() | The method is used for providing a payment option in the shopping cart page. | **Return type:** String |

**Method design of Histories’ MVC**

**MTD-26: HistoryController**



**Attributes**

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Name** | **Description** | **Remarks** |
| 1 | historyService | The object that related to HistoryService class. | **Return type:** HistoryService |

**Methods**

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Name** | **Description** | **Remarks** |
| 1 | HistoryPage() | The method is used for providing the history page to the website. | **Return type:** String  **Parameter:** model |
| 2 | addHistory () | The method is used for adding a histories’ description into the database. | **Return type:** String  **Parameter:** model |
| 3 | addValidHistory() | The method is used for checking valid of histories’ description. | **Return type:** String  **Parameter:** history, bindingResult, model |
| 4 | showHistoryDetail() | The method is used for providing a histories’ description in the history page. | **Return type:** id, String  **Parameter:** model |
| 5 | showAllCustomerHistory() | The method is used for providing all customers’ histories in the history page for administrator. | **Return type:** String  **Parameter:** model |

**MTD-27: HistoryDAO**



**Methods**

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Name** | **Description** | **Remarks** |
| 1 | getHistory () | The method is used for getting a history list from the database. | **Return type:** List<History> |
| 2 | getHistoryById() | The method is used for getting a history list by id of history. | **Return type:** List<History>  **Parameter:** id |
| 3 | addHistory () | The method is used for adding a histories’ description into the database. | **Return type:** void  **Parameter:** history |

**MTD-28: HistoryDAOImpl**



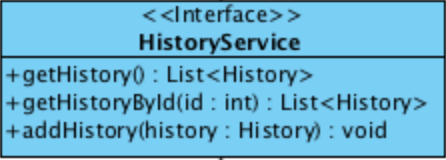
**Attributes**

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Name** | **Description** | **Remarks** |
| 1 | history | The object that related to History class. | **Return type:** History |

**Methods**

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Name** | **Description** | **Remarks** |
| 1 | getHistory () | The method is used for getting a history list from the database. | **Return type:** List<History> |
| 2 | getHistoryById() | The method is used for getting a history list by id of history. | **Return type:** List<History>  **Parameter:** id |
| 3 | addHistory () | The method is used for adding a histories’ description into the database. | **Return type:** void  **Parameter:** history |

**MTD-29: HistoryService**



**Method**

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Name** | **Description** | **Remarks** |
| 1 | getHistory () | The method is used for getting a history list from the database. | **Return type:** List<History> |
| 2 | getHistoryById() | The method is used for getting a history list by id of history. | **Return type:** List<History>  **Parameter:** id |
| 2 | addHistory () | The method is used for adding a description of history into the database. | **Return type:** void  **Parameter:** history |

**MTD-30: HistoryServiceImpl**



**Method**

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Name** | **Description** | **Remarks** |
| 1 | getHistory () | The method is used for getting a history list from the database. | **Return type:** List<History> |
| 2 | getHistoryById() | The method is used for getting a history list by id of history. | **Return type:** List<History>  **Parameter:** id |
| 3 | addHistory () | The method is used for adding a description of history into the database. | **Return type:** void  **Parameter:** history |

# *7. TRACEABILITY MATRIX*

Table of Contents

*1. USER REQUIREMENT* 1

1.1 List 1

1.2 User Requirement Specification 2

*2. SYSTEM REQUIREMENT SPECIFICATION* 32

2.1 List SRS 32

*3. Use Case Diagram* 35

3.1 List Use Case 36

*4. Class Diagram* 37

*5. SEQUENCE DIAGRAM* 43

*6. Method Design* 53

6.1 List Method Design 53

*7. TRACEABILITY MATRIX* 82