

**ELLENA**

**ONLINE WOMEN FASHION SHOP**

**Software Design Document**

– Ho Chi Minh, May 2022 –

**Table of Contents**

[I. Overview](#_heading=h.gjdgxs) **5**

[1. Code Packages/Namespaces](#_heading=h.30j0zll) 5

[2. Database Schema](#_heading=h.1fob9te) 6

[II. Code Designs](#_heading=h.3znysh7) **7**

[1. Account Management](#_heading=h.2et92p0) 7

[a. Class Diagram](#_heading=h.tyjcwt) 7

[b. Class Specifications](#_heading=h.3dy6vkm) 8

[UserDAO](#_heading=h.cnyhy7x1kuhn) 8

[UserDTO](#_heading=h.2ate2qu2p65h) 9

[UserError](#_heading=h.czcjvhr013qq) 9

[DBUtils](#_heading=h.cfao0ftjj2ah) 9

[AddAccountController](#_heading=h.b5vupuaenvbm) 9

[ShowAccountController](#_heading=h.v12k4xwbbuxd) 9

[RegisterController](#_heading=h.wzpy3496fzo7) 9

[ActivateAccountController](#_heading=h.gl6kc7yua7ut) 10

[DeactivateAccountController](#_heading=h.x1afc6ioq44a) 10

[SearchAccountController](#_heading=h.12j0x1fgkq8j) 10

[UpdateAccountController](#_heading=h.dga682onbevh) 10

[c. Sequence Diagram(s)](#_heading=h.1t3h5sf) 11

[d. Database queries](#_heading=h.4d34og8) 13

[2. Product Management](#_heading=h.2s8eyo1) 15

[a. Class Diagram](#_heading=h.7xmj9hnejglb) 15

[b. Class Specifications](#_heading=h.x3asgkp3s3p0) 15

[ProductDAO](#_heading=h.h8yab3ye2xjz) 15

[ProductDTO](#_heading=h.frrq3i828tmb) 16

[DBUtils](#_heading=h.pa5y79dvi5k4) 16

[AddProductController](#_heading=h.u30jl48ss8nx) 16

[ManagerUpdateProductController](#_heading=h.lv3qedqq124e) 17

[ProductRouteController](#_heading=h.1d44m6s4b2zz) 17

[ManagerShowProductController](#_heading=h.hegp10vnlehv) 17

[ManagerShowProductDetailController](#_heading=h.lafuem42awpi) 17

[ViewHomeController](#_heading=h.2sak79bohq9o) 17

[ActivateProductController](#_heading=h.or7hsme82km4) 18

[DeactivateProductController](#_heading=h.vg0mkcmolsl1) 18

[FilterSearchedProductController](#_heading=h.ywikortbx67c) 18

[FilterCategoryProductController](#_heading=h.tabhlkp5ntuj) 18

[c. Sequence Diagram(s)](#_heading=h.ajesgtvrkblg) 19

[Figure 2.1: Create a product](#_heading=h.4ljxt5scj9up) 19

[Figure 2.2: Search for products - Manager](#_heading=h.1ou2gx5642ej) 20

[d. Database queries](#_heading=h.m8mxxnpghzi5) 21

[3. Order Management](#_heading=h.ldgn5aeq9ky) 23

[a. Class Diagram](#_heading=h.r21fwgfp4d1h) 23

[b. Class Specifications](#_heading=h.hqjvpye84dst) 23

[OrderDAO](#_heading=h.4p5vcqpeofr) 23

[OrderDTO](#_heading=h.na6u1omc90ri) 24

[OrderDetailDTO](#_heading=h.eteg5wxn0lou) 24

[OrderStatusDTO](#_heading=h.t6tlaelynj3i) 24

[OrderError](#_heading=h.oq6r3f85g7co) 24

[DBUtils](#_heading=h.2d82fmob55vi) 25

[SearchOrderController](#_heading=h.1xhu0gp02o4g) 25

[UpdateOrderController](#_heading=h.p648uih1snv0) 25

[CancelOrderController](#_heading=h.egqnqlwshqqn) 25

[ViewOrderHistoryController](#_heading=h.w27iw9ew063p) 25

[CheckoutController](#_heading=h.n34x0gs0ihr6) 25

[MomoRequestController](#_heading=h.q4gyc9ltyod6) 26

[CustomerViewOrderDetailController](#_heading=h.mmuos0d9e40t) 26

[ShowOrderController](#_heading=h.fsfk3r1wx0fj) 26

[ajaxServlet](#_heading=h.iwv18evn1l8w) 26

[JavaMailUtils](#_heading=h.6jeog6fd12zt) 26

[c. Sequence Diagram(s)](#_heading=h.rquvd1w2u1gz) 27

[d. Database queries](#_heading=h.ei5akzpefrta) 28

[III. Database Tables](#_heading=h.17dp8vu) **30**

[1. tblRoles](#_heading=h.3rdcrjn) 30

[2. tblUsers](#_heading=h.26in1rg) 30

[3. tblOrderStatus](#_heading=h.iewa46njr94v) 30

[4. tblOrder](#_heading=h.j2zhf4jj128a) 30

[5. tblCategory](#_heading=h.tnxddehx3ryc) 31

[6. tblProduct](#_heading=h.nw7jr5ng8hpt) 31

[7. tblProductColors](#_heading=h.y3y4igknz7cw) 31

[8. tblColorImage](#_heading=h.9474cupe9jky) 31

[9. tblColorSizes](#_heading=h.51gus9oswio6) 31

[10. tblRating](#_heading=h.phkptkg450nb) 32

[11. tblOrderDetail](#_heading=h.k413b7dt8geo) 32

[12. tblOrderStatusUpdate](#_heading=h.e0zjk2oq13dj) 32

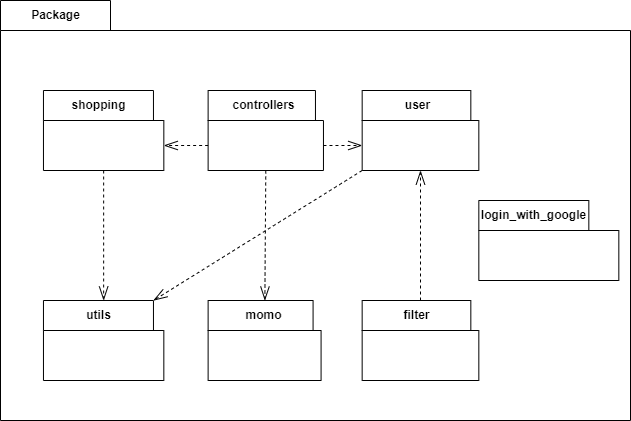
[13. tblCart](#_heading=h.x6dq5l1ojkn3) 32

[14. tblCartItems](#_heading=h.x852c3t18g60) 33

[15. tblReturns](#_heading=h.awix47qy06qw) 33

# I. Overview

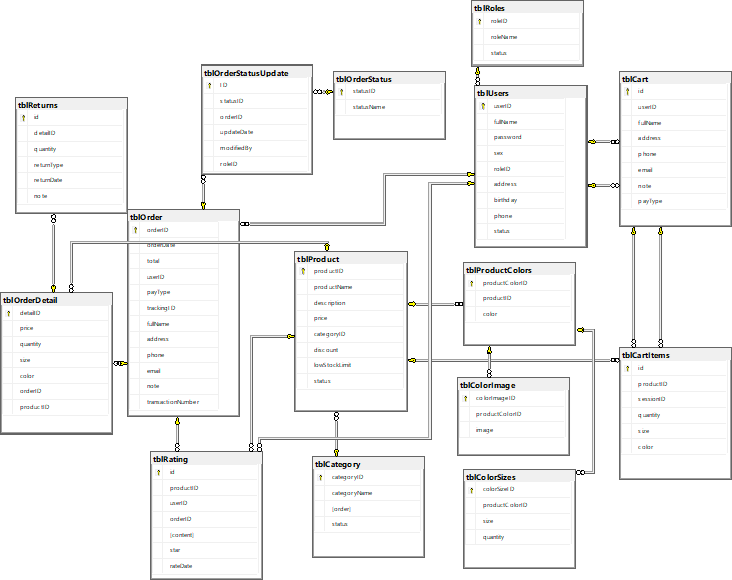
## 1. Code Packages/Namespaces



***Package descriptions & package class naming conventions***

| **No** | **Package** | **Description** |
| --- | --- | --- |
| 01 | store | This package contains all the other packages in this Web app. |
| 02 | controllers | This package contains classes for all the controllers in the app  Class naming convention: ActionController |
| 03 | shopping | This package contains classes for product items and carts.  Class naming convention: ItemDAO, ItemDTO,... |
| 04 | user | This package contains classes for processing users’ data.  Class naming convention: UserDAO, UserDTO,... |
| 05 | login\_with\_google | This package contains classes for processing Google Sign-in accounts’ information. |
| 06 | momo | This package contains classes for processing Momo online payment. |
| 07 | utils | This package contains classes for common functions. |

## 2. Database Schema



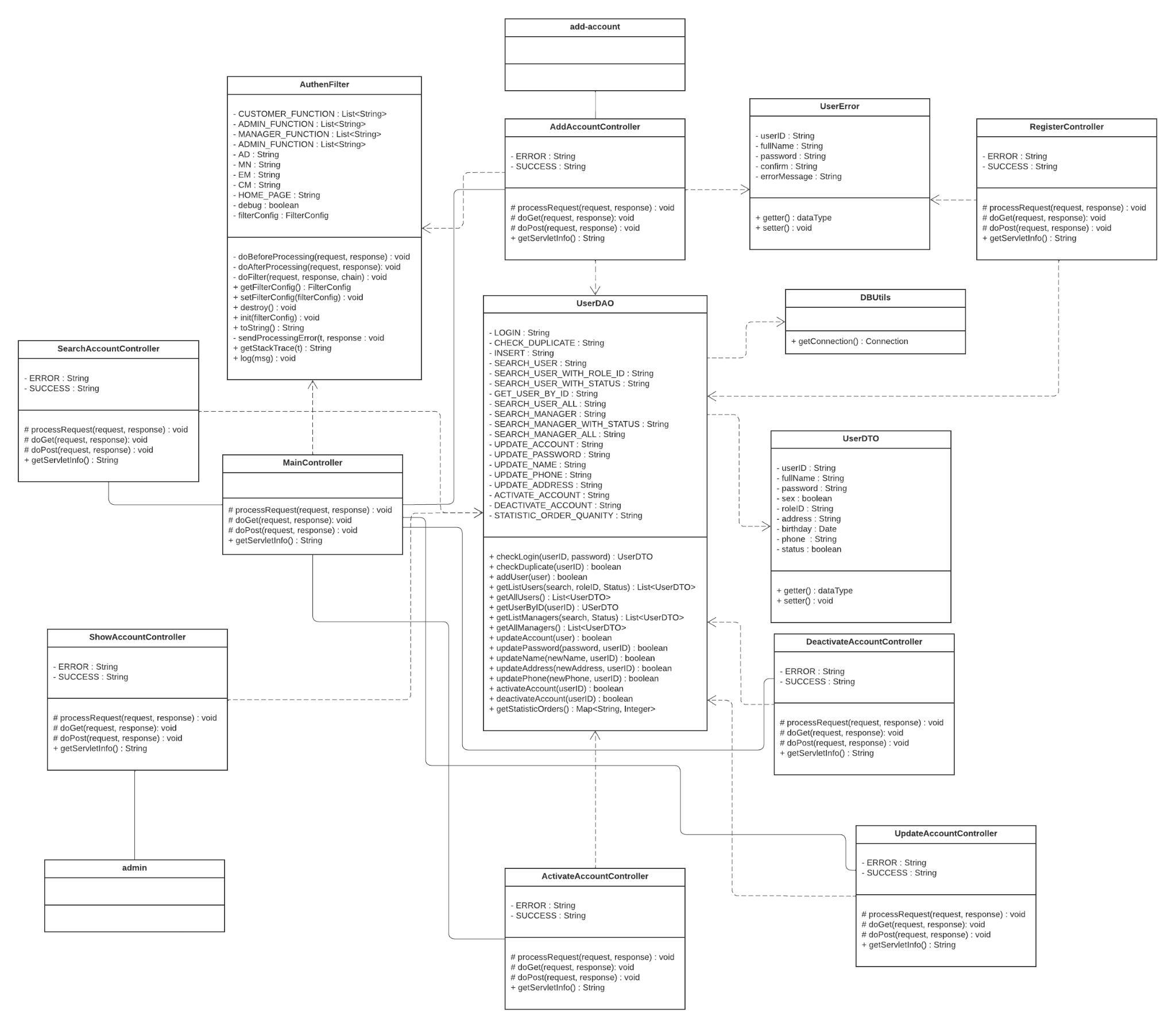
***Table descriptions & package class naming conventions are as below***

| **No** | **Table** | **Description** |
| --- | --- | --- |
| 1 | tblUsers | Used to store all accounts infomation.  - Primary key: userID  - Foreign key: roleID |
| 2 | tblRoles | Used to limit account permission.  - Primary key: roleID |
| 3 | tblRating | Used to store products rating from customers.  - Primary keys: productID, userID  - Foreign keys: productID, userID |
| 4 | tblOrder | Used to store all orders infomation.  - Primary key: orderID  - Foreign key: statusID |
| 5 | tblOrderStatus | Used to differentiate each order status.  - Primary key: statusID |
| 6 | tblOrderDetail | Used to store the detail in each customer’s order (products, quantity, total price)  - Primary key: detailID  - Foreign keys: orderID, productID |
| 7 | tblProduct | Used to store all the product details  - Primary key: productID  - Foreign key: categoryID |
| 8 | tblCategory | Used to separate different kinds of product.  - Primary key: categoryID |
| 9 | tblProductDetail | Used to store all the product’s variants detail (size,color)  - Primary keys: productID, size, color  - Foreign key: productID |

# II. Code Designs

## 1. Account Management

### a. Class Diagram



### b. Class Specifications

#### UserDAO

| **No** | **Method** | **Description** |
| --- | --- | --- |
| 01 | checkLogin(userID, password) | Check whether an user has an existing account or not |
| 02 | checkDuplicate(userID) | Check whether an userID is duplicated |
| 03 | addUser(user) | Insert an user into the database |
| 04 | getListUsers(search, roleID, Status) | Return a list of users with some criteria |
| 05 | getAllUsers() | Return the list of all users |
| 06 | getUserByID(userID) | Return a user that has the requested userID. |
| 07 | getListManagers(search, Status) | Return a list of searched managers |
| 08 | getAllManagers() | Return the list of all managers |
| 09 | updateAccount(user) | Update an account |
| 10 | updatePassword(password, userID) | Update password of an account |
| 11 | updateName(newName, userID) | Update name of an account |
| 12 | updateAddress(newAddress, userID) | Update address of an account |
| 13 | updatePhone(newPhone, userID) | Update phone of an account |
| 14 | updateSex(newSex, userID) | Update sex of an account |
| 16 | updateBirthday(newBirthday, userID) | Update birthday of an account |
| 17 | deactivateAccount(userID) | Change an account’s status to inactive |
| 18 | activateAccount(userID) | Change an account’s status to active |

#### UserDTO

| **No** | **Method** | **Description** |
| --- | --- | --- |
| 01 | getter() | Get an attribute’s value |
| 02 | setter() | Set an attribute’s value |

#### UserError

| **No** | **Method** | **Description** |
| --- | --- | --- |
| 01 | getter() | Get an attribute’s value |
| 02 | setter() | Set an attribute’s value |

#### 

#### DBUtils

| **No** | **Method** | **Description** |
| --- | --- | --- |
| 01 | getConnection() | Return a Connection to connect to the database |

#### 

#### AddAccountController

| **No** | **Method** | **Description** |
| --- | --- | --- |
| 01 | processRequest() | Process a request from the user. |
| 02 | doGet() | Process a GET request from the user. |
| 03 | doPost() | Process a POST request from the user. |
| 04 | getServletInfo() | Get the servlet’s info |

#### ShowAccountController

| **No** | **Method** | **Description** |
| --- | --- | --- |
| 01 | processRequest() | Process a request from the user. |
| 02 | doGet() | Process a GET request from the user. |
| 03 | doPost() | Process a POST request from the user. |
| 04 | getServletInfo() | Get the servlet’s info |

#### RegisterController

| **No** | **Method** | **Description** |
| --- | --- | --- |
| 01 | processRequest() | Process a request from the user. |
| 02 | doGet() | Process a GET request from the user. |
| 03 | doPost() | Process a POST request from the user. |
| 04 | getServletInfo() | Get the servlet’s info |

#### 

#### ActivateAccountController

| **No** | **Method** | **Description** |
| --- | --- | --- |
| 01 | processRequest() | Process a request from the user. |
| 02 | doGet() | Process a GET request from the user. |
| 03 | doPost() | Process a POST request from the user. |
| 04 | getServletInfo() | Get the servlet’s info |

#### 

#### DeactivateAccountController

| **No** | **Method** | **Description** |
| --- | --- | --- |
| 01 | processRequest() | Process a request from the user. |
| 02 | doGet() | Process a GET request from the user. |
| 03 | doPost() | Process a POST request from the user. |
| 04 | getServletInfo() | Get the servlet’s info |

#### SearchAccountController

| **No** | **Method** | **Description** |
| --- | --- | --- |
| 01 | processRequest() | Process a request from the user. |
| 02 | doGet() | Process a GET request from the user. |
| 03 | doPost() | Process a POST request from the user. |
| 04 | getServletInfo() | Get the servlet’s info |

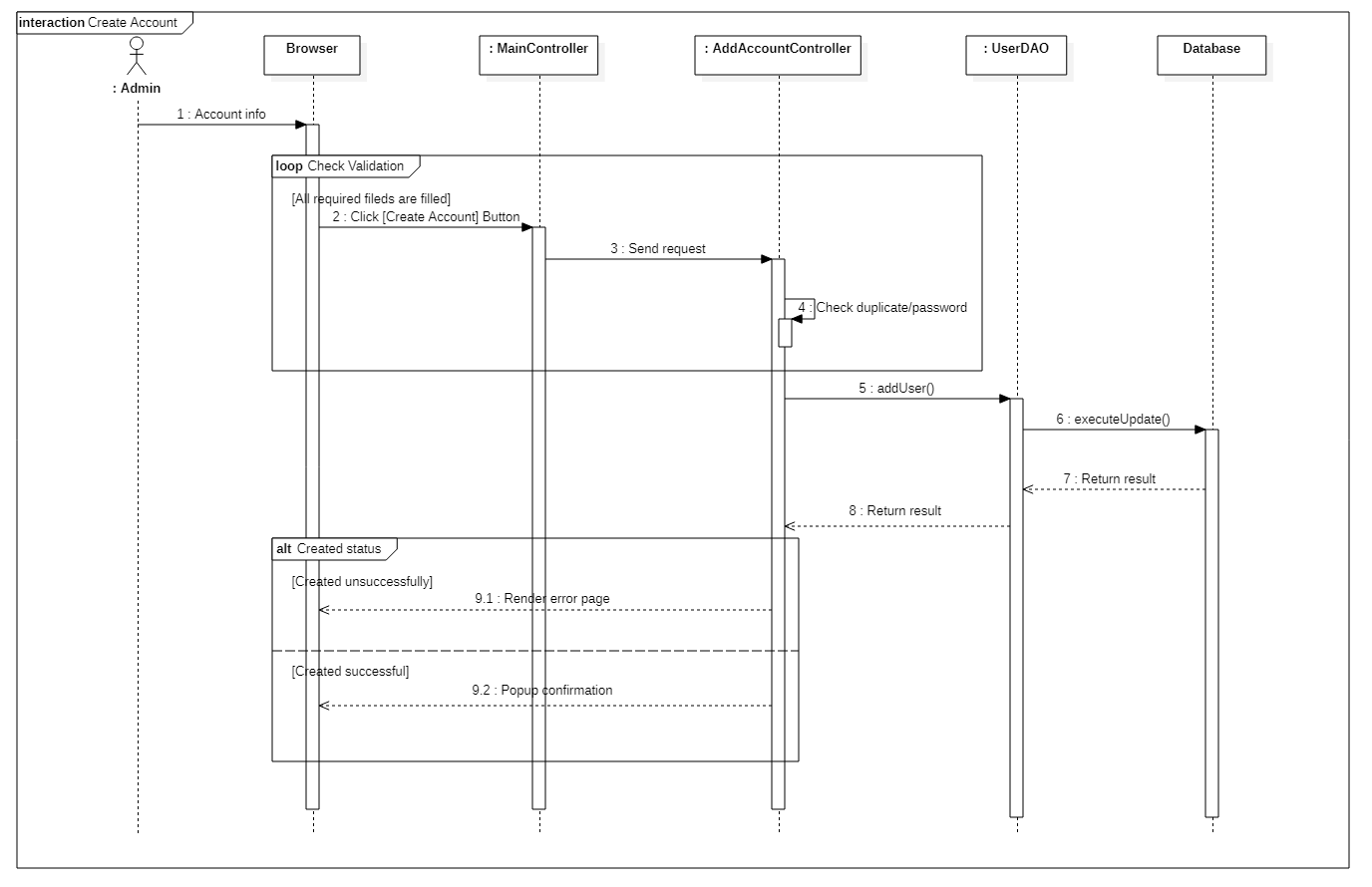
#### UpdateAccountController

| **No** | **Method** | **Description** |
| --- | --- | --- |
| 01 | processRequest() | Process a request from the user. |
| 02 | doGet() | Process a GET request from the user. |
| 03 | doPost() | Process a POST request from the user. |
| 04 | getServletInfo() | Get the servlet’s info |

### c. Sequence Diagram(s)

#### Figure 1.1: Sign up

#### Figure 1.2: Login



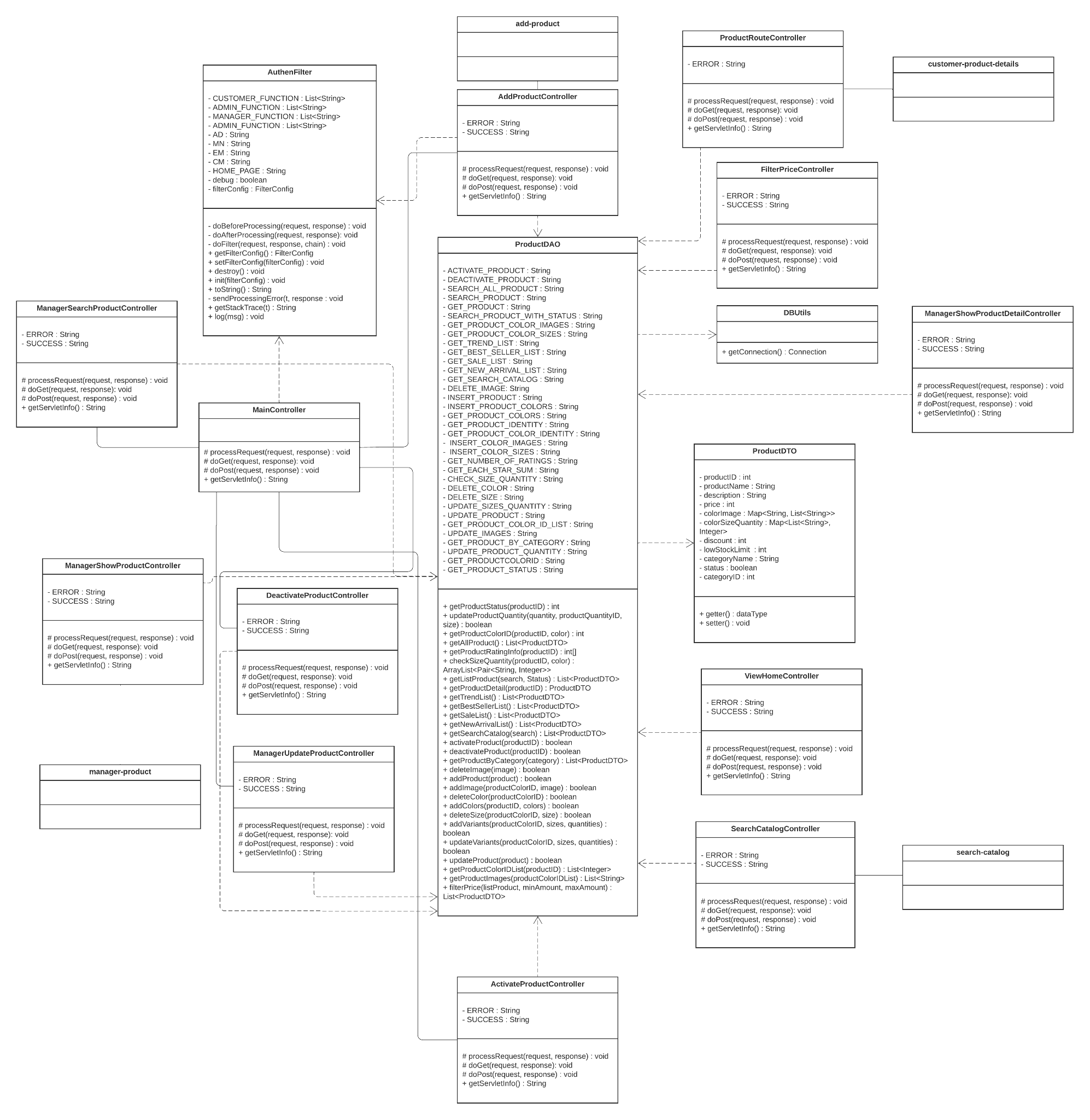
#### Figure 1.3: Create an account - Administrator

### d. Database queries

* String INSERT = INSERT tblUsers(userID, fullName, password, sex, roleID, address, birthday, phone, status) VALUES(?, ?, ?, ?, ?, ?, ?, ?, ?)
* String CHECK\_DUPLICATE = "SELECT fullName FROM tblUsers WHERE userID=?"
* String LOGIN = "SELECT fullName, sex, roleID, address, birthday, phone, status FROM tblUsers WHERE userID=? AND password=?"
* String SEARCH\_USER = "SELECT userID, fullName, sex, roleID, address, birthday, phone, status FROM tblUsers WHERE userID LIKE ?"
* String SEARCH\_USER\_WITH\_ROLE\_ID = "SELECT userID, fullName, sex, roleID, address, birthday, phone, status FROM tblUsers WHERE userID LIKE ? AND roleID LIKE ?"
* String SEARCH\_USER\_WITH\_STATUS = "SELECT userID, fullName, sex, roleID, address, birthday, phone, status FROM tblUsers WHERE userID LIKE ? AND roleID LIKE ? AND status=?"
* String GET\_USER\_BY\_ID = "SELECT userID, fullName, password, sex, roleID, address, birthday, phone, status FROM tblUsers WHERE userID=?"
* String SEARCH\_USER\_ALL = "SELECT userID, fullName, sex, roleID, address, birthday, phone, status FROM tblUsers"
* String SEARCH\_MANAGER = "SELECT userID, fullName, sex, roleID, address, birthday, phone, status FROM tblUsers WHERE userID LIKE ? AND roleID LIKE 'MN'"
* String SEARCH\_MANAGER\_WITH\_STATUS = "SELECT userID, fullName, sex, roleID, address, birthday, phone, status FROM tblUsers WHERE userID LIKE ? AND roleID LIKE 'MN' AND status=?"
* String SEARCH\_MANAGER\_ALL = "SELECT userID, fullName, sex, roleID, address, birthday, phone, status FROM tblUsers WHERE roleID LIKE 'MN'"
* String UPDATE\_ACCOUNT = "UPDATE tblUsers SET fullName=?, sex=?, roleID=?, address=?, birthday=?, phone=? WHERE userID=?"
* String UPDATE\_PASSWORD = "UPDATE tblUsers SET password=? WHERE userID=?"
* String UPDATE\_NAME = "UPDATE tblUsers SET fullName=? WHERE userID=?"
* String UPDATE\_SEX = "UPDATE tblUsers SET sex=? WHERE userID=?"
* String UPDATE\_BIRTHDAY = "UPDATE tblUsers SET birthday=? WHERE userID=?"
* String UPDATE\_ADDRESS = "UPDATE tblUsers SET address=? WHERE userID=?"
* String UPDATE\_PHONE = "UPDATE tblUsers SET phone=? WHERE userID=?"
* String ACTIVATE\_ACCOUNT = "UPDATE tblUsers SET status=1 WHERE userID=?"
* String DEACTIVATE\_ACCOUNT = "UPDATE tblUsers SET status=0 WHERE userID=?"

## 2. Product Management

### a. Class Diagram



### b. Class Specifications

#### ProductDAO

| **No** | **Method** | **Description** |
| --- | --- | --- |
| 01 | addProduct(product) | Add a product to the database |
| 02 | getProductStatus(productID) | Return a product’s status |
| 03 | updateProductQuantity(quantity, productQuantityID, size) | Update a product quantity by its size |
| 04 | getProductColorID(productID, color) | Return ID of a product’s color |
| 05 | getAllProduct() | Return the list of all products |
| 06 | getProductRatingInfo(productID) | Return details of a product’s rating |
| 07 | checkSizeQuantity(productID, color) | Return an array list of all sizes and colors of a product |
| 08 | getListProduct(search, Status) | Return the list of searched products |
| 09 | getProductDetail(productID) | Return the all detailed information of a product |
| 10 | getTrendList() | Get the list of trendy clothes |
| 11 | getBestSellerList() | Get the list of best seller clothes |
| 12 | getSaleList() | Get the list of clothes that are on sales |
| 13 | getNewArrivalList() | Get the list of new clothes |
| 14 | getSearchCatalog(search) | Return the list of searched products for customers |
| 16 | getProductByCategory(category) | Retrieve all products under a category |
| 17 | deactivateProduct(productID) | Change a product’s status to inactive |
| 18 | activateProduct(productID) | Change a product’s status to active |
| 19 | deleteImage(image) | Delete an image of a product |
| 20 | addImage(productColorID, image) | Add an image of a product’s color |
| 21 | filterSearchedProducts(search, minAmount, maxAmount) | Return the searched list of products that matched the criteria |
| 22 | getProductImages(productColorIDList) | Get all product images for the inputted colors |
| 23 | getProductColorIDList(productID) | Get all IDs for a product’s color |
| 24 | deleteColor(productColorID) | Delete a color from a product |
| 25 | addVariants(productColorID, sizes, quantities) | Add a variant for a product |
| 26 | updateVariants(productColorID, sizes, quantities) | Update a variant for a product |
| 27 | updateProduct(product) | Update a product |
| 28 | addColors(productID, colors) | Add a list of colors to a product |
| 29 | deleteSize(productColorID, size) | Delete a size of a product |
| 30 | filterCategoryProducts(category, minAmount, maxAmount) | Return the list of products that matched the criteria in a specific category |

#### ProductDTO

| **No** | **Method** | **Description** |
| --- | --- | --- |
| 01 | getter() | Get an attribute’s value |
| 02 | setter() | Set an attribute’s value |

#### 

#### DBUtils

| **No** | **Method** | **Description** |
| --- | --- | --- |
| 01 | getConnection() | Return a Connection to connect to the database |

#### 

#### AddProductController

| **No** | **Method** | **Description** |
| --- | --- | --- |
| 01 | processRequest() | Process a request from the user. |
| 02 | doGet() | Process a GET request from the user. |
| 03 | doPost() | Process a POST request from the user. |
| 04 | getServletInfo() | Get the servlet’s info |

#### ManagerUpdateProductController

| **No** | **Method** | **Description** |
| --- | --- | --- |
| 01 | processRequest() | Process a request from the user. |
| 02 | doGet() | Process a GET request from the user. |
| 03 | doPost() | Process a POST request from the user. |
| 04 | getServletInfo() | Get the servlet’s info |

#### ProductRouteController

| **No** | **Method** | **Description** |
| --- | --- | --- |
| 01 | processRequest() | Process a request from the user. |
| 02 | doGet() | Process a GET request from the user. |
| 03 | doPost() | Process a POST request from the user. |
| 04 | getServletInfo() | Get the servlet’s info |

#### ManagerShowProductController

| **No** | **Method** | **Description** |
| --- | --- | --- |
| 01 | processRequest() | Process a request from the user. |
| 02 | doGet() | Process a GET request from the user. |
| 03 | doPost() | Process a POST request from the user. |
| 04 | getServletInfo() | Get the servlet’s info |

#### ManagerShowProductDetailController

| **No** | **Method** | **Description** |
| --- | --- | --- |
| 01 | processRequest() | Process a request from the user. |
| 02 | doGet() | Process a GET request from the user. |
| 03 | doPost() | Process a POST request from the user. |
| 04 | getServletInfo() | Get the servlet’s info |

#### 

#### ViewHomeController

| **No** | **Method** | **Description** |
| --- | --- | --- |
| 01 | processRequest() | Process a request from the user. |
| 02 | doGet() | Process a GET request from the user. |
| 03 | doPost() | Process a POST request from the user. |
| 04 | getServletInfo() | Get the servlet’s info |

#### ActivateProductController

| **No** | **Method** | **Description** |
| --- | --- | --- |
| 01 | processRequest() | Process a request from the user. |
| 02 | doGet() | Process a GET request from the user. |
| 03 | doPost() | Process a POST request from the user. |
| 04 | getServletInfo() | Get the servlet’s info |

#### DeactivateProductController

| **No** | **Method** | **Description** |
| --- | --- | --- |
| 01 | processRequest() | Process a request from the user. |
| 02 | doGet() | Process a GET request from the user. |
| 03 | doPost() | Process a POST request from the user. |
| 04 | getServletInfo() | Get the servlet’s info |

#### FilterSearchedProductController

| **No** | **Method** | **Description** |
| --- | --- | --- |
| 01 | processRequest() | Process a request from the user. |
| 02 | doGet() | Process a GET request from the user. |
| 03 | doPost() | Process a POST request from the user. |
| 04 | getServletInfo() | Get the servlet’s info |

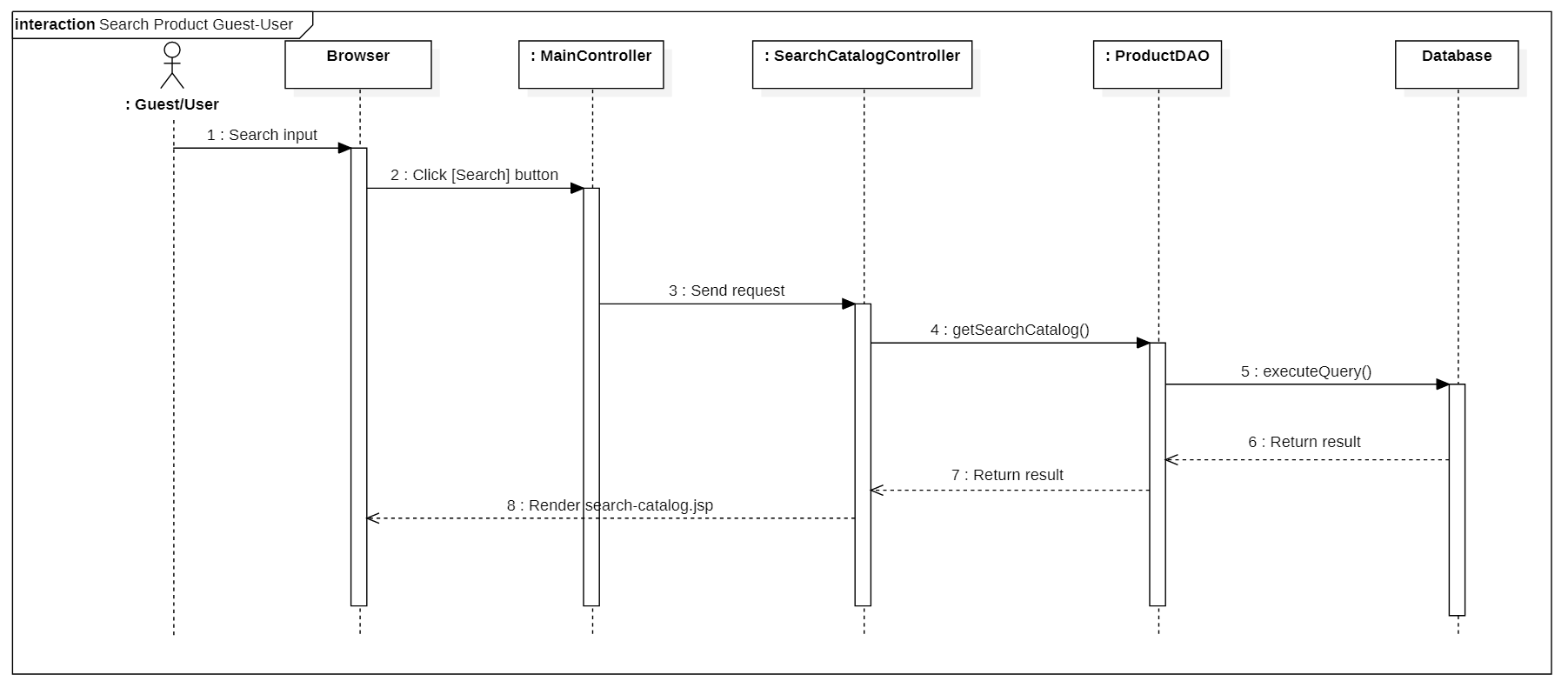
#### FilterCategoryProductController

| **No** | **Method** | **Description** |
| --- | --- | --- |
| 01 | processRequest() | Process a request from the user. |
| 02 | doGet() | Process a GET request from the user. |
| 03 | doPost() | Process a POST request from the user. |
| 04 | getServletInfo() | Get the servlet’s info |

### c. Sequence Diagram(s)

#### Figure 2.1: Create a product

#### Figure 2.2: Search for products - Manager



#### Figure 2.3: Search for products - Customer/Guest

#### Figure 2.4: Add a product - Manager

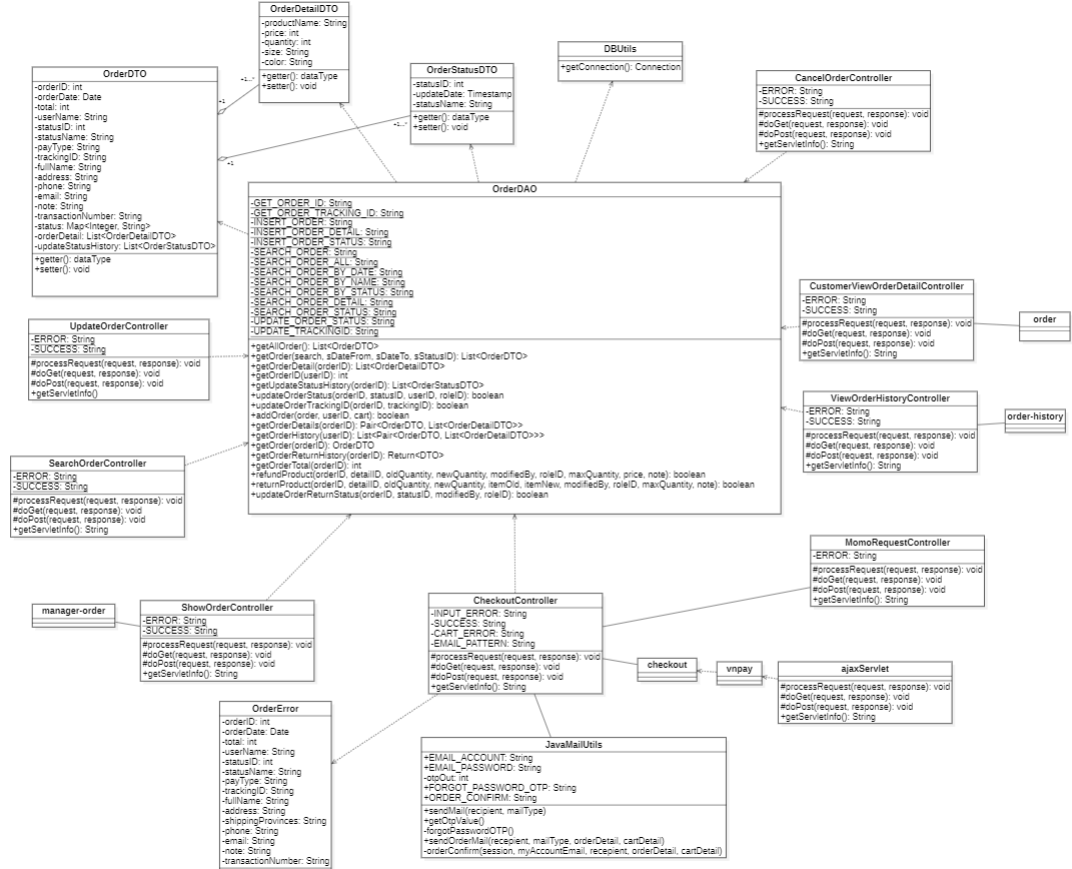
#### Figure 2.5: Update a product - Manager

### d. Database queries

* String INSERT\_PRODUCT = "INSERT INTO tblProduct(productName, description, price, categoryID, discount, lowStockLimit, status) VALUES(?, ?, ?, ?, ?, ?, ?)"
* String GET\_PRODUCT\_IDENTITY = "SELECT IDENT\_CURRENT('tblProduct')"
* INSERT\_PRODUCT\_COLORS = "INSERT INTO tblProductColors(productID, color) VALUES(?, ?)"
* String INSERT\_COLOR\_IMAGES = "INSERT INTO tblColorImage(productColorID, image) VALUES(?, ?)"
* String INSERT\_COLOR\_SIZES = "INSERT INTO tblColorSizes(productColorID, size, quantity) VALUES(?, ?, ?)"
* String SEARCH\_PRODUCT = "SELECT productID, productName, price, categoryName, discount, lowStockLimit, p.status FROM tblProduct p JOIN tblCategory c ON p.categoryID=c.categoryID AND dbo.fuChuyenCoDauThanhKhongDau(productName) LIKE ?"
* String SEARCH\_PRODUCT\_WITH\_STATUS = "SELECT productID, productName, price, categoryName, discount, lowStockLimit, p.status FROM tblProduct p JOIN tblCategory c ON p.categoryID=c.categoryID AND dbo.fuChuyenCoDauThanhKhongDau(productName) LIKE ? AND p.status=?"
* String GET\_SEARCH\_CATALOG = "SELECT p.productID, p.productName, p.price, p.discount, i.image, color, size FROM tblProduct p JOIN tblProductColors pc ON p.productID = pc.productID JOIN tblColorImage i ON pc.productColorID = i.productColorID JOIN tblColorSizes cs ON cs.productColorID = pc.productColorID WHERE dbo.fuChuyenCoDauThanhKhongDau(p.productName) LIKE ?"
* INSERT\_PRODUCT = “INSERT INTO tblProduct(productName, description, price, categoryID, discount, lowStockLimit, status) VALUES(?, ?, ?, ?, ?, ?, ?)”
* GET\_PRODUCT\_IDENTITY = “SELECT IDENT\_CURRENT('tblProduct')”
* INSERT\_PRODUCT\_COLORS = “INSERT INTO tblProductColors(productID, color) VALUES(?, ?)”
* GET\_PRODUCT\_COLOR\_IDENTITY = “SELECT IDENT\_CURRENT('tblProductColors')”
* INSERT\_COLOR\_IMAGES = “INSERT INTO tblColorImage(productColorID, image) VALUES(?, ?)”
* INSERT\_COLOR\_SIZES = ”INSERT INTO tblColorSizes(productColorID, size, quantity) VALUES(?, ?, ?)”
* SEARCH\_ALL\_PRODUCT = “SELECT productID, productName, price, categoryName, discount, lowStockLimit, p.status FROM tblProduct p JOIN tblCategory c ON p.categoryID=c.categoryID”
* UPDATE\_PRODUCT = “UPDATE tblProduct SET productName=?, description=?, price=?, categoryID=?, discount=?, lowStockLimit=?, status=? WHERE productID=?”
* GET\_PRODUCT\_COLOR\_ID\_LIST = “SELECT productColorID FROM tblProductColors WHERE productID = ?”
* getProductImage() = “SELECT image FROM tblColorImage WHERE productColorID in (...?..., …image…)”
* UPDATE\_IMAGES = “UPDATE tblColorImage SET image=? WHERE image=?”
* GET\_PRODUCT = “SELECT productID, productName, price, description, categoryName, discount, lowStockLimit, p.status FROM tblProduct p JOIN tblCategory c ON p.categoryID=c.categoryID AND productID=?”
* GET\_PRODUCT\_COLORS = “SELECT color FROM tblProductColors WHERE productID=?”
* GET\_PRODUCT\_COLOR\_IMAGES = “SELECT color, image FROM tblProduct p JOIN tblProductColors pc ON p.productID = pc.productID AND p.productID = ? JOIN tblColorImage ci ON ci.productColorID = pc.productColorID”
* GET\_PRODUCT\_COLOR\_SIZES = “SELECT color, size, quantity FROM tblProduct p JOIN tblProductColors pc ON p.productID = pc.productID AND p.productID = ? JOIN tblColorSizes cs ON cs.productColorID = pc.productColorID”

## 3. Order Management

### a. Class Diagram



### b. Class Specifications

#### OrderDAO

| **No** | **Method** | **Description** |
| --- | --- | --- |
| 01 | getOrder(search, sDateFrom, sDateTo, sStatusID) | Return a list of orders that matched inputted criteria |
| 02 | getOrderDetail(orderID) | Get order detail for a specific order |
| 03 | addOrder(order, userID, cart) | Insert an order with items into the database |
| 04 | getOrderID(userID) | Get the newest order from the customer |
| 05 | getAllOrder() | Get all orders |
| 06 | getUpdateStatusHistory(orderID) | Return a list of statuses of an order |
| 07 | updateOrderStatus(orderID, statusID, userID, roleID) | Update status of an order |
| 08 | updateOrderTrackingID(orderID, trackingID) | Update a tracking ID of an order |
| 09 | getOrderDetails(orderID) | Return a pair in which the order is the key and order details are the values. |
| 10 | getOrderHistory(userID) | Return order history of a customer |
| 11 | getOrder(orderID) | Get an order by its order ID |
| 12 | getOrderReturnHistory(orderID) | Retrieve the product return/exchanged history for an order of a customer |
| 13 | getOrderTotal(orderID) | Return the total price of an order |
| 14 | refundProduct(orderID, detailID, oldQuantity,newQuantity, modifiedBy, roleID, maxQuantity, price, note) | Add a product to the return history and update the quantity in order detail |
| 15 | returnProduct(orderID, detailID, oldQuantity, newQuantity, itemOld, itemNew, modifiedBy,roleID, maxQuantity, note) | Add a product to the exchanged history and update the quantity in order detail |
| 16 | updateOrderReturnStatus(orderID, statusID, modifiedBy, roleID) | Update the order’s status to “Đã đổi/trả” |

#### OrderDTO

| **No** | **Method** | **Description** |
| --- | --- | --- |
| 01 | getter() | Get an attribute’s value |
| 02 | setter() | Set an attribute’s value |

#### OrderDetailDTO

| **No** | **Method** | **Description** |
| --- | --- | --- |
| 01 | getter() | Get an attribute’s value |
| 02 | setter() | Set an attribute’s value |

#### OrderStatusDTO

| **No** | **Method** | **Description** |
| --- | --- | --- |
| 01 | getter() | Get an attribute’s value |
| 02 | setter() | Set an attribute’s value |

#### OrderError

| **No** | **Method** | **Description** |
| --- | --- | --- |
| 01 | getter() | Get an attribute’s value |
| 02 | setter() | Set an attribute’s value |

#### 

#### DBUtils

| **No** | **Method** | **Description** |
| --- | --- | --- |
| 01 | getConnection() | Return a Connection to connect to the database |

#### 

#### SearchOrderController

| **No** | **Method** | **Description** |
| --- | --- | --- |
| 01 | processRequest() | Process a request from the user. |
| 02 | doGet() | Process a GET request from the user. |
| 03 | doPost() | Process a POST request from the user. |
| 04 | getServletInfo() | Get the servlet’s info |

#### UpdateOrderController

| **No** | **Method** | **Description** |
| --- | --- | --- |
| 01 | processRequest() | Process a request from the user. |
| 02 | doGet() | Process a GET request from the user. |
| 03 | doPost() | Process a POST request from the user. |
| 04 | getServletInfo() | Get the servlet’s info |

#### CancelOrderController

| **No** | **Method** | **Description** |
| --- | --- | --- |
| 01 | processRequest() | Process a request from the user. |
| 02 | doGet() | Process a GET request from the user. |
| 03 | doPost() | Process a POST request from the user. |
| 04 | getServletInfo() | Get the servlet’s info |

#### ViewOrderHistoryController

| **No** | **Method** | **Description** |
| --- | --- | --- |
| 01 | processRequest() | Process a request from the user. |
| 02 | doGet() | Process a GET request from the user. |
| 03 | doPost() | Process a POST request from the user. |
| 04 | getServletInfo() | Get the servlet’s info |

#### CheckoutController

| **No** | **Method** | **Description** |
| --- | --- | --- |
| 01 | processRequest() | Process a request from the user. |
| 02 | doGet() | Process a GET request from the user. |
| 03 | doPost() | Process a POST request from the user. |
| 04 | getServletInfo() | Get the servlet’s info |

#### MomoRequestController

| **No** | **Method** | **Description** |
| --- | --- | --- |
| 01 | processRequest() | Process a request from the user. |
| 02 | doGet() | Process a GET request from the user. |
| 03 | doPost() | Process a POST request from the user. |
| 04 | getServletInfo() | Get the servlet’s info |

#### CustomerViewOrderDetailController

| **No** | **Method** | **Description** |
| --- | --- | --- |
| 01 | processRequest() | Process a request from the user. |
| 02 | doGet() | Process a GET request from the user. |
| 03 | doPost() | Process a POST request from the user. |
| 04 | getServletInfo() | Get the servlet’s info |

#### ShowOrderController

| **No** | **Method** | **Description** |
| --- | --- | --- |
| 01 | processRequest() | Process a request from the user. |
| 02 | doGet() | Process a GET request from the user. |
| 03 | doPost() | Process a POST request from the user. |
| 04 | getServletInfo() | Get the servlet’s info. |

#### ajaxServlet

| **No** | **Method** | **Description** |
| --- | --- | --- |
| 01 | processRequest() | Process a request from the user. |
| 02 | doGet() | Process a GET request from the user. |
| 03 | doPost() | Process a POST request from the user. |
| 04 | getServletInfo() | Get the servlet’s info |

#### JavaMailUtils

| **No** | **Method** | **Description** |
| --- | --- | --- |
| 01 | sendMail() | Send a “Forgot Password” mail to the customer. |
| 02 | getOtpValue() | Return an OPT value. |
| 03 | forgotPasswordOTP() | Return a “ForgotPassword” mail content |
| 04 | orderConfirm(session, myAccountEmail, recipient, orderDetail, cartDetail) | Return a “Order Confirmation” mail content |
| 05 | sendOrderMail(recipient, mailType, orderDetail, cartDetail) | Send a “Order Confirmation” mail to the customer |

### c. Sequence Diagram(s)

#### Figure 3.1: Update an order

#### Figure 3.2: View order history

### d. Database queries

* String UPDATE\_TRACKINGID = "UPDATE tblOrder SET trackingID = ? WHERE orderID = ?"
* String SEARCH\_ORDER\_ALL = "SELECT v1.orderID, orderDate, total, userID, fullName, statusID, statusName, payType, trackingID, [orderFullName], [address], phone, email, note, transactionNumber FROM currentStatusRow v1 JOIN orderReview v2 ON v1.ID = v2.ID "
* String SEARCH\_ORDER\_BY\_ID = "SELECT v1.orderID, orderDate, total, userID, fullName, statusID, statusName, payType, trackingID, [orderFullName], [address], phone, email, note, transactionNumber FROM currentStatusRow v1 JOIN orderReview v2 ON v1.ID = v2.ID WHERE v1.orderID = ?"
* String SEARCH\_ORDER\_BY\_STATUS = "SELECT v1.orderID, orderDate, total, userID, fullName, statusID, statusName, payType, trackingID, [orderFullName], [address], phone, email, note, transactionNumber FROM currentStatusRow v1 JOIN orderReview v2 ON v1.ID = v2.ID AND statusID = ? AND [TenKhongDau] LIKE '%' + [dbo].[fuChuyenCoDauThanhKhongDau](?) + '%'";
* String SEARCH\_ORDER = "SELECT v1.orderID, orderDate, total, userID, fullName, statusID, statusName, payType, trackingID, [orderFullName], [address], phone, email, note, transactionNumber FROM currentStatusRow v1 JOIN orderReview v2 ON v1.ID = v2.ID WHERE (orderDate BETWEEN ? AND ?) AND statusID = ? AND [TenKhongDau] LIKE '%' + [dbo].[fuChuyenCoDauThanhKhongDau](?) + '%'"
* String SEARCH\_ORDER\_BY\_DATE = "SELECT v1.orderID, orderDate, total, userID, fullName, statusID, statusName, payType, trackingID, [orderFullName], [address], phone, email, note, transactionNumber FROM currentStatusRow v1 JOIN orderReview v2 ON v1.ID = v2.ID WHERE (orderDate BETWEEN ? AND ?) AND [TenKhongDau] LIKE '%' + [dbo].[fuChuyenCoDauThanhKhongDau](?) + '%'"
* String UPDATE\_ORDER\_STATUS = "INSERT INTO tblOrderStatusUpdate(statusID, orderID, updateDate, modifiedBy, roleID) VALUES (?, ?, GETDATE(), ?, ?)"
* String INSERT\_ORDER\_STATUS = "INSERT INTO tblOrderStatusUpdate(statusID, orderID, updateDate, modifiedBy, roleID) VALUES (?, ?, GETDATE(), 'System', '')"
* String SEARCH\_ORDER\_BY\_NAME = "SELECT v1.orderID, orderDate, total, userID, fullName, statusID, statusName, payType, trackingID, [orderFullName], [address], phone, email, note, transactionNumber FROM currentStatusRow v1 JOIN orderReview v2 ON v1.ID = v2.ID AND [TenKhongDau] LIKE '%' + [dbo].[fuChuyenCoDauThanhKhongDau](?) + '%'"
* String SEARCH\_ORDER\_DETAIL = "SELECT detailID, t1.productID, productName, t1.price, quantity, size, color tblOrderDetail t1 JOIN tblProduct t2 ON t1.productID = t2.productID WHERE orderID = ? AND quantity > 0"
* String SEARCH\_ORDER\_STATUS = "SELECT t1.statusID, updateDate, statusName, modifiedBy, roleID FROM tblOrderStatusUpdate t1 JOIN tblOrderStatus t2 ON t1.statusID = t2.statusID WHERE orderID = ?"
* String INSERT\_ORDER = "INSERT INTO tblOrder(orderDate, total, userID, payType, fullName, [address], phone, email, note, transactionNumber) VALUES(?, ?, ?, ?, ?, ?, ?, ?, ?, ?)"
* String INSERT\_ORDER\_DETAIL = "INSERT INTO tblOrderDetail(price, quantity, size, color, orderID, productID) VALUES(?, ?, ?, ?, ?, ?)"
* String GET\_ORDER\_ID = "SELECT TOP 1 orderID FROM tblOrder WHERE userID LIKE ? + '%' ORDER BY orderID DESC"
* String GET\_ORDER\_TRACKING\_ID = "SELECT trackingID FROM tblOrder WHERE orderID = ?";
* String GET\_ORDER\_HISTORY = "SELECT v2.orderID, orderDate, total, statusName, payType FROM currentStatusRow v1 JOIN orderReview v2 ON v1.ID = v2.ID WHERE v2.orderID in (SELECT orderID FROM tblOrder WHERE userID = ?) ORDER BY v2.orderID desc"
* String GET\_ORDER\_DETAIL = "SELECT top 1 with ties p.productID, productName, od.price, od.quantity, od.color, od.size, image FROM tblProduct p JOIN tblOrderDetail od ON p.productID = od.productID JOIN tblProductColors pc ON p.productID = pc.productID AND od.color = pc.color JOIN tblColorImage ci ON ci.productColorID = pc.productColorID WHERE orderID = ? ORDER BY ROW\_NUMBER() over (partition by p.productID, od.color, od.size order by image)"
* String GET\_ORDER = "SELECT v1.orderID, orderDate, total, statusID, statusName, payType, trackingID, fullName, address, phone, email, note FROM currentStatusRow v1 JOIN orderReview v2 ON v1.ID = v2.ID WHERE v2.orderID = ?"
* String GET\_STATUS\_HISTORY = "SELECT statusID, updateDate FROM tblOrderStatusUpdate WHERE orderID = ?"
* String UPDATE\_ORDER\_DETAIL = "UPDATE tblOrderDetail SET quantity = ? WHERE detailID = ?"
* String UPDATE\_ORDER\_TOTAL = "UPDATE tblOrder SET total = ? WHERE orderID = ?"
* String INSERT\_ORDER\_RETURN = "INSERT INTO tblReturns(detailID, quantity, returnType, returnDate, note) VALUES(?, ?, ?, GETDATE(), ?)"
* String GET\_ORDER\_TOTAL = "SELECT total FROM tblOrder WHERE orderID = ?"
* String CHECK\_ORDER\_DUPLICATE\_ITEM = "SELECT detailID, quantity FROM tblOrderDetail WHERE orderID = ? AND productID = ? AND color LIKE ? AND size LIKE ?"
* String GET\_RETURN\_HISTORY = "SELECT t1.detailID, t1.productID, productName, t1.price, t1.quantity, size, color, t3.quantity AS [returnQuantity], returnType, returnDate, note FROM tblOrderDetail t1 JOIN tblProduct t2 ON t1.productID = t2.productID JOIN tblReturns t3 ON t1.detailID = t3.detailID WHERE orderID = ?"

# III. Database Tables

## 1. tblRoles

This table contains information about the user roles.

| **#** | **Field name** | **Type** | **Size** | **Unique** | **Not Null** | **PK/FK** | **Notes** |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | roleID | nvarchar | 10 | yes | yes | PK |  |
| 2 | roleName | nvarchar | 50 | yes | yes |  |  |
| 3 | status | bit |  |  | yes |  |  |

## 2. tblUsers

This table contains information about the users.

| **#** | **Field name** | **Type** | **Size** | **Unique** | **Not Null** | **PK/FK** | **Notes** |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | userID | char | 100 | yes | yes | PK |  |
| 2 | fullName | nvarchar | 100 |  | yes |  |  |
| 3 | password | varchar | 100 |  | yes |  |  |
| 4 | sex | bit |  |  | yes |  |  |
| 5 | roleID | nvarchar | 10 |  | yes | FK | tblRoles |
| 6 | address | nvarchar | 150 |  | no |  |  |
| 7 | birthday | date |  |  | yes |  |  |
| 8 | phone | varchar | 20 | yes | yes |  |  |
| 9 | status | bit |  |  | yes |  |  |

## 3. tblOrderStatus

This table contains the types of order status.

| **#** | **Field name** | **Type** | **Size** | **Unique** | **Not Null** | **PK/FK** | **Notes** |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | statusID | int |  | yes | yes | PK |  |
| 2 | statusName | nvarchar | 50 | yes | yes |  |  |

## 4. tblOrder

This table contains data of the orders.

| **#** | **Field name** | **Type** | **Size** | **Unique** | **Not Null** | **PK/FK** | **Notes** |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | orderID | int |  | yes | yes | PK | IDENTITY(1,1) |
| 2 | orderDate | date |  |  | yes |  |  |
| 3 | total | int |  |  | yes |  |  |
| 4 | userID | char | 100 |  | yes | FK | tblUsers |
| 5 | payType | nvarchar | 50 |  | yes |  |  |
| 6 | trackingID | varchar | 40 | yes | yes |  |  |
| 7 | fullName | nvarchar | 100 |  | yes |  |  |
| 8 | address | nvarchar | 150 |  | yes |  |  |
| 9 | phone | varchar | 20 |  | yes |  |  |
| 10 | email | char | 100 |  | yes | FK | tblOrderStatus |
| 11 | note | nvarchar | 100 |  |  |  |  |
| 12 | transactionNumber | varchar | 100 | yes |  |  |  |

## 5. tblCategory

This table contains the categories of the product.

| **#** | **Field name** | **Type** | **Size** | **Unique** | **Not Null** | **PK/FK** | **Notes** |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | categoryID | int |  | yes | yes | PK | IDENTITY(1,1) |
| 2 | categoryName | nvarchar | 50 | yes | yes |  |  |
| 3 | order | int |  | yes | yes |  |  |
| 4 | status | bit |  |  | yes |  |  |

## 6. tblProduct

This table contains overall information of a product.

| **#** | **Field name** | **Type** | **Size** | **Unique** | **Not Null** | **PK/FK** | **Notes** |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | productID | int |  | yes | yes | PK | IDENTITY(1,1) |
| 2 | productName | nvarchar | 50 |  | yes |  |  |
| 3 | description | nvarchar | 500 |  |  |  |  |
| 4 | price | int |  |  | yes |  |  |
| 5 | categoryID | int |  |  | yes | FK | tblCategory |
| 6 | discount | int |  |  |  |  |  |
| 7 | lowStockLimit | int |  |  |  |  |  |
| 8 | status | bit |  |  | yes |  |  |

## 7. tblProductColors

This table contains the product details.

| **#** | **Field name** | **Type** | **Size** | **Unique** | **Not Null** | **PK/FK** | **Notes** |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | productColorID | int |  | yes | yes | PK | IDENTITY(1,1) |
| 2 | productID | int |  |  | yes | FK | tblProduct |
| 3 | color | nvarchar | 50 |  | yes |  |  |

## 

## 8. tblColorImage

This table contains the product details.

| **#** | **Field name** | **Type** | **Size** | **Unique** | **Not Null** | **PK/FK** | **Notes** |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | colorImageID | int |  | yes | yes | PK | IDENTITY(1,1) |
| 2 | productColorID | int |  |  | yes | FK | tblProductColors |
| 3 | image | nvarchar | 250 |  | yes |  |  |

## 

## 9. tblColorSizes

This table contains the product details.

| **#** | **Field name** | **Type** | **Size** | **Unique** | **Not Null** | **PK/FK** | **Notes** |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | colorSizeID | int |  | yes | yes | PK | IDENTITY(1,1) |
| 2 | productColorID | int |  |  | yes | FK | tblProductColors |
| 3 | size | varchar | 50 |  | yes |  |  |
| 4 | quantity | int |  |  | yes |  |  |

## 10. tblRating

This table contains the products’ ratings.

| **#** | **Field name** | **Type** | **Size** | **Unique** | **Not Null** | **PK/FK** | **Notes** |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | id | int |  | yes | yes | PK | IDENTITY(1,1) |
| 2 | productID | int |  |  | yes | FK | tblProduct |
| 3 | userID | char | 100 |  | yes | FK | tblUsers |
| 4 | orderID | int |  |  | yes | FK | tblOrder |
| 5 | content | nvarchar | 500 |  | yes |  |  |
| 6 | star | int |  |  | yes |  |  |
| 7 | rateDate | date |  |  | yes |  |  |

## 11. tblOrderDetail

This table contains the order details.

| **#** | **Field name** | **Type** | **Size** | **Unique** | **Not Null** | **PK/FK** | **Notes** |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | detailID | int |  | yes | yes | PK | IDENTITY(1,1) |
| 2 | price | int |  |  | yes |  |  |
| 3 | quantity | int |  |  | yes |  |  |
| 4 | orderID | int |  |  | yes | FK | tblOrder |
| 5 | productID | int |  |  | yes | FK | tblProductDetail |
| 6 | size | varchar |  |  | yes | FK | tblProductDetail |
| 7 | color | nvarchar |  |  | yes | FK | tblProductDetail |

## 

## 12. tblOrderStatusUpdate

This table contains detailed order status information.

| **#** | **Field name** | **Type** | **Size** | **Unique** | **Not Null** | **PK/FK** | **Notes** |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | ID | int |  | yes | yes | PK | IDENTITY(1,1) |
| 2 | statusID | int |  |  | yes | FK | tblOrderStatus |
| 3 | orderID | int |  |  | yes | FK | tblOrder |
| 4 | updateDate | smalldatetime |  |  | yes |  |  |
| 5 | modifiedBy | char | 100 |  | yes |  |  |
| 6 | roleID | nvarchar | 50 |  | yes |  |  |

## 13. tblCart

This table contains cart session information.

| **#** | **Field name** | **Type** | **Size** | **Unique** | **Not Null** | **PK/FK** | **Notes** |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | id | int |  | yes | yes | PK | IDENTITY(1,1) |
| 2 | userID | char | 100 |  | yes | FK | tblUsers |
| 3 | fullName | nvarchar | 100 |  | yes |  |  |
| 4 | address | nvarchar | 150 |  | yes |  |  |
| 5 | phone | varchar | 20 |  | yes |  |  |
| 6 | email | char | 100 |  | yes |  |  |
| 7 | note | nvarchar | 200 |  |  |  |  |
| 8 | payType | nvarchar | 50 |  | yes |  |  |

## 14. tblCartItems

This table contains a cart’s item information.

| **#** | **Field name** | **Type** | **Size** | **Unique** | **Not Null** | **PK/FK** | **Notes** |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | id | int |  | yes | yes | PK | IDENTITY(1,1) |
| 2 | productID | int |  |  | yes | FK | tblProduct |
| 3 | sessionID | int |  |  | yes | FK | tblCart |
| 4 | quantity | int |  |  | yes |  |  |
| 5 | size | varchar | 50 |  | yes |  |  |
| 6 | color | nvarchar | 50 |  | yes |  |  |

## 

## 15. tblReturns

This table contains detailed order status information.

| **#** | **Field name** | **Type** | **Size** | **Unique** | **Not Null** | **PK/FK** | **Notes** |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | id | int |  | yes | yes | PK | IDENTITY(1,1) |
| 2 | detailID | int |  |  | yes | FK | tblOrderDetail |
| 3 | quantity | int |  |  | yes |  |  |
| 4 | returnType | nvarchar | 50 |  | yes |  |  |
| 5 | returnDate | date |  |  | yes |  |  |
| 6 | note | nvarchar | 100 |  | yes |  |  |