

**CLOTHING SHOP PROJECT REPORT**

**Software Design Document**

– Hanoi, Summer 2022 –

**Table of Contents**

[I. Overview 6](#_gjdgxs)

[1. Code Packages/Namespaces 6](#_30j0zll)

[2. Coding Conventions 7](#_1fob9te)

[II. Code Designs 7](#_3znysh7)

[1. <Feature/Function Name1> 8](#_2et92p0)

[2. Login 8](#_tyjcwt)

[a. Class Diagram 8](#_3dy6vkm)

[b. Class Specifications 8](#_1t3h5sf)

[c. Sequence Diagram(s) 9](#_4d34og8)

[d. Database queries 9](#_2s8eyo1)

[4. Product detail 10](#_3rdcrjn)

[a. Class Diagram 10](#_26in1rg)

[b. Class Specifications 10](#_lnxbz9)

[c. Sequence Diagram(s) 10](#_1ksv4uv)

[d. Database queries 11](#_2jxsxqh)

[5. Register 11](#_z337ya)

[a. Class Diagram 11](#_3j2qqm3)

[b. Class Specifications 12](#_1y810tw)

[c. Sequence Diagram(s) 12](#_2xcytpi)

[d. Database queries 13](#_3whwml4)

[6. Add to cart 13](#_2bn6wsx)

[a. Class Diagram 13](#_qsh70q)

[b. Class Specifications 13](#_3as4poj)

[c. Sequence Diagram(s) 14](#_2p2csry)

[d. Database queries 14](#_23ckvvd)

[7. Blog List 14](#_ihv636)

[a. Class Diagram 14](#_32hioqz)

[b. Class Specifications 14](#_1hmsyys)

[c. Sequence Diagram(s) 15](#_2grqrue)

[d. Database queries 16](#_3fwokq0)

[8. Blog Detail 16](#_1v1yuxt)

[a. Class Diagram 16](#_4f1mdlm)

[b. Class Specifications 16](#_2u6wntf)

[c. Sequence Diagram(s) 17](#_3tbugp1)

[d. Database queries 17](#_nmf14n)

[9. Logout 17](#_37m2jsg)

[a. Class Diagram 17](#_1mrcu09)

[b.Class Specifications 18](#_46r0co2)

[c. Sequence Diagram(s) 18](#_111kx3o)

[d. Database queries 19](#_206ipza)

[10. Home Page 19](#_4k668n3)

[a. Class Diagram 19](#_2zbgiuw)

[b. Class Specifications 19](#_1egqt2p)

[c. Sequence Diagram(s) 20](#_sqyw64)

[11. Change amount 21](#_1rvwp1q)

[a. Class Diagram 21](#_4bvk7pj)

[b. Class Specifications 21](#_2r0uhxc)

[c. Sequence Diagram(s) 22](#_25b2l0r)

[12. Delete cart 22](#_34g0dwd)

[a. Class Diagram 22](#_1jlao46)

[b. Class Specifications 22](#_43ky6rz)

[c. Sequence Diagram(s) 23](#_xvir7l)

[13. View List User 24](#_1x0gk37)

[a. Class Diagram 24](#_4h042r0)

[b. Class Specifications 24](#_2w5ecyt)

[c. Sequence Diagram(s) 25](#_3vac5uf)

[d. Database queries 26](#_pkwqa1)

[14. Edit Information User 26](#_39kk8xu)

[a. Class Diagram 26](#_1opuj5n)

[b. Class Specifications 26](#_48pi1tg)

[c. Sequence Diagram(s) 27](#_1302m92)

[d. Database queries 28](#_2250f4o)

[15.Set Role User 28](#_haapch)

[a. Class Diagram 28](#_319y80a)

[b. Class Specifications 28](#_1gf8i83)

[c. Sequence Diagram(s) 29](#_2fk6b3p)

[d. Database queries 29](#_3ep43zb)

[16. Block User 30](#_1tuee74)

[a. Class Diagram 30](#_4du1wux)

[b. Class Specifications 30](#_2szc72q)

[c. Sequence Diagram(s) 30](#_3s49zyc)

[d. Database queries 31](#_meukdy)

[17. Reset Password 31](#_36ei31r)

[a. Class Diagram 31](#_1ljsd9k)

[b. Class Specifications 31](#_45jfvxd)

[c. Sequence Diagram(s) 32](#_3jtnz0s)

[d. Database queries 33](#_4iylrwe)

[18. Change Password 33](#_2y3w247)

[a. Class Diagram 33](#_1d96cc0)

[b. Class Specifications 33](#_3x8tuzt)

[c. Sequence Diagram(s) 34](#_rjefff)

[d. Database queries 35](#_1qoc8b1)

[19. User profile 35](#_4anzqyu)

[a. Class Diagram 35](#_2pta16n)

[b. Class Specifications 35](#_14ykbeg)

[c. Sequence Diagram(s) 36](#_243i4a2)

[d. Database queries 37](#_338fx5o)

[20. Order History 37](#_1idq7dh)

[a. Class Diagram 37](#_42ddq1a)

[b. Class Specifications 37](#_2hio093)

[c. Sequence Diagram(s) 38](#_3gnlt4p)

[d. Database queries](#_4fsjm0b)

24. Manager Product

1. Class Diagram
2. Class Specifications
3. Sequence Digrams
4. Database queries

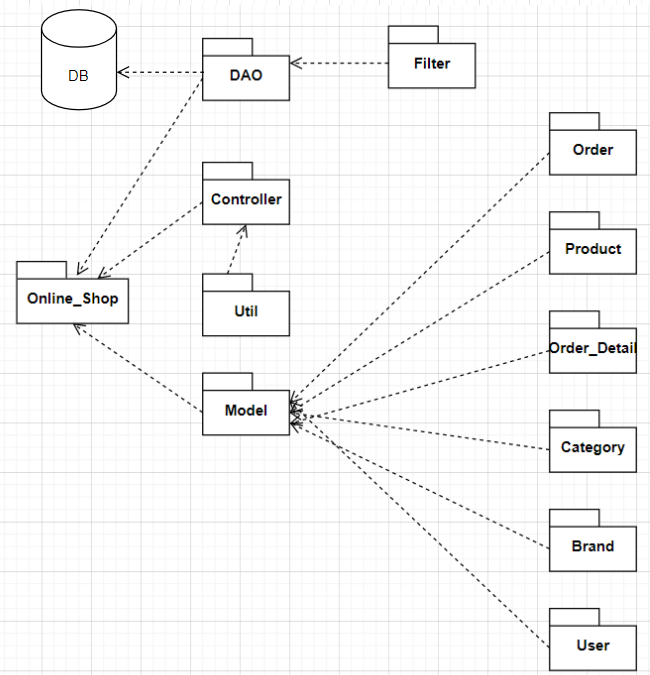
[III. Database Design 40](#_2uxtw84)

[1. Database Schema 40](#_1a346fx)

[2. Table Description 40](#_3u2rp3q)

# I. Overview

## 1. Code Packages/Namespaces



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

| **No** | **Package** | **Description** |
| --- | --- | --- |
| *01* | *Controllers* | This package contains all controllers for this MVC2 project and authentication or dispatcher filters  Servlet Naming convention: Starting with a verb.  Ex: LoginController, SignUpController,... |
| *02* | *Views* | This package contains all views (HTML,CSS,JS,JSP file) for this project  HTML/CSS/JS/JSP file naming convention: first letter is a lowercase letter, using camelCase naming |
| *03* | *Model* | This package provide the connection to the Database, Extract data for the system  Naming convention: first letter is a lowercase letter, the name is using camelCase naming  This package .java files which allow developers to skip writing DAOs for their persistence objects when they usingJDBC. |
| *04* | *Utils* | This package contains DBcontext.java file which is a bridge connection between database and programming and contains sendEmail.java allows users to send email  Naming convention: first letter is a lowercase letter, the name is using camelCase naming |
| *05* | *Filter* | This package handles the Authentication that helps avoid duplicate in controller  Naming convention: first letter is a lowercase letter, the name is using camelCase naming |

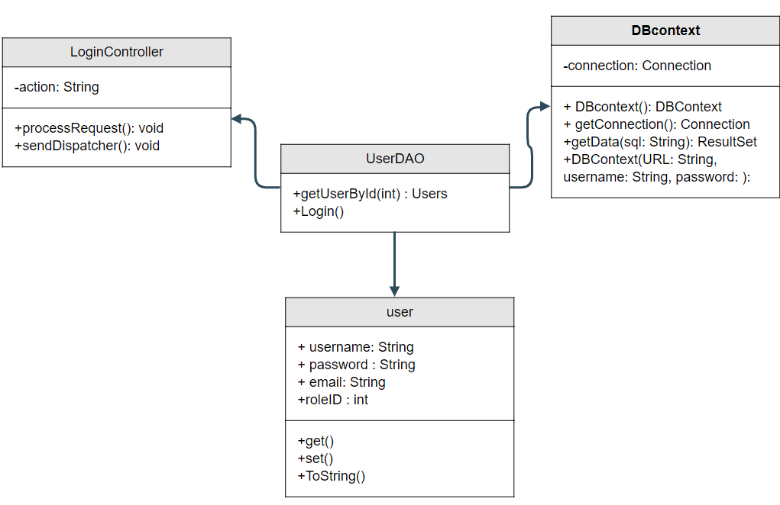
## 2. Coding Conventions

# II. Code Designs

## 1. <Feature/Function Name1>

## 2. Login

### Class Diagram



### Class Specifications

UserController

| No | Method | Description |
| --- | --- | --- |
| 01 | processRequest() | get all new information from user Form and update the database |

UserDAO

[Provide the detailed description for the class methods]

| No | Method | Description |
| --- | --- | --- |
| 01 | getUserList() | get the list of user |
| 02 | getUserById(int uId) | get the user where the user id is |

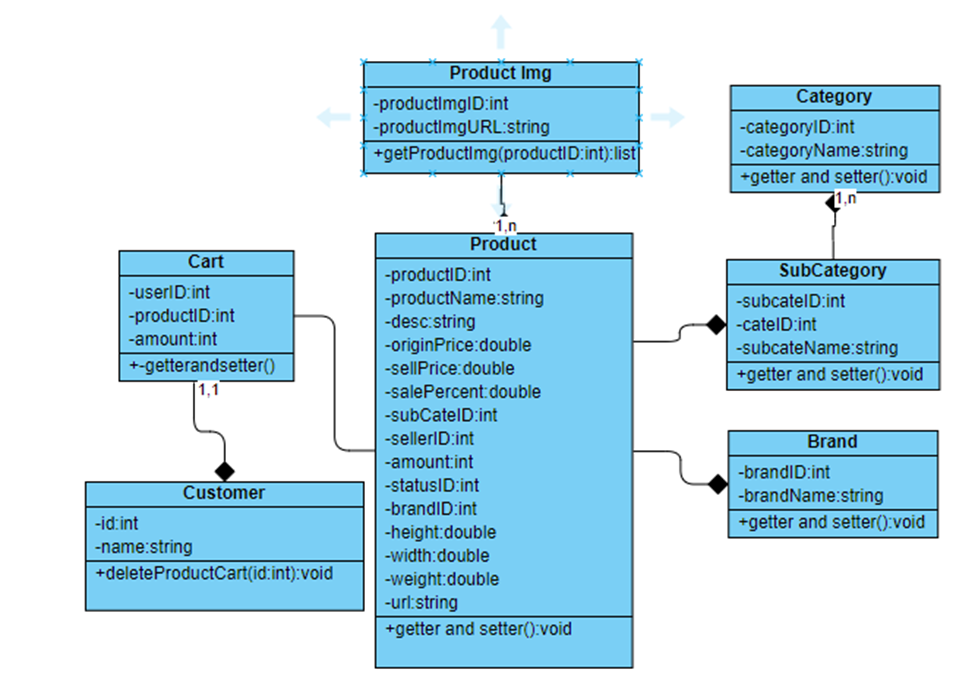
### Sequence Diagram(s)

### Database queries

**"select \* from [User] where username=? and password=?"**

**3.Cart Detail**

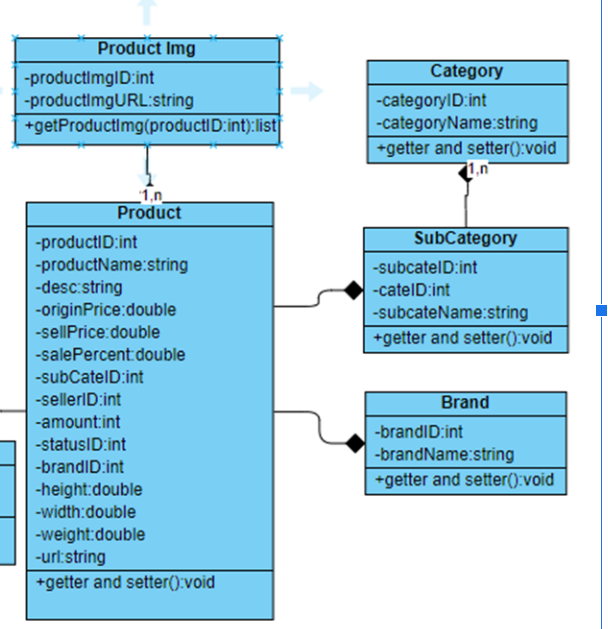
1. **Class Diagram**



### 

## 4. Product detail

### Class Diagram



### Class Specifications

| No | Method | Description |
| --- | --- | --- |
| 01 | processRequest() |  |

ProductDAO

| No | Method | Description |
| --- | --- | --- |
| 01 | getProductByID(int id) | get Product by ID |

### 

### c. Sequence Diagram(s)

### 

### d. Database queries

"select p.\*,s.SubCategoryName,c.\*,b.BrandName from Product p join SubCategory s \n"

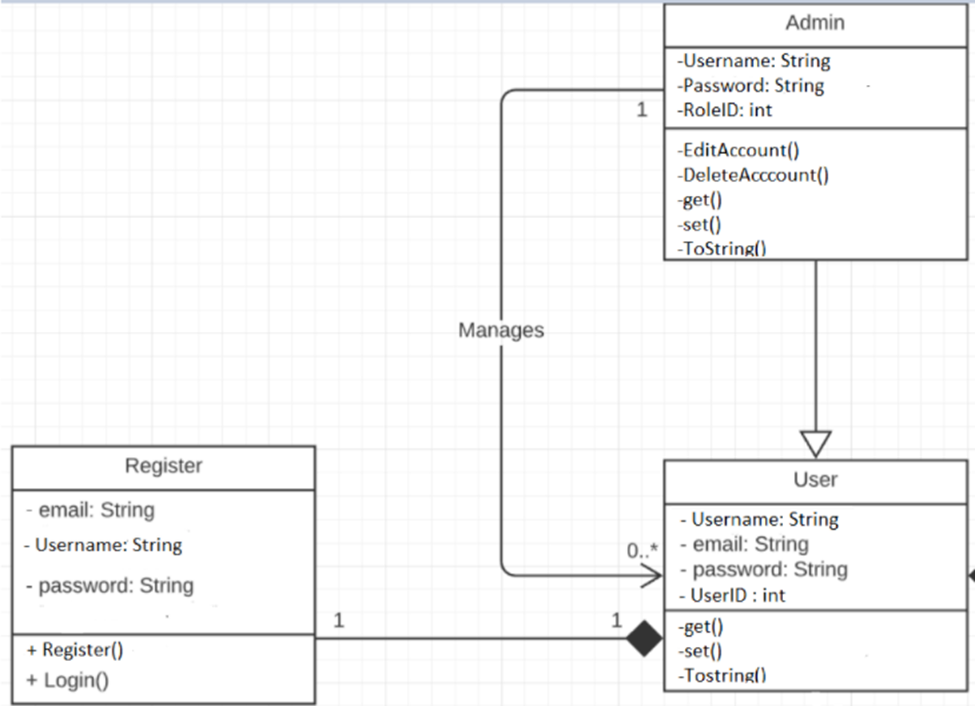
+ "on p.SubCategoryID = s.SubCategoryID\n"

+ "join Brand b on b.BrandID = p.BrandID\n"

+ "join Category c on c.CategoryID = s.CategoryID"

## 5. Register

### **a. Class Diagram**



### b. Class Specifications

UserController

| No | Method | Description |
| --- | --- | --- |
| 01 | processRequest() |  |

UserDAO

[Provide the detailed description for the class methods]

| No | Method | Description |
| --- | --- | --- |
| 01 | Register( String username,String password, String mai) | insert a new user with Name , Pass, Mail) |
| 02 | checkExistAccount(String xName) | - check if the username is already exist  - return null |

### 

### c. Sequence Diagram(s)

### 

### d. Database queries

**"**INSERT INTO [dbo].[Users]

([Username

,[Password]

,[email]

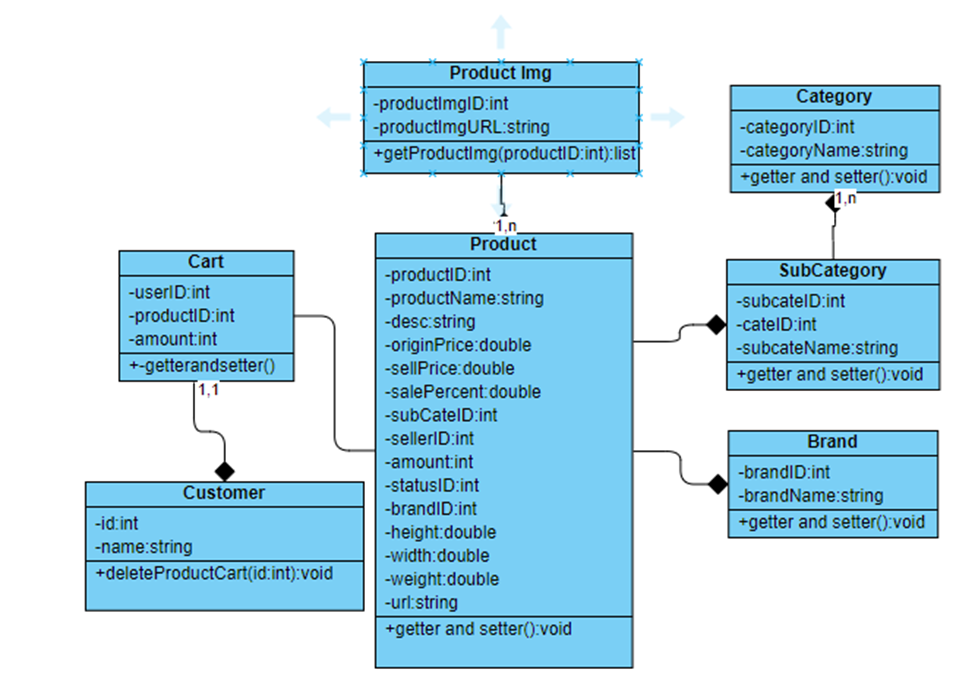
,[RoleID]

,[isBlock])

VALUES (?,?,?,?,?)"**"**

## 6. Add to cart

### Class Diagram



### Class Specifications

CartDetailController

| No | Method | Description |
| --- | --- | --- |
| 01 | processRequest() |  |

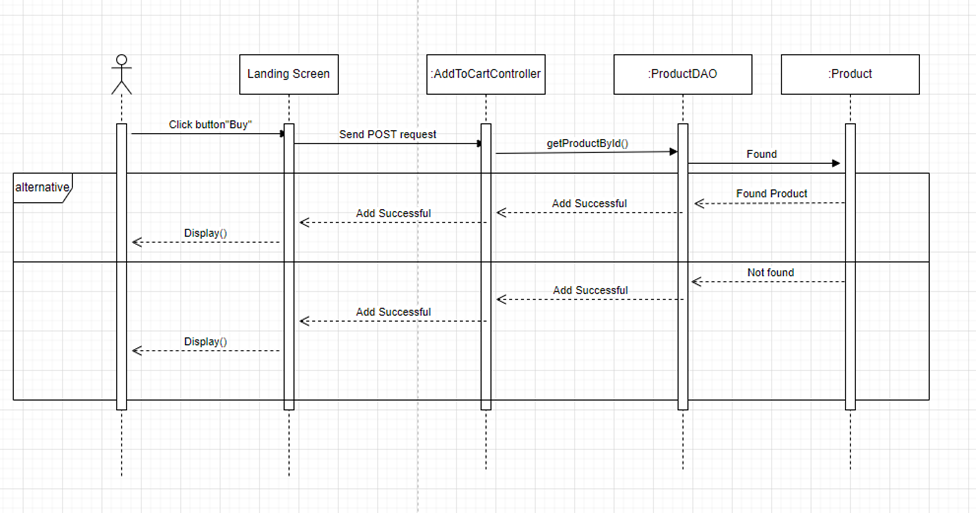
CartDAO

[Provide the detailed description for the class methods]

| No | Method | Description |
| --- | --- | --- |
| *01* | *deleteProductCart()* | *Remove a product() from cart* |

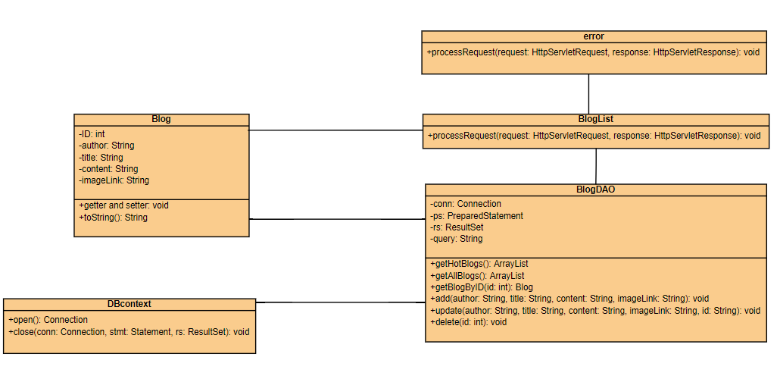
### 

### c. Sequence Diagram(s)



## Blog List

### Class Diagram



### Class Specifications

- BlogList class

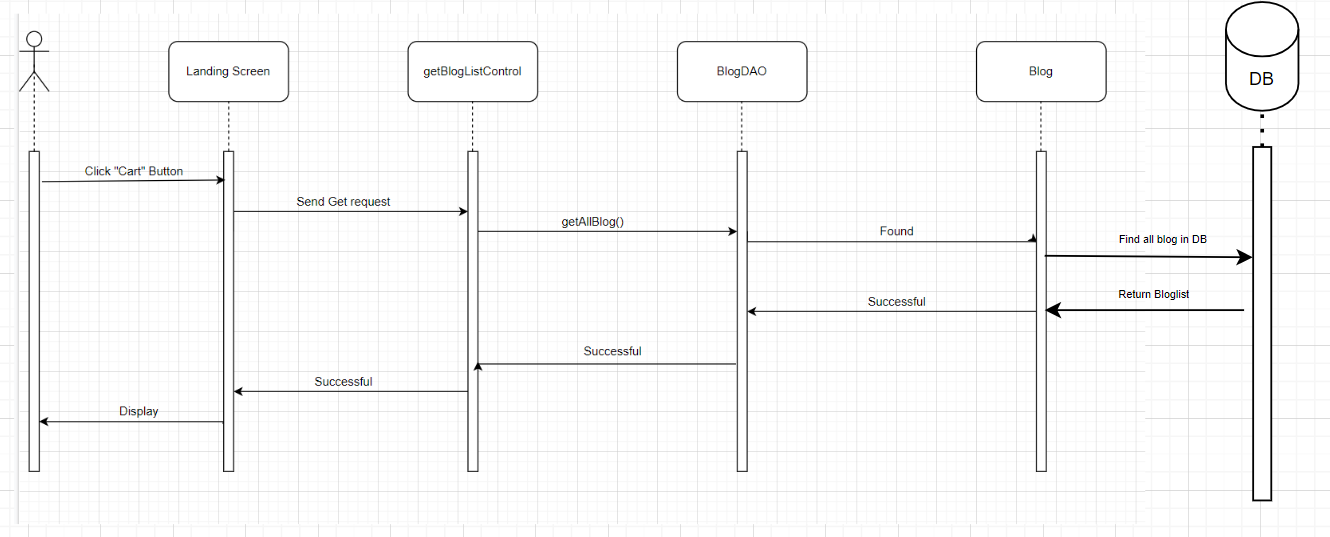
| **No** | **Method** | **Description** |
| --- | --- | --- |
| 01 | processRequest() | get blog list from the database to display on BlogList.jsp |

- BlogDAO class

| **No** | **Method** | **Description** |
| --- | --- | --- |
| 01 | getBlogByID(int id) | Return: Blog Detail where blog id contain parameter  Input: ID of blog |
| 02 | getHotBlogs() | Return: top 3 newest blog from database |
| 03 | getAllBlog() | Return: All blog in database |
| 04 | add(String author, String title, String content, String imageLink) | Add blog with detail entered in addBlog.jsp page  Return: null  Input: author, title, content, imageLink |
| 05 | update(String author, String title, String content, String imageLink, String id) | Update blog with detail entered in updateBlog.jsp page  Return: null  Input: author, title, content, imageLink, id |
| 06 | delete(int id) | Update blog with id entered in blogManage.jsp  Return: null  Input: id |

### 

### c. Sequence Diagram(s)



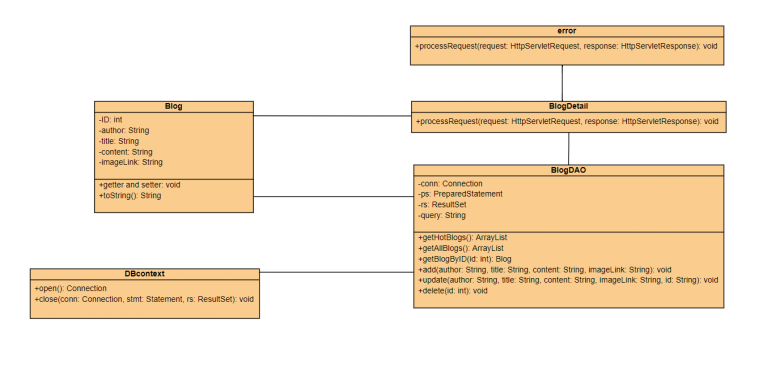
### 

### d. Database queries

**Select \* from Blog desc**

## 8. Blog Detail

### Class Diagram



### Class Specifications

- BlogDetail class

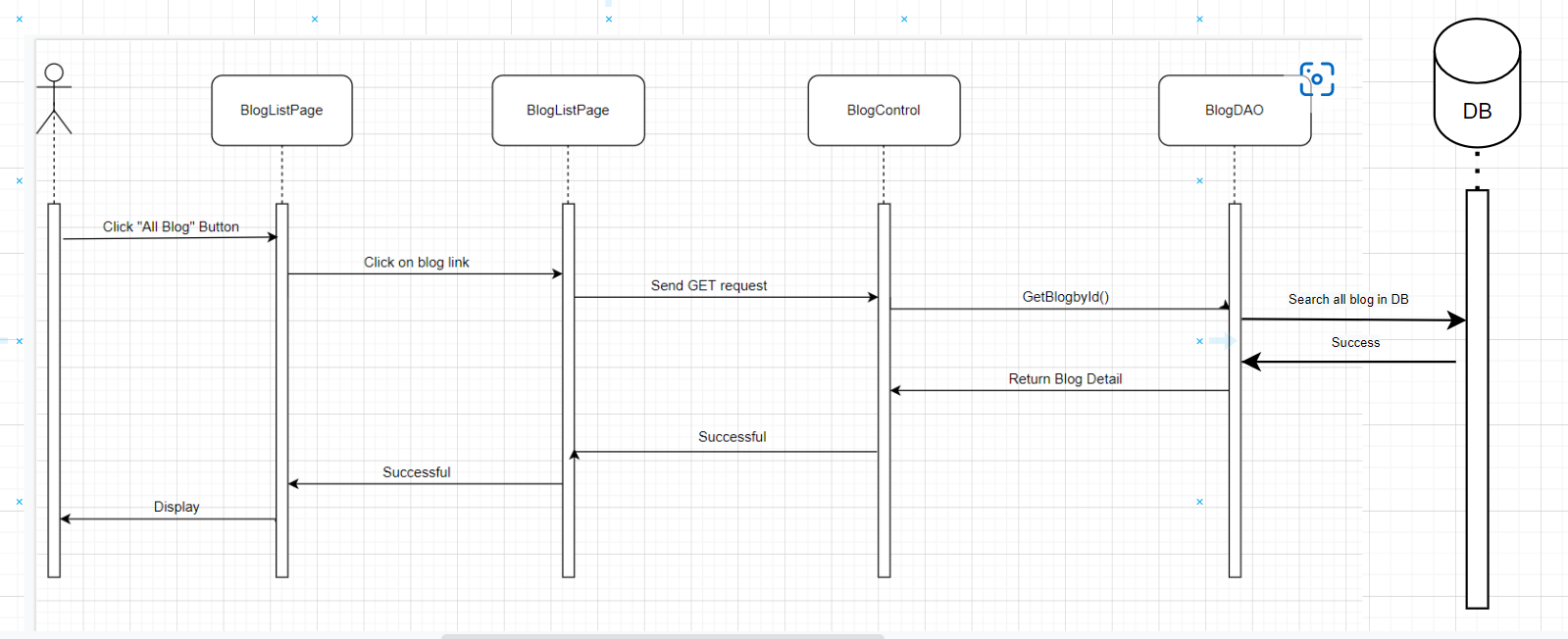
| **No** | **Method** | **Description** |
| --- | --- | --- |
| 01 | processRequest() | get blog detail from the database to display on BlogDetail.jsp |

- BlogDAO class

| **No** | **Method** | **Description** |
| --- | --- | --- |
| 01 | getBlogByID(int id) | Return: Blog Detail where blog id contain parameter  Input: ID of blog |
| 02 | getHotBlogs() | Return: top 3 newest blog from database |
| 03 | getAllBlog() | Return: All blog in database |
| 04 | add(String author, String title, String content, String imageLink) | Add blog with detail entered in addBlog.jsp page  Return: null  Input: author, title, content, imageLink |
| 05 | update(String author, String title, String content, String imageLink, String id) | Update blog with detail entered in updateBlog.jsp page  Return: null  Input: author, title, content, imageLink, id |
| 06 | delete(int id) | Update blog with id entered in blogManage.jsp  Return: null  Input: id |

### 

### c. Sequence Diagram(s)



### 

### d. Database queries

SELECT \* FROM Blog WHERE ID = ?

## 9. Logout

### a. Class Diagram

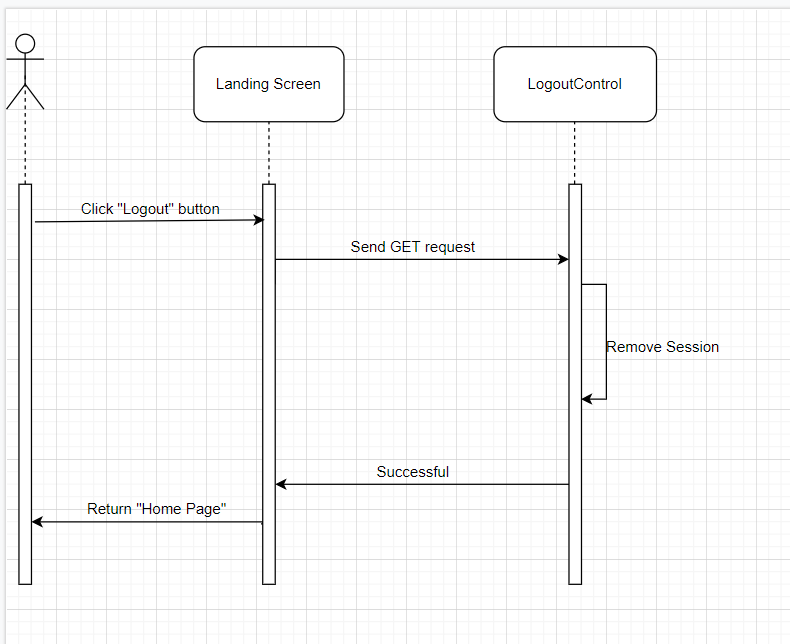
### b.Class Specifications

LogoutController

| No | Method | Description |
| --- | --- | --- |
| 01 | processRequest() | Remove attribute from session |

### 

### c. Sequence Diagram(s)

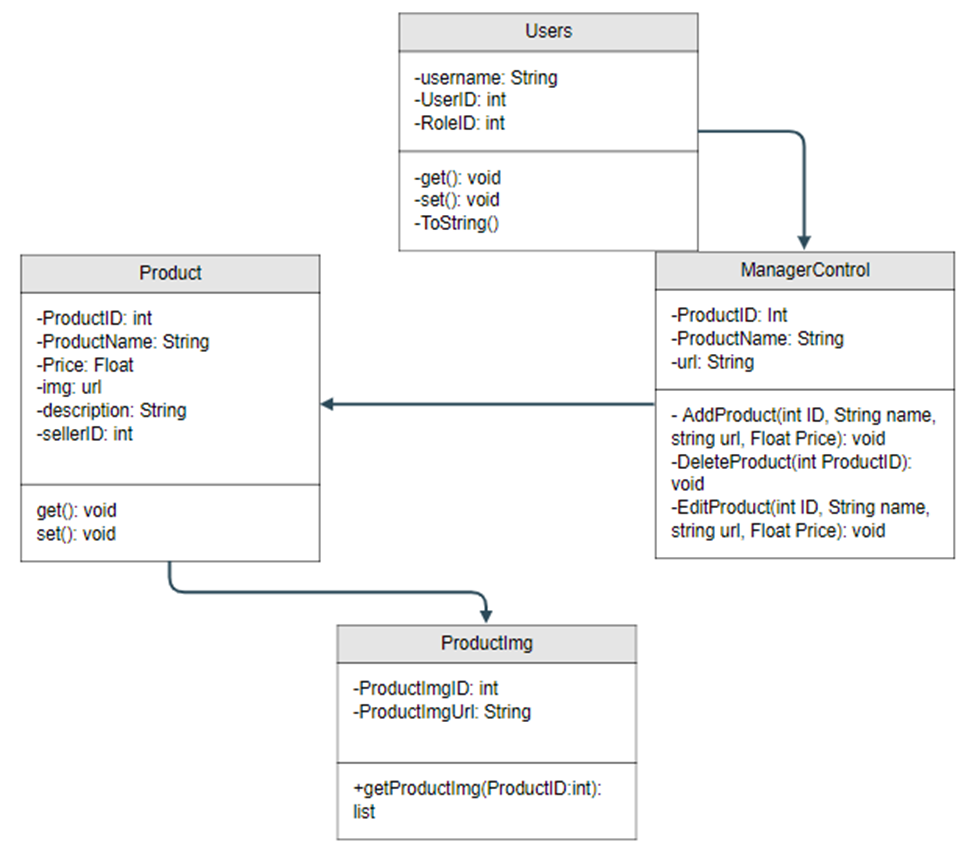


### 

### d. Database queries

## 10. Add Product

### a. Class Diagram

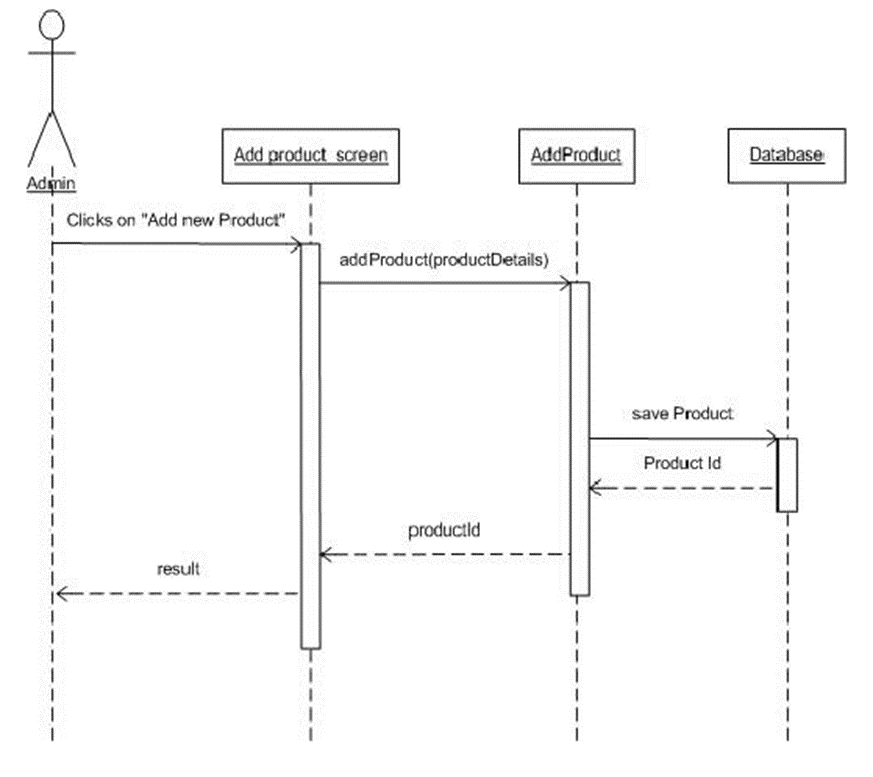


### b. Class Specifications

| No | Method | Description |
| --- | --- | --- |
| 01 | addProduct() | AddProduct from the database to display on ManageProduct.jsp |

### 

### c. Sequence Diagram(s)



## 

## 11. Change amount

### Class Diagram

### Class Specifications

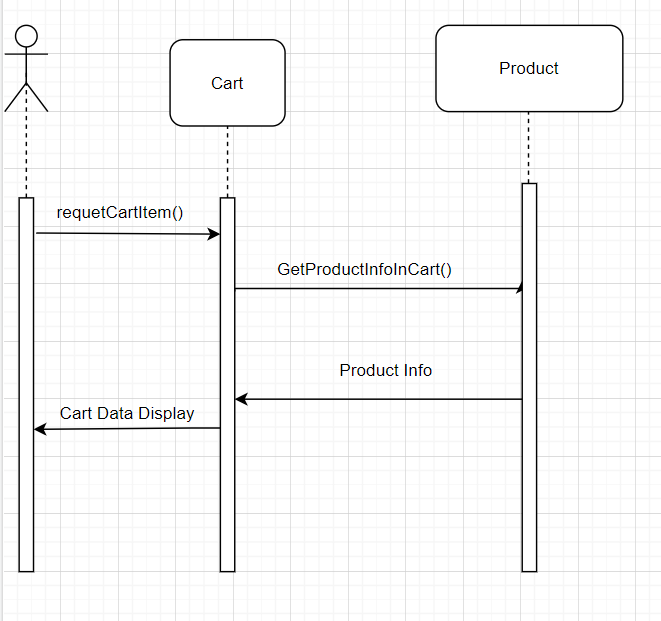
### 

UserController

| No | Method | Description |
| --- | --- | --- |
| 01 | processRequest() | Update Quantity Cart from Cart Detail |

### 

### c. Sequence Diagram(s)



### 

## 12. Delete cart

### a. Class Diagram

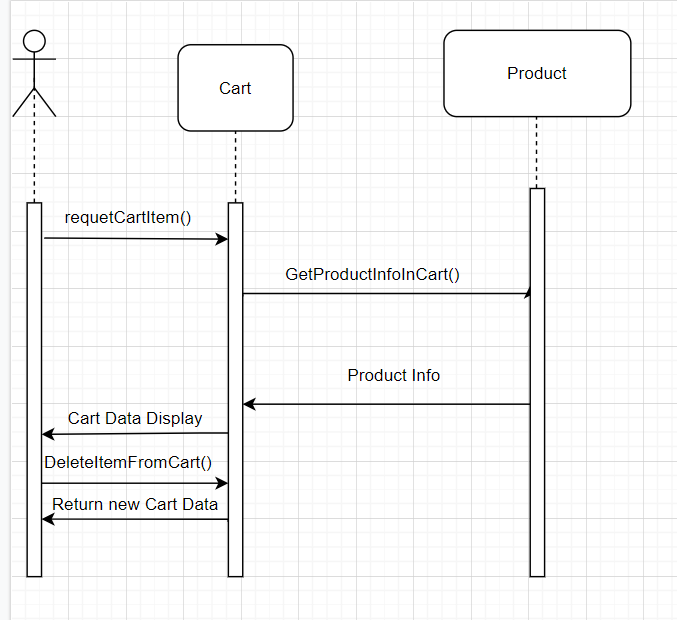
### b. Class Specifications

UserController

| No | Method | Description |
| --- | --- | --- |
| 01 | processRequest() | Delete Cart by ProductID through HashMap |

### 

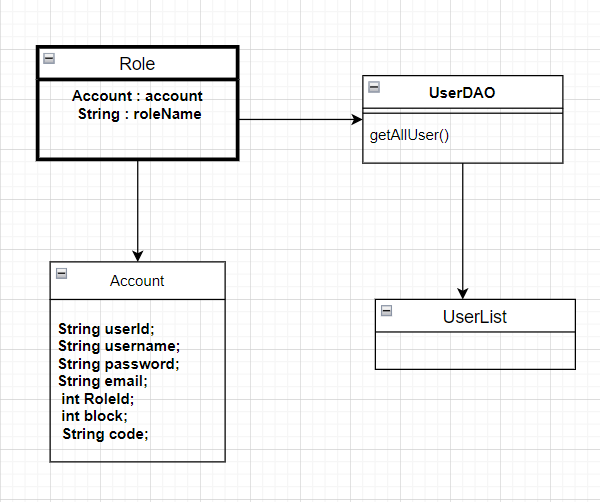
### c. Sequence Diagram(s)



### 

## 13. View List User

### a. Class Diagram



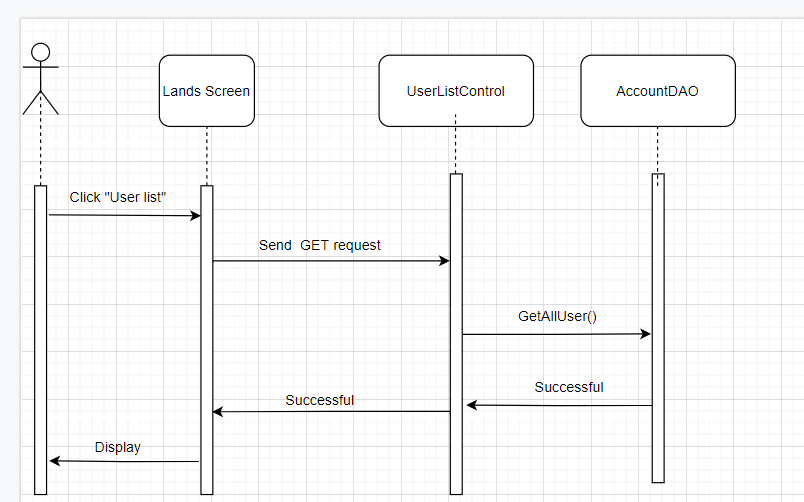
### b. Class Specifications

UserDAO

| No | Method | Description |
| --- | --- | --- |
| 01 | GetAllUser() | Get All User from Database |

### 

### c. Sequence Diagram(s)



### 

### d. Database queries

select u.\*,r.RoleName from Users u inner join role r on u.RoleID = r.RoleID

## 14. Edit Information User

### a. Class Diagram

### b. Class Specifications

UserController

| No | Method | Description |
| --- | --- | --- |
| 01 | processRequest() |  |

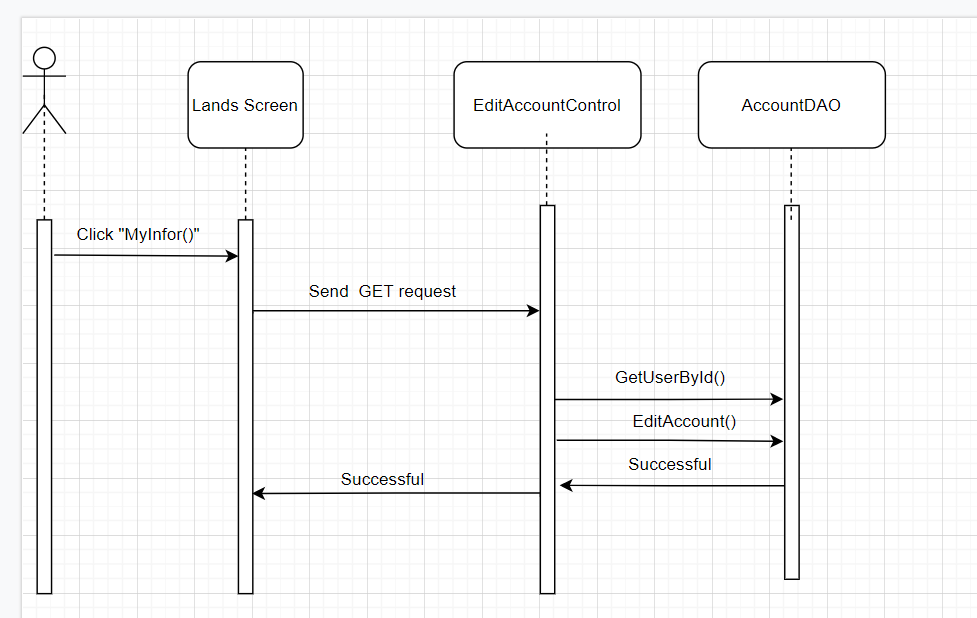
BlogDAO

[Provide the detailed description for the class methods]

| No | Method | Description |
| --- | --- | --- |
| 01 | Create User | Insert new user to database |
| 02 | Validate user | Check user if exists |

### 

### c. Sequence Diagram(s)



### 

### d. Database queries

Select \* from Users

## 15.Set Role User

### a. Class Diagram

### b. Class Specifications

UserController

| No | Method | Description |
| --- | --- | --- |
| 01 | processRequest() |  |

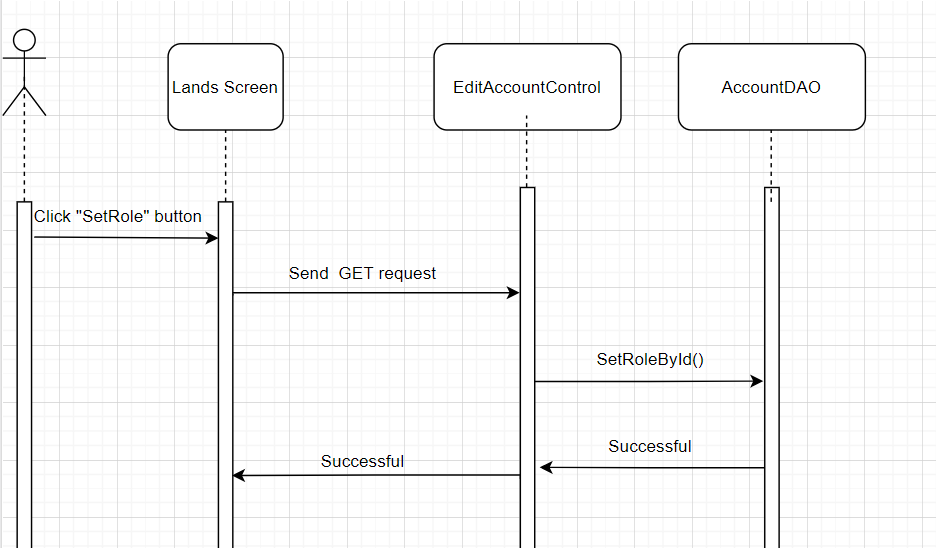
BlogDAO

[Provide the detailed description for the class methods]

| No | Method | Description |
| --- | --- | --- |
| 01 | Create User | Insert new user to database |
| 02 | Validate user | Check user if exists |

### 

### c. Sequence Diagram(s)



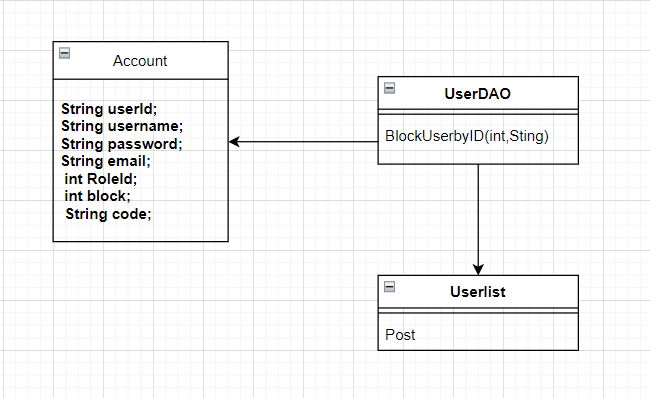
### 

### d. Database queries

Select \* from Users

## 16. Block User

### a. Class Diagram



### b. Class Specifications

Userlist

| No | Method | Description |
| --- | --- | --- |
| 01 | Post() | get UserID and change isBlock on Users |

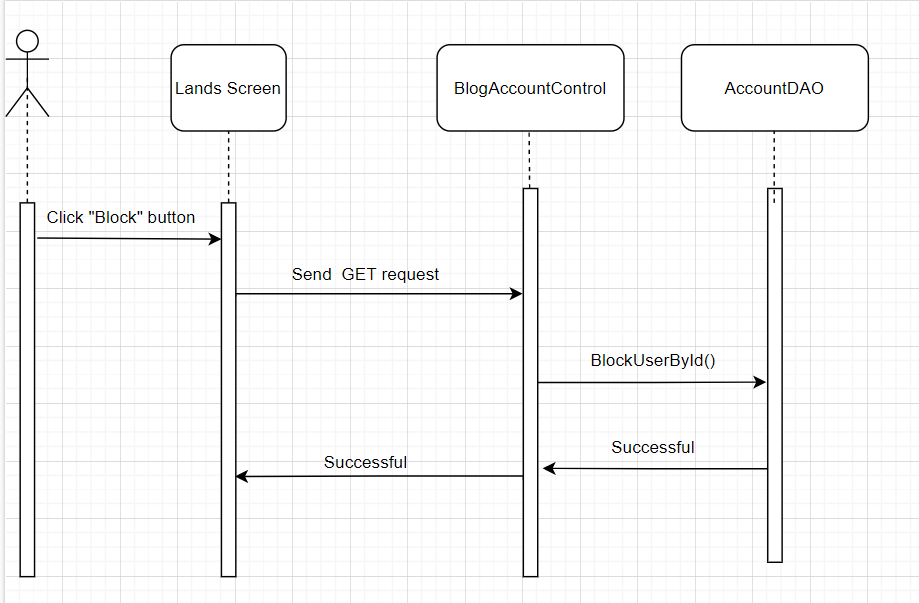
UserDAO

[Provide the detailed description for the class methods]

| No | Method | Description |
| --- | --- | --- |
| 01 | BlockUserbyID | Change isBlock user on database |

### 

### c. Sequence Diagram(s)



### 

### d. Database queries

Update users

set isBlock = ?

where UserID = ?

## 17. Reset Password

### Class Diagram

### Class Specifications

### 

UserController

| No | Method | Description |
| --- | --- | --- |
| 01 | processRequest() |  |

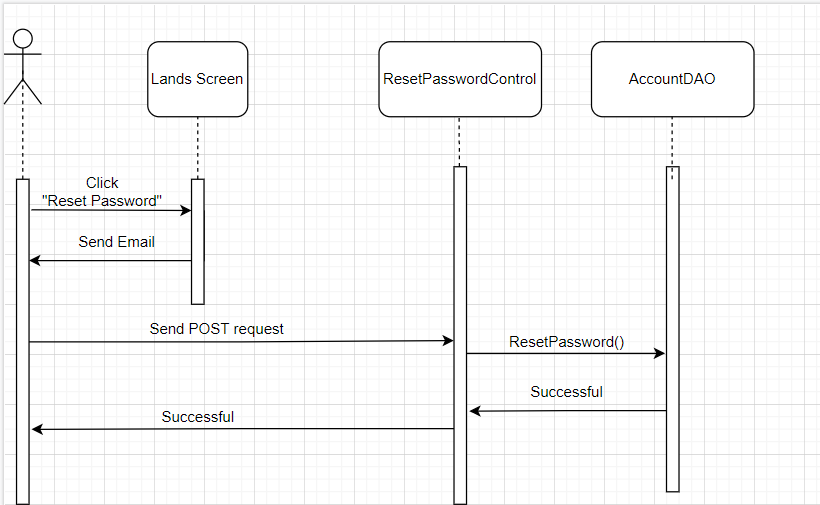
BlogDAO

[Provide the detailed description for the class methods]

| No | Method | Description |
| --- | --- | --- |
| 01 | Create User | Insert new user to database |
| 02 | Validate user | Check user if exists |

### 

### c. Sequence Diagram(s)



### 

### d. Database queries

Select \* from Users

## 18. Change Password

### a. Class Diagram

### b. Class Specifications

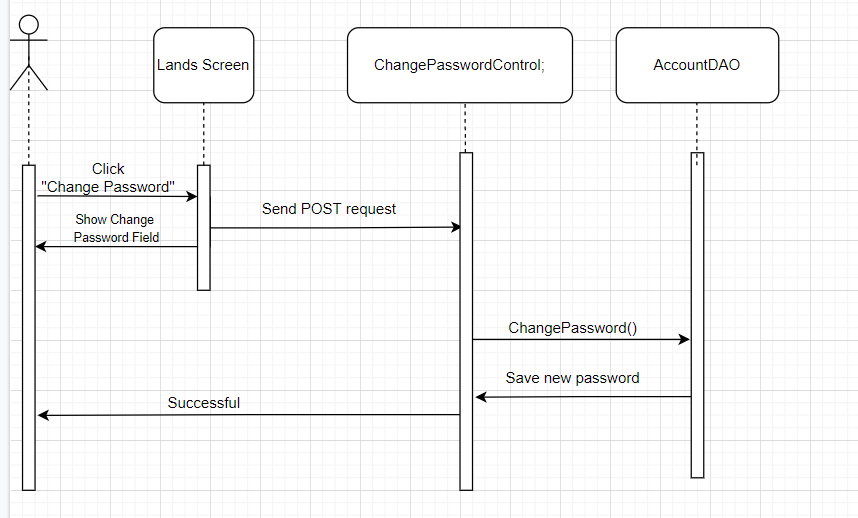
UserController

| No | Method | Description |
| --- | --- | --- |
| 01 | processRequest() |  |

| No | Method | Description |
| --- | --- | --- |
| 01 | Create User | Insert new user to database |
| 02 | Validate user | Check user if exists |

### 

### c. Sequence Diagram(s)



### 

### d. Database queries

## 19. User profile

### a. Class Diagram

### b. Class Specifications

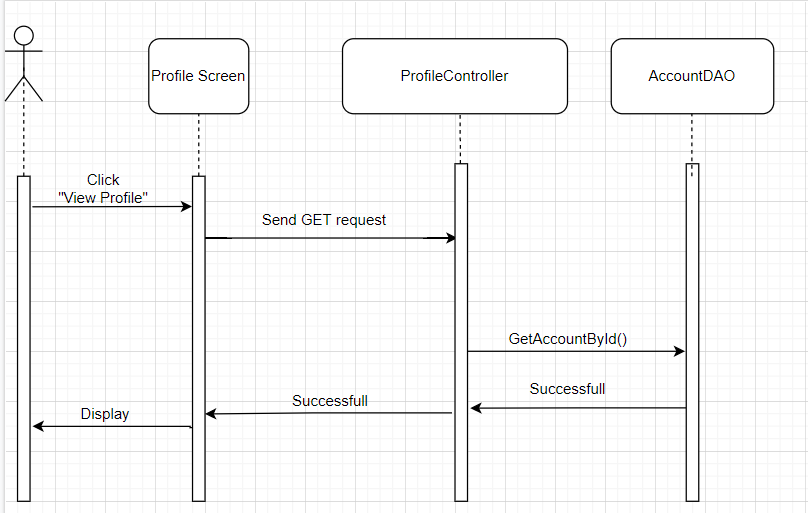
UserController

| No | Method | Description |
| --- | --- | --- |
| 01 | processRequest() |  |

| No | Method | Description |
| --- | --- | --- |
| 01 | Create User | Insert new user to database |
| 02 | Validate user | Check user if exists |

### 

### c. Sequence Diagram(s)



### 

### d. Database queries

Select \* from Users

## 20. Order History

### a. Class Diagram

### b. Class Specifications

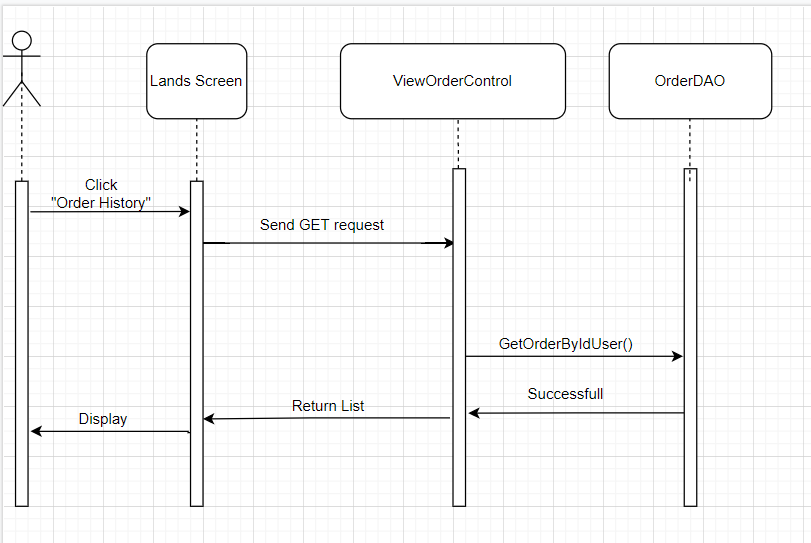
UserController

| No | Method | Description |
| --- | --- | --- |
| 01 | processRequest() |  |

| No | Method | Description |
| --- | --- | --- |
| 01 | Create User | Insert new user to database |
| 02 | Validate user | Check user if exists |

### 

### c. Sequence Diagram(s)



### 

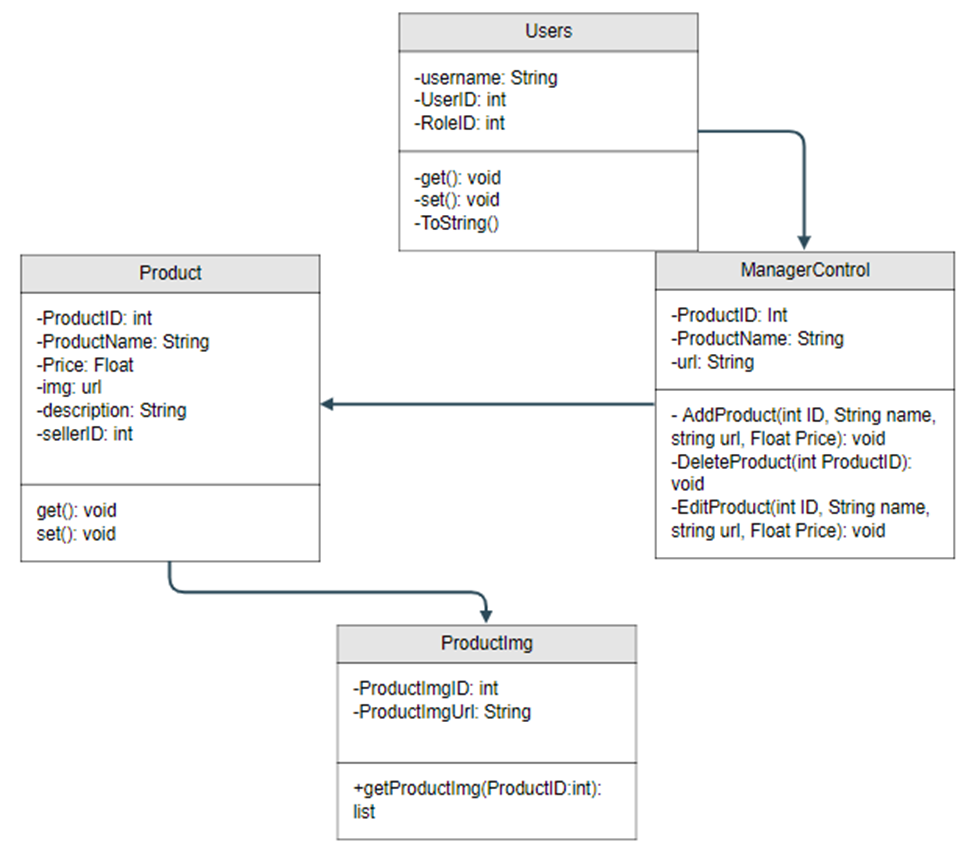
### d. Database queries

Select \* from Users

,

## 21. Delete Product

### a. Class Diagram



### b. Class Specifications

ProductDAO

| No | Method | Description |
| --- | --- | --- |
| 01 | deleteProduct() | *DeleteProduct from the database to display on ManageProduct.jsp* |

### 

### c. Sequence Diagram(s)

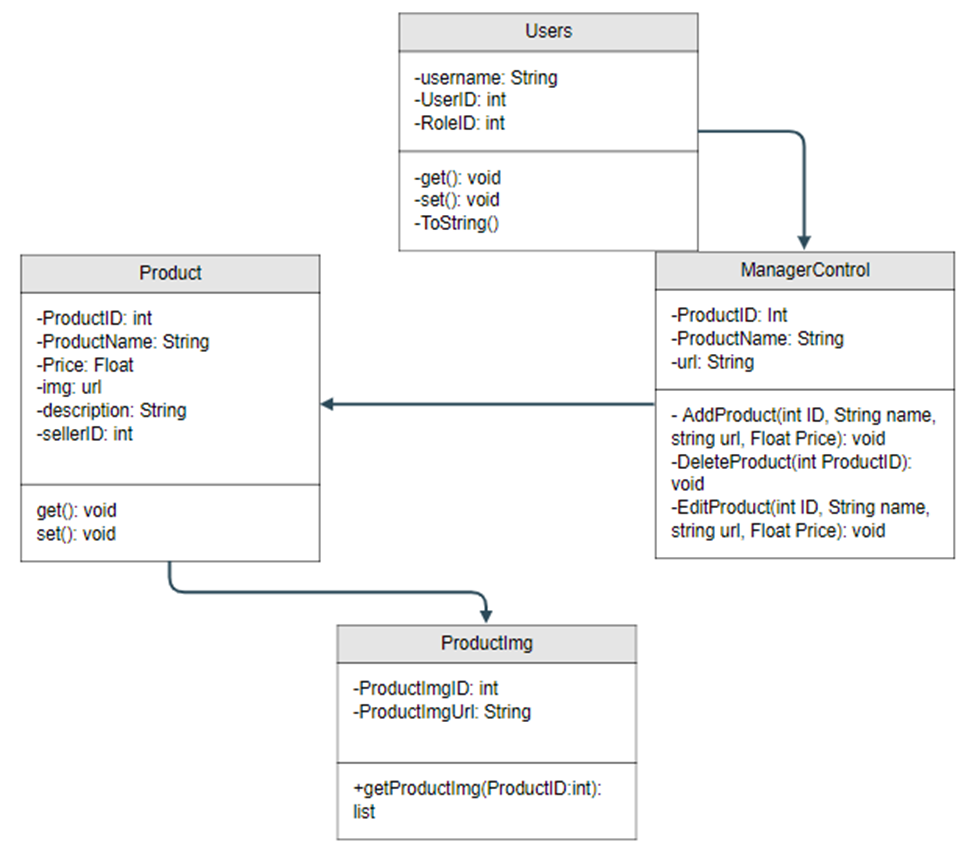
### 

### d. Database queries

delete from Product where ProductID = ?

## 22. Edit Product

### a. Class Diagram



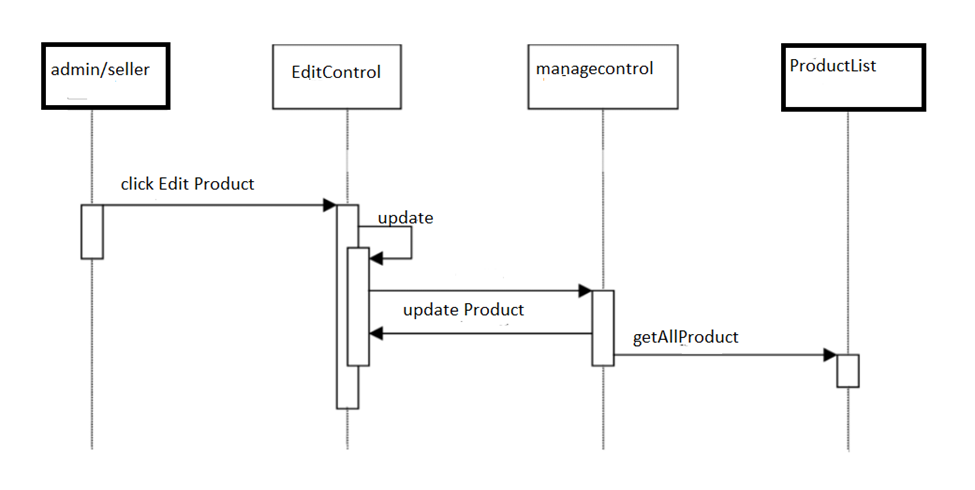
### b. Class Specifications

ProductDAO

| No | Method | Description |
| --- | --- | --- |
| 01 | editProduct() | *Update from the database to display on ManageProduct.jsp* |

### 

### c. Sequence Diagram(s)



### 

### d. Database queries

UPDATE datatable

SET ... = ?

WHERE …... = ?

## 23. Checkout

### a. Class Diagram

### b. Class Specifications

| No | Method | Description |
| --- | --- | --- |
| 01 | getProductById() | *Get Product which confirm to buy* |
| 02 | saveOrder() | *Insert information in Order table* |
| 03 | SaveOderDetail() | *Insert information in OrderDetail table* |
| 04 | SaveShipping() | *Insert information in Shipping table* |

### 

### c. Sequence Diagram(s)

### 

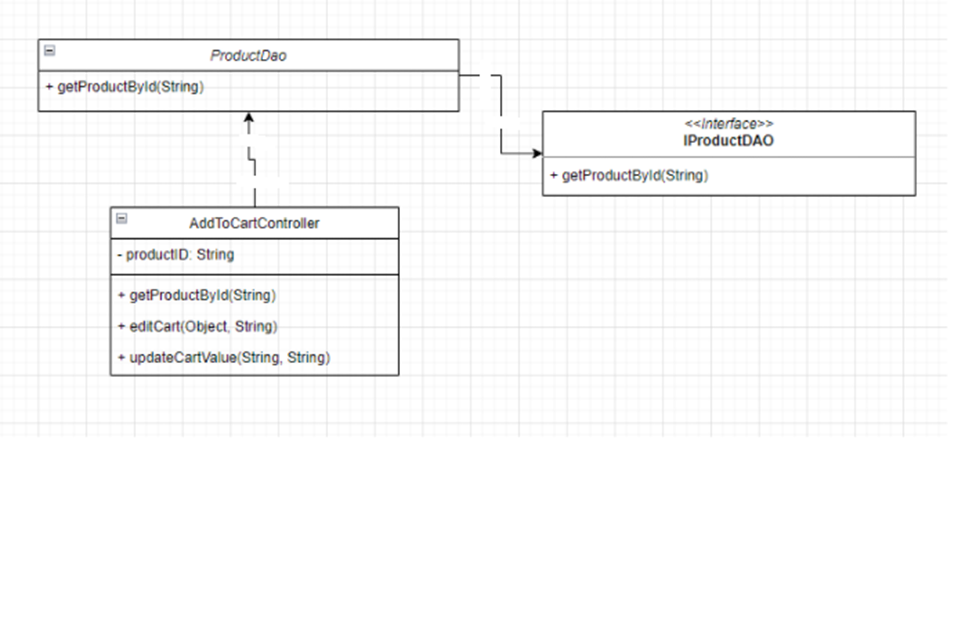
### d. Database queries

INSERT INTO [dbo].[Orders]

([UserID]

,[TotalPrice]

,[Note]

 ,[Status]

,[Date]

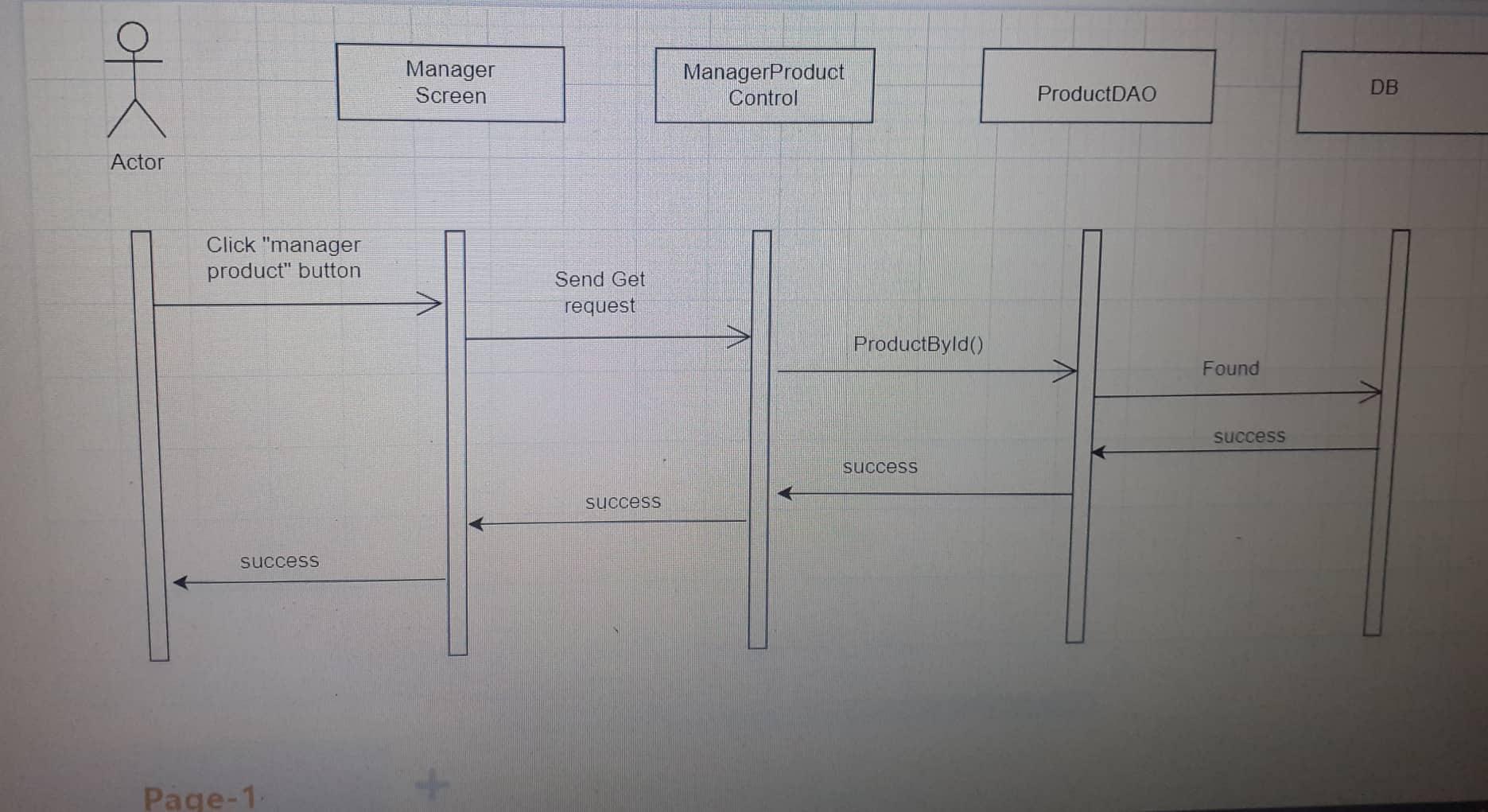
,[ShippingID])

VALUES

(?,?,?,?,GETDATE(),?)

## 24.Manager Product

1. Class diagram
2. Class specifications
3. Sequence Diagram(s)



1. Database queries

select p.\*,pr.ProductImgURL from Product p join ProductImg pr

+ "on p.ProductID = pr.ProductID

## 25.HomePage

1. Class Diagram
2. Class Specifications

ProductDAO

| No | Method | Description |
| --- | --- | --- |
| 01  02 | getAll  pagingProductBy6 | *Get all product*  *product pagination 6 products 1 page* |

SubCategoryDAO

| No | Method | Description |
| --- | --- | --- |
| 01 | getAllSubCategory | *Get all SubCategory* |

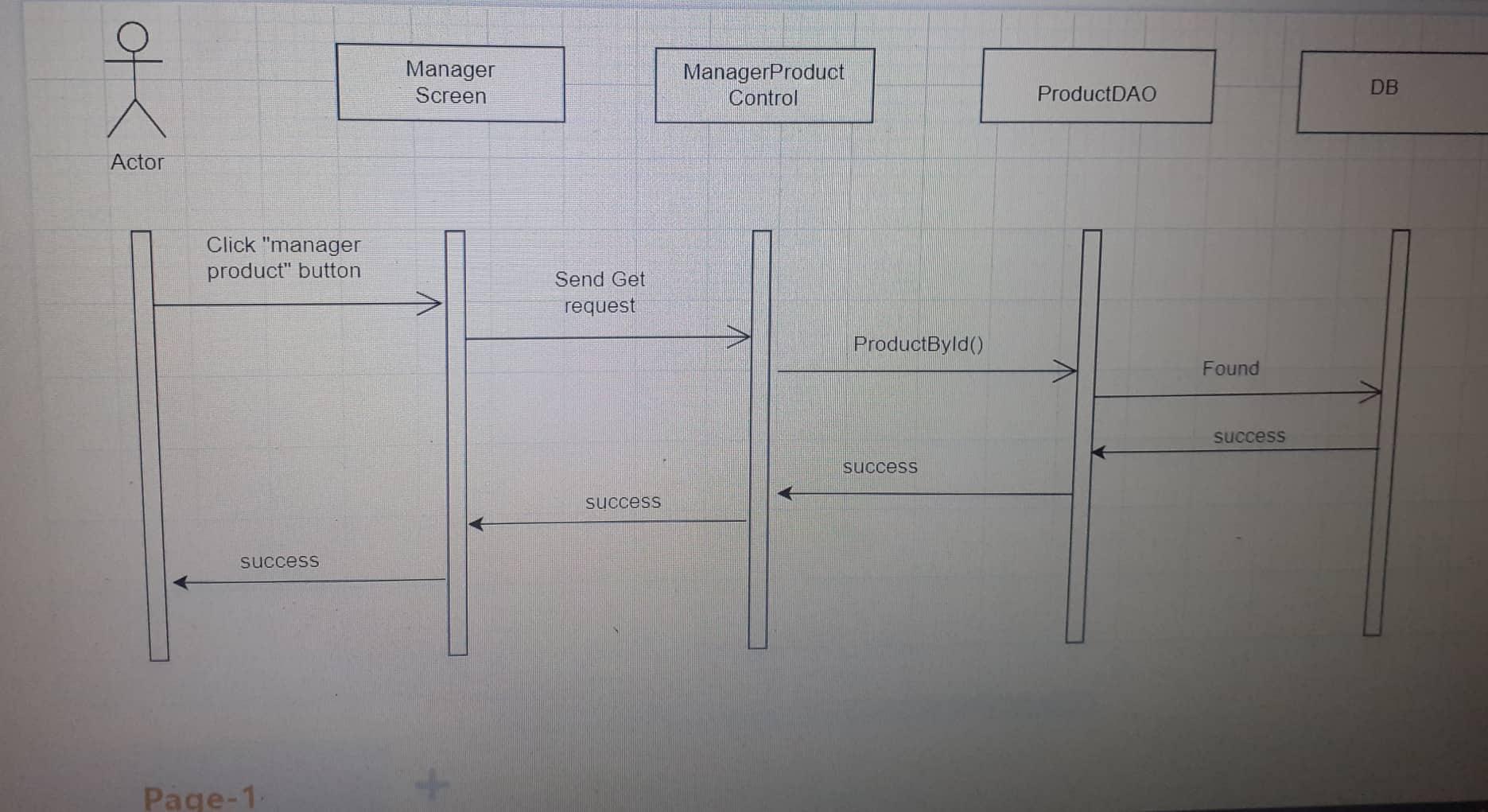
CategoryDAO

| No | Method | Description |
| --- | --- | --- |
| 01 | getAllCategory | *Get all Category* |

BrandDAO

| No | Method | Description |
| --- | --- | --- |
| 01 | getAllBrand | *Get all brand* |

1. Sequence Diagram



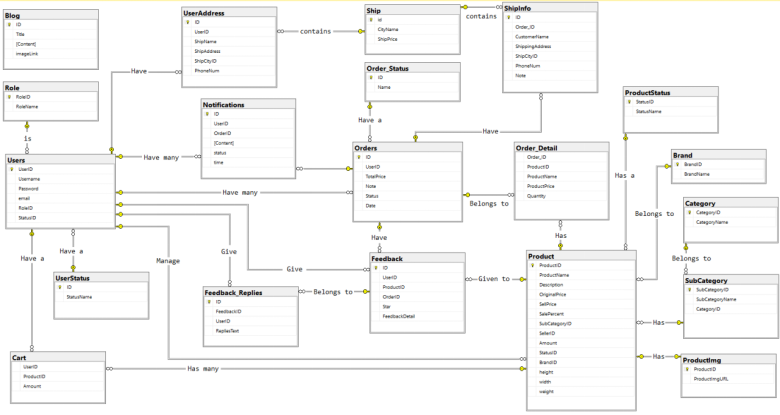
1. Database queries

select p.\*,pr.ProductImgURL from Product p join ProductImg pr

+ "on p.ProductID = pr.ProductID

# III. Database Design

## 1. Database Schema

**

## 2. Table Description

| **No** | **Table** | **Description** |
| --- | --- | --- |
| *01* | *Blog* | *Blog content + image*  *- Primary keys: ID*  *- Foreign keys:* |
| *02* | *Role* | *Role of a user ( Admin,Marketing,Sale,Customer )*  *- Primary keys: RoleID*  *- Foreign keys:* |
| *03* | *UserStatus* | *Account activation status*  *- Primary keys: ID*  *- Foreign keys:* |
| *04* | *Ship* | *Shipping price for each city*  *- Primary keys: id*  *- Foreign keys:* |
| *05* | *Order\_Detail* | *Detail of each product in order*  *- Primary keys:*  *- Foreign keys: OrderID, ProductID* |
| *06* | *UserAddress* | *Address for each user*  *- Primary keys:ID*  *- Foreign keys: UserID, ShipCityID* |
| *07* | *Users* | *Users*  *- Primary keys: UserID*  *- Foreign keys: RoleID,StatusID* |
| *08* | *Notification* | *Notification data*  *- Primary keys: ID*  *- Foreign keys: OrderID, UserID* |
| *09* | *ShipInfo* | *Shipping info for each order*  *- Primary keys: ID*  *- Foreign keys: Order\_ID, ShipCityID* |
| *10* | *Order* | *Order data*  *- Primary keys: ID*  *- Foreign keys: UserID,Status* |
| *11* | *Feedback* | *Feedback of a product order*  *- Primary keys: ID*  *- Foreign keys: UserID,ProductID, OrderID* |
| *12* | *Feedback\_Replies* | *Replies of feedbacks*  *- Primary keys: id*  *- Foreign keys: FeedbackID,UserID* |
| *13* | *Order\_Status* | *Status of a order*  *- Primary keys: ID*  *- Foreign keys:* |
| *14* | *Product* | *Product Data*  *- Primary keys: ProductID*  *- Foreign keys: SubCategoryID,SellerID,StatusID,BrandID* |
| *15* | *Category* | *Category Data*  *- Primary keys: CategoryID*  *- Foreign keys:* |
| *16* | *SubCategory* | *Sub Category ID*  *- Primary keys: SubCategoryID*  *- Foreign keys: CategoryID* |
| *17* | *Productimg* | *Image of products*  *- Primary keys:*  *- Foreign keys: ProductID* |
| *18* | *ProductStatus* | *Status of a product*  *- Primary keys: StatusID*  *- Foreign keys: StatusName* |
| *19* | *Brand* | *Brand data*  *- Primary keys: BrandID*  *- Foreign keys:* |
| *20* | *Cart* | *Cart info for each user*  *- Primary keys: UserID*  *- Foreign keys: ProductID* |