

**CLOTHING SHOP PROJECT REPORT**

**Software Design Document**

– Hanoi, Summer 2022 –

**Table of Contents**

[I. Overview](#_gjdgxs) **9**

[1. Code Packages/Namespaces](#_30j0zll) 9

[2. Coding Conventions](#_1fob9te) 10

[II. Code Designs](#_3znysh7) **10**

[1. Product List](#_2et92p0) 11

[a. Class Diagram](#_tyjcwt) 11

[b. Class Specifications](#_3dy6vkm) 11

[c. Sequence Diagram(s)](#_1t3h5sf) 12

[d. Database queries](#_4d34og8) 12

[2. Login](#_2s8eyo1) 13

[Class Diagram](#_17dp8vu) 13

[Class Specifications](#_3rdcrjn) 13

[Sequence Diagram(s)](#_26in1rg) 14

[Database queries](#_lnxbz9) 14

[4. Product detail](#_1ksv4uv) 16

[Class Diagram](#_44sinio) 16

[Class Specifications](#_2jxsxqh) 16

[c. Sequence Diagram(s)](#_3j2qqm3) 17

[d. Database queries](#_4i7ojhp) 17

[5. Register](#_2xcytpi) 18

[a. Class Diagram](#_1ci93xb) 18

[b. Class Specifications](#_3whwml4) 18

[c. Sequence Diagram(s)](#_qsh70q) 19

[d. Database queries](#_1pxezwc) 20

[6. Add to cart](#_49x2ik5) 20

[Class Diagram](#_2p2csry) 21

[Class Specifications](#_147n2zr) 21

[c. Sequence Diagram(s)](#_ihv636) 22

[d. Database queries](#_1hmsyys) 22

[Blog List](#_41mghml) 22

[Class Diagram](#_2grqrue) 23

[Class Specifications](#_vx1227) 23

[c. Sequence Diagram(s)](#_1v1yuxt) 24

[d. Database queries](#_2u6wntf) 24

[8. Blog Detail](#_19c6y18) 24

[Class Diagram](#_3tbugp1) 25

[Class Specifications](#_28h4qwu) 25

[c. Sequence Diagram(s)](#_37m2jsg) 26

[d. Database queries](#_46r0co2) 26

[9. Logout](#_2lwamvv) 26

[a. Class Diagram](#_111kx3o) 26

[b.Class Specifications](#_3l18frh) 27

[c. Sequence Diagram(s)](#_4k668n3) 27

[d. Database queries](#_1egqt2p) 28

[10. Add Product](#_3ygebqi) 29

[a. Class Diagram](#_2dlolyb) 29

[b. Class Specifications](#_sqyw64) 29

[c. Sequence Diagram(s)](#_4bvk7pj) 30

[11. Change amount](#_1664s55) 30

[a. Class Diagram](#_3q5sasy) 31

[b. Class Specifications](#_25b2l0r) 31

[c. Sequence Diagram(s)](#_kgcv8k) 32

[d. Database queries](#_34g0dwd) 32

[12. View List User](#_4h042r0) 32

[a. Class Diagram](#_2w5ecyt) 33

[b. Class Specifications](#_1baon6m) 33

[c. Sequence Diagram(s)](#_2afmg28) 34

[d. Database queries](#_39kk8xu) 34

[14. Edit Information User](#_1opuj5n) 35

[a. Class Diagram](#_48pi1tg) 35

[b. Class Specifications](#_2nusc19) 35

[c. Sequence Diagram(s)](#_3mzq4wv) 36

[d. Database queries](#_haapch) 36

[15.Set Role User](#_319y80a) 36

[a. Class Diagram](#_1gf8i83) 37

[b. Class Specifications](#_40ew0vw) 37

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

[d. Database queries](#_1tuee74) 38

[16. Block User](#_4du1wux) 38

[a. Class Diagram](#_2szc72q) 39

[b. Class Specifications](#_184mhaj) 39

[c. Sequence Diagram(s)](#_279ka65) 40

[d. Database queries](#_36ei31r) 40

[17. Reset Password](#_1ljsd9k) 41

[Class Diagram](#_45jfvxd) 41

[Class Specifications](#_2koq656) 41

[c. Sequence Diagram(s)](#_1yyy98l) 42

[d. Database queries](#_2y3w247) 42

[18. Change Password](#_1d96cc0) 43

[a. Class Diagram](#_3x8tuzt) 43

[b. Class Specifications](#_2ce457m) 43

[c. Sequence Diagram(s)](#_3bj1y38) 44

[d. Database queries](#_4anzqyu) 44

[19. User profile](#_2pta16n) 44

[a. Class Diagram](#_14ykbeg) 44

[b. Class Specifications](#_3oy7u29) 45

[c. Sequence Diagram(s)](#_j8sehv) 45

[d. Database queries](#_1idq7dh) 46

[20. Order History](#_42ddq1a) 46

[a. Class Diagram](#_2hio093) 47

[b. Class Specifications](#_wnyagw) 47

[c. Sequence Diagram(s)](#_1vsw3ci) 48

[d. Database queries](#_2uxtw84) 48

[21. Delete Product](#_1a346fx) 49

[a. Class Diagram](#_3u2rp3q) 49

[b. Class Specifications](#_2981zbj) 49

[c. Sequence Diagram(s)](#_38czs75) 50

[d. Database queries](#_47hxl2r) 50

[22. Edit Product](#_2mn7vak) 50

[a. Class Diagram](#_11si5id) 51

[b. Class Specifications](#_3ls5o66) 51

[c. Sequence Diagram(s)](#_4kx3h1s) 52

[d. Database queries](#_1f7o1he) 52

[23. Checkout](#_3z7bk57) 53

[a. Class Diagram](#_2eclud0) 53

[b. Class Specifications](#_thw4kt) 53

[c. Sequence Diagram(s)](#_1smtxgf) 54

[d. Database queries](#_2rrrqc1) 54

[24.Manager Product](#_16x20ju) 55

[Class diagram](#_3qwpj7n) 56

[Class specifications](#_l7a3n9) 56

[Sequence Diagram(s)](#_356xmb2) 57

[Database queries](#_1kc7wiv) 57

[25.Home Page](#_44bvf6o) 57

[Class Diagram](#_2jh5peh) 57

[Class Specifications](#_1xrdshw) 57

[Sequence Diagram](#_4hr1b5p) 58

[d. Database queries](#_1c1lvlb) 59

[26. List Customer](#_3w19e94) 59

[Class Diagram](#_2b6jogx) 59

[Class Specification](#_qbtyoq) 59

[Sequence Diagram(s)](#_3abhhcj) 60

[Database queries](#_49gfa85) 60

[26. Add Ship Information](#_3sv78d1) 60

[a. Class Diagram](#_280hiku) 60

[b. Class Specifications](#_n5rssn) 60

[c. Sequence Diagram(s)](#_375fbgg) 61

[d. Database queries](#_69keh8u9n0ru) 61

[27. Feedback](#_k4vp54mclt76) 61

[a. Class Diagram](#_6zfwwcyqbgob) 62

[b. Class Specifications](#_w9ncp81ytxou) 62

[c. Sequence Diagram(s)](#_vn3wyvvxo2q0) 63

[d. Database queries](#_ulrgmbz14wjk) 63

[28. Feedback Replies](#_iwkilhp0w9iv) 63

[a. Class Diagram](#_l73p565drg8c) 64

[b. Class Specifications](#_cdyrb8aexl07) 64

[c. Sequence Diagram(s)](#_oqfkhvsz226i) 65

[d. Database queries](#_kaadar76m5lj) 65

[29. Notification](#_k8wkqq2nlltx) 65

[a. Class Diagram](#_5vcknfc6d7g6) 66

[b. Class Specifications](#_ny7krnpmk8ke) 66

[c. Sequence Diagram(s)](#_khjzc4mykhg) 67

[d. Prod. Database queries](#_t78qxydh3bu3) 67

[30. Manage Feedback](#_v51mabvacmtk) 67

[a. Class Diagram](#_gbyxcnanwv1r) 67

[b. Class Specifications](#_huvlp6jgtvc) 67

[c. Sequence Diagram(s)](#_2ns983ovk5sk) 68

[d. Database queries](#_pryz3uygum17) 68

[31. Management Blog](#_nmst8ffegj3l) 68

[a. Class Diagram](#_y6tfrkn4wfa5) 68

[b. Class Specifications](#_22zeyko1i8va) 68

[c. Sequence Diagram(s)](#_l70t218ejqln) 70

[d. Database queries](#_f3v4dng4ouh3) 70

[32. Manager Brand](#_9anj7fm14u1g) 70

[a. Class Diagram](#_mpow4qn8r5td) 70

[b. Class Specifications](#_ta2wacmuo9oq) 70

[c. Sequence Diagram(s)](#_ie0seofb9r4k) 72

[d. Database queries](#_3bzyz5mcvggd) 72

[33.Edit Brand](#_uvppmzkv0ew4) 72

[a. Class Diagram](#_bw1ttl20z0k5) 73

[b. Class Specifications](#_s66zskmp91ku) 73

[c. Sequence Diagram(s)](#_3fpftganrmpb) 74

[d. Database queries](#_bgq97mwmsf59) 74

[34. Add Brand](#_j2a0gne0febp) 74

[a. Class Diagram](#_2x628ssk8ade) 75

[b. Class Specifications](#_gcamx5hppvy1) 75

[c. Sequence Diagram(s)](#_4p8a335x6qp) 76

[d. Database queries](#_pthry6hm5b0b) 76

[35. Admin DashBoard](#_gxu2f2kkqfyo) 76

[a. Class Diagram](#_qsmlu44jgr35) 77

[b. Class Specifications](#_solw5op9hqto) 77

[c. Sequence Diagram(s)](#_woy3sg7v42sx) 77

[d. Database queries](#_2dbkdd2hy8vj) 77

[36. Order Detail[Sale]](#_xgyh2wx7ux7x) 77

[a. Class Diagram](#_idpby1a1a63b) 77

[b. Class Specifications](#_pl8u3bi5rpjc) 77

[c. Sequence Diagram(s)](#_kvzn2tx79f2n) 78

[d. Database queries](#_h4xqu9fnn9g0) 78

[36. Cart Completion](#_513z28z5ue40) 78

[a. Class Diagram](#_mlagyndii04b) 79

[b. Class Specifications](#_krd65ua77mag) 79

[c. Sequence Diagram(s)](#_ynoyefmv82ee) 79

[d. Database queries](#_ck05jhdr6bf9) 79

[36. Comment Product](#_bfc605elrs8y) 79

[a. Class Diagram](#_spwatsu4ftvt) 79

[b. Class Specifications](#_t6ycior48c72) 79

[c. Sequence Diagram(s)](#_yh84b1m5zkkd) 80

[d. Database queries](#_b8azfpuvic8m) 80

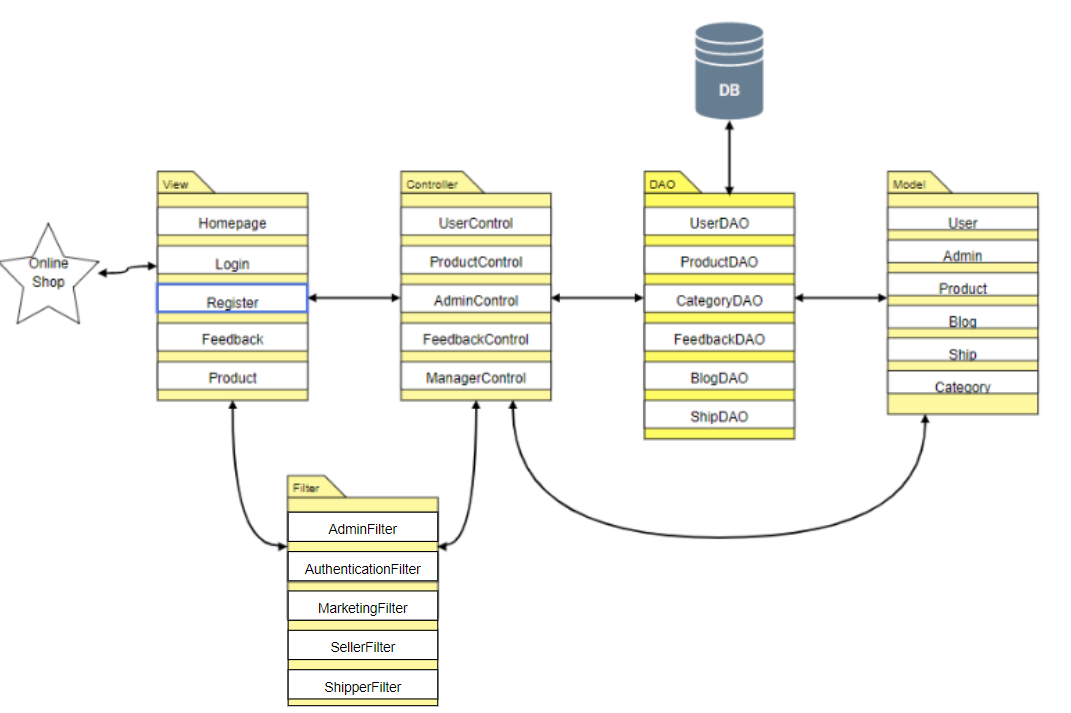
[III. Database Design](#_1maplo9) **81**

[1. Database Schema](#_46ad4c2) 81

[2. Table Description](#_2lfnejv) 81

# I. Overview

## 1. Code Packages/Namespaces



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

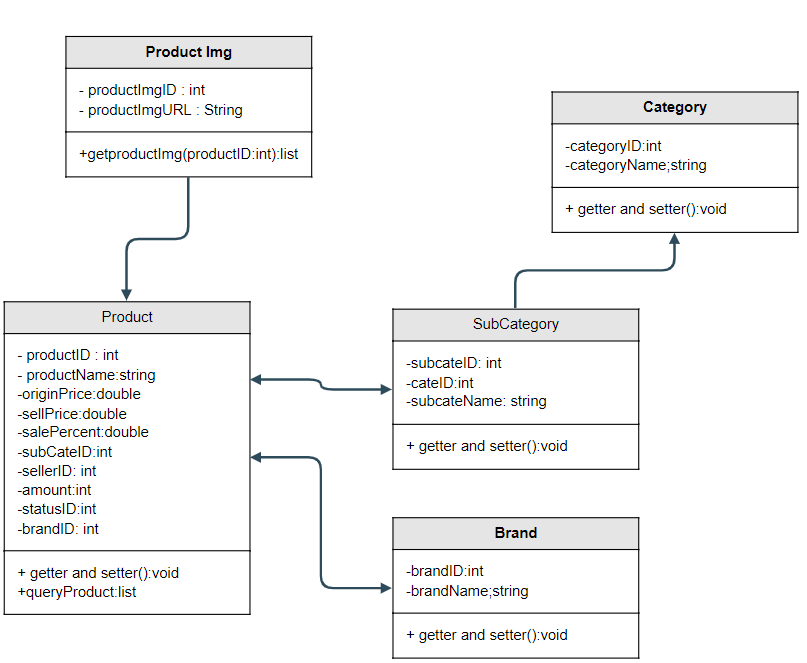
| **No** | **Package** | **Description** |
| --- | --- | --- |
| *01* | *Controllers* | This package contains all controllers for this MVC2 project and authentication or dispatcher filters  Servlet Naming convention: Starting with a verb.  Ex: LoginController, SignUpController,... |
| *02* | *Views* | 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. Product List

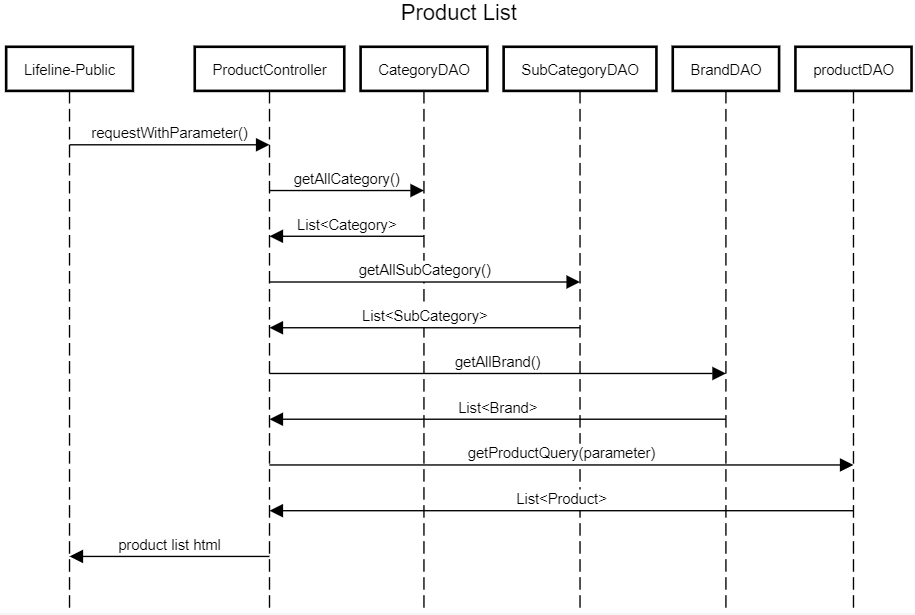
### a. Class Diagram



### b. Class Specifications

| **No** | **Method** | **Description** |
| --- | --- | --- |
| *01* | *getAllProduct()* | *Get all product from database* |
| *02* | *GetAllCategory()* | *Get all category from database* |
| *03* | *GetAllSubCategory()* | *Get all subcategory from database* |
| *04* | *GetAllBrand()* | *Get all brand from database* |

### c. Sequence Diagram(s)

**

### d. Database queries

SELECT \* FROM (SELECT p.ProductID,MIN(p.ProductName) AS ProductName,MIN(p.Description) AS Description,

MIN(p.OriginalPrice) AS OriginalPrice,MIN(p.SalePrice) AS SalePrice,

MIN(p.Sale) as SalePercent,\n"

MIN(p.SubCategoryID) AS SubCategoryID,MIN(p.Amount) AS Amount,

MIN(p.BrandID) AS BrandID,MIN(p.sell\_id) AS sell\_id,

MIN(ProI.ProductImgURL) AS ProductImgURL FROM

dbo.Product p JOIN dbo.ProductImg ProI ON ProI.ProductID = p.ProductID GROUP BY p.ProductID ) t

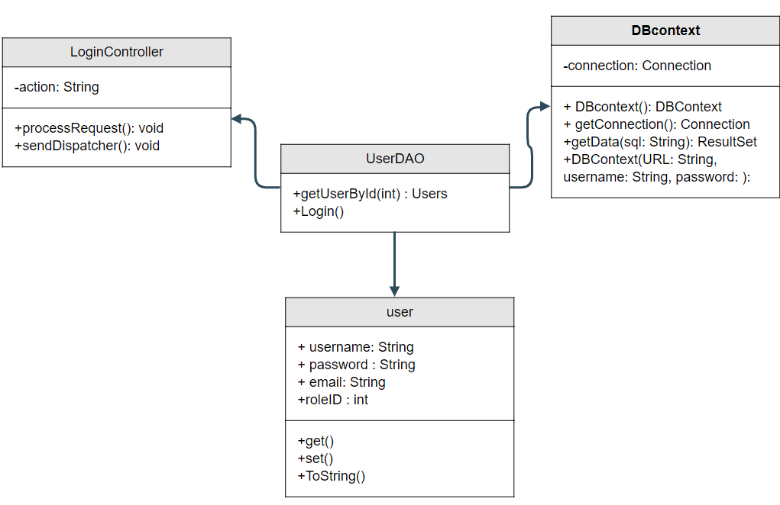
SELECT \* from Category

SELECT \* FROM dbo.SubCategory

SELECT \* FROM dbo.Brand

## 2. Login

### Class Diagram



### Class Specifications

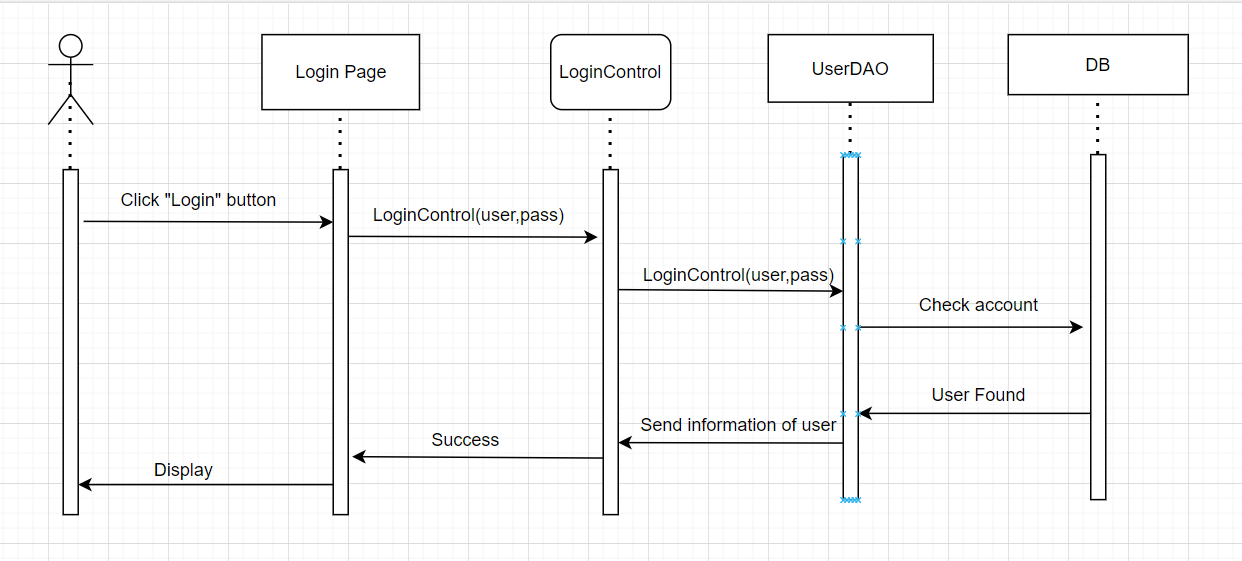
UserController

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

UserDAO

| 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)

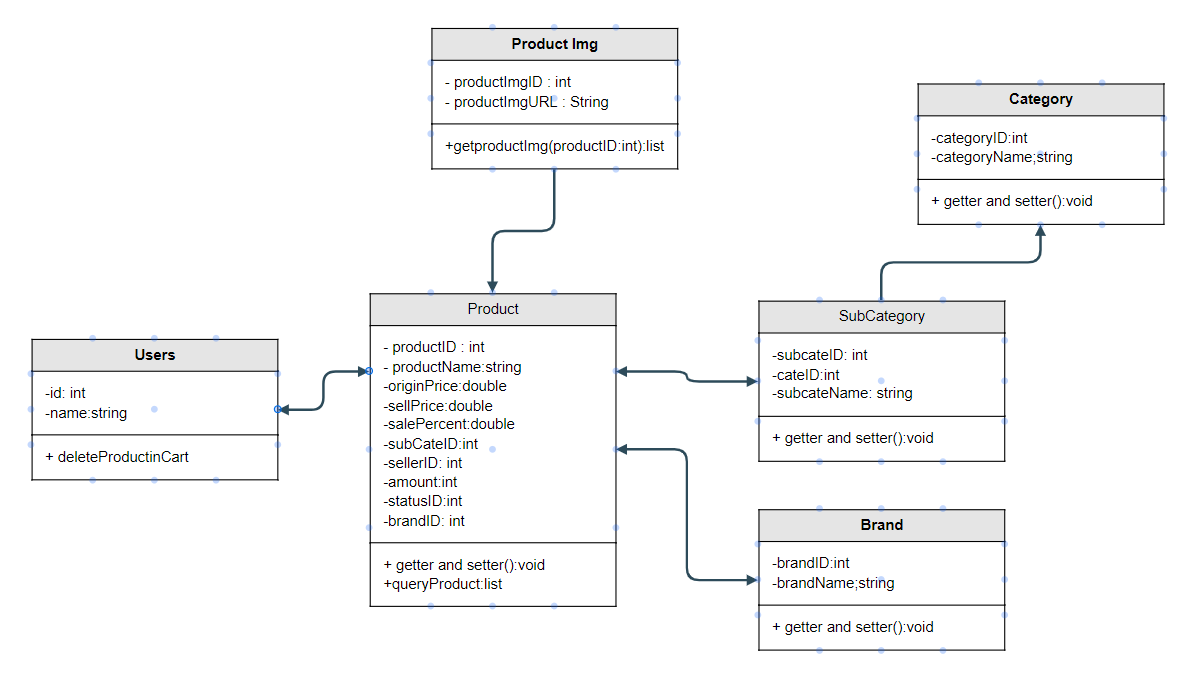


### Database queries

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

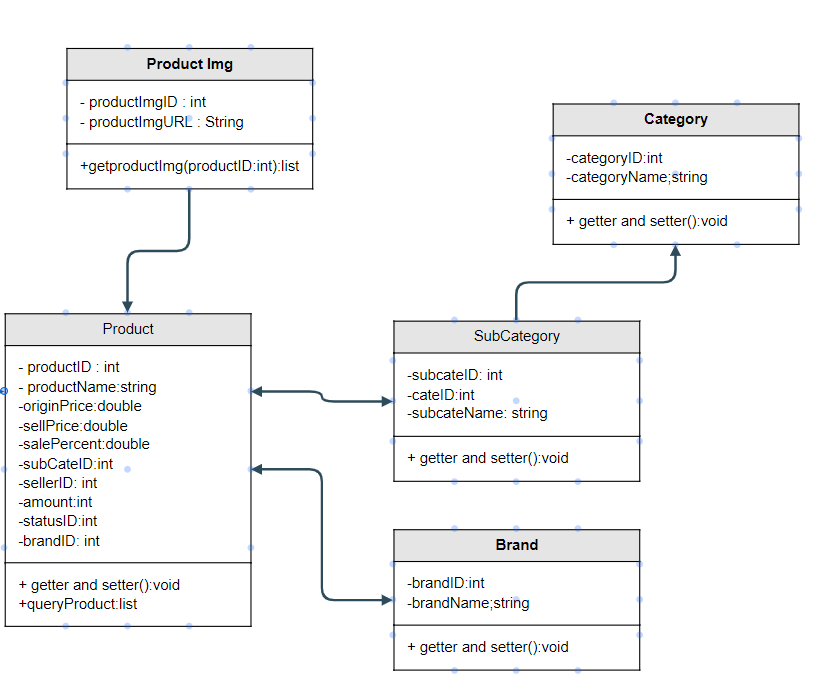
**3.Cart Detail**

1. **Class Diagram**



## 4. Product detail

### Class Diagram



### Class Specifications

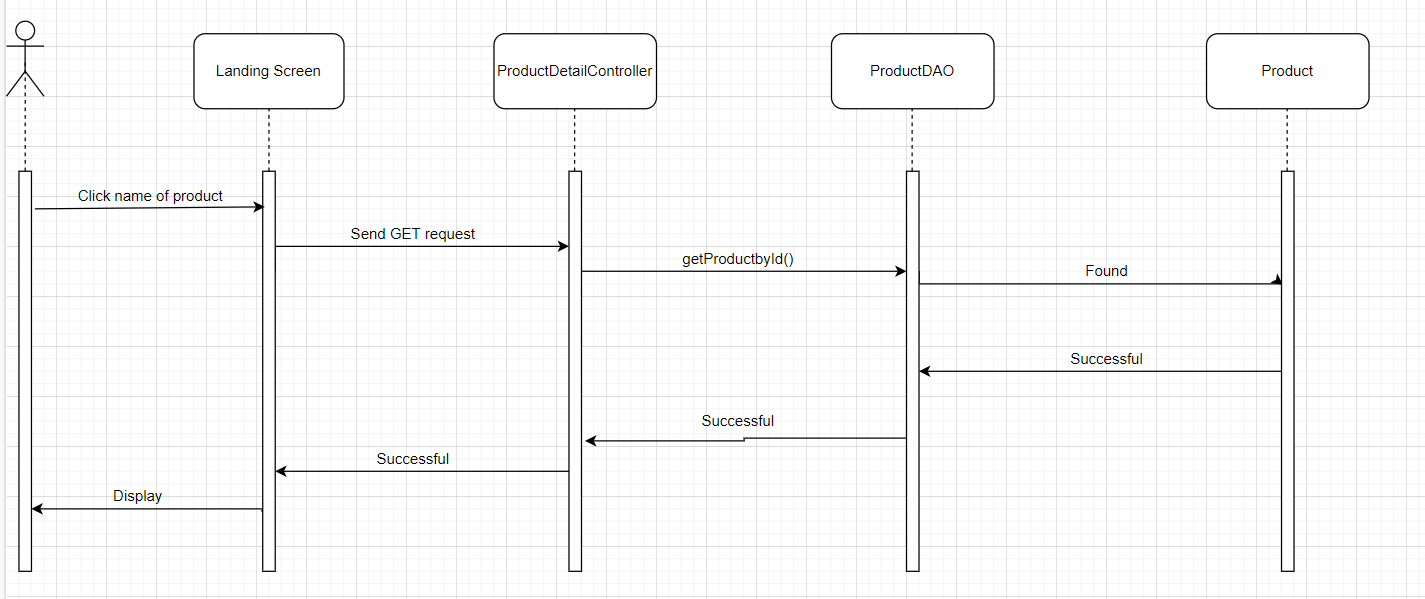
ProductDAO

| No | Method | Description |
| --- | --- | --- |
| 01 | getProductByID | get Product by ID |
| 02 | getTop4ListProByCategory | Get 4 products that same subcategory vs current product |

CommentDAO

| No | Method | Description |
| --- | --- | --- |
| 01 | insertComment | Add Comment into Product |
| 02 | getComment | View all comment by productId |

### 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"

SELECT top 4 \* FROM (SELECT p.ProductID,MIN(p.ProductName) AS ProductName,MIN(p.Description) AS Description,\n"

MIN(p.OriginalPrice) AS OriginalPrice,MIN(p.SalePrice) AS SalePrice,

MIN(p.SubCategoryID) AS SubCategoryID,MIN(p.Amount) AS Amount,

MIN(p.BrandID) AS BrandID,MIN(p.sell\_id) AS sell\_id,

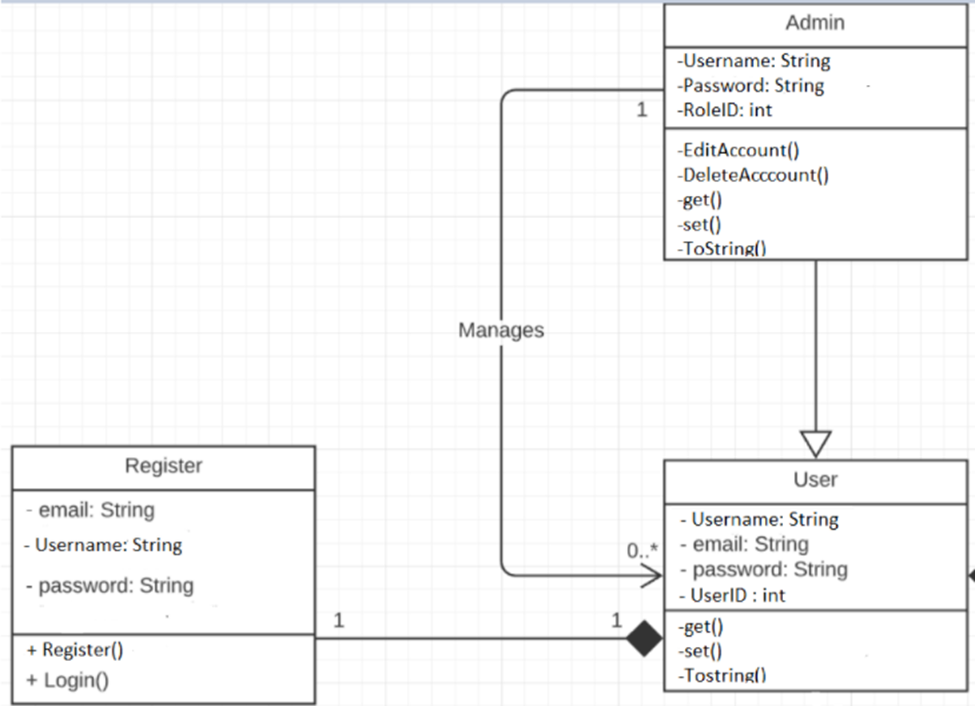
MIN(ProI.ProductImgURL) AS ProductImgURL FROM

dbo.Product p JOIN dbo.ProductImg ProI ON ProI.ProductID = p.ProductID GROUP BY p.ProductID ) t where t.SubCategoryID = ? and t.ProductID != ?

SELECT \* from comment where productid = ?

## 5. Register

### **a. Class Diagram**



### b. Class Specifications

UserController

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

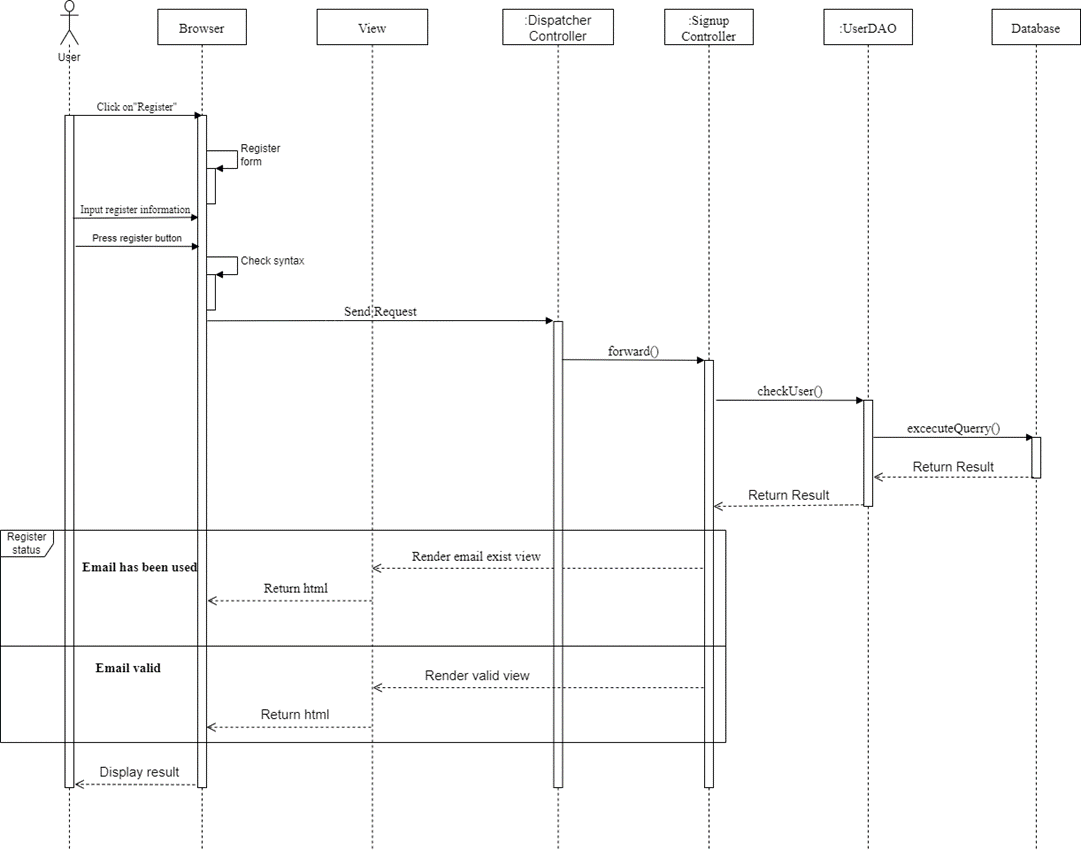
UserDAO

[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]

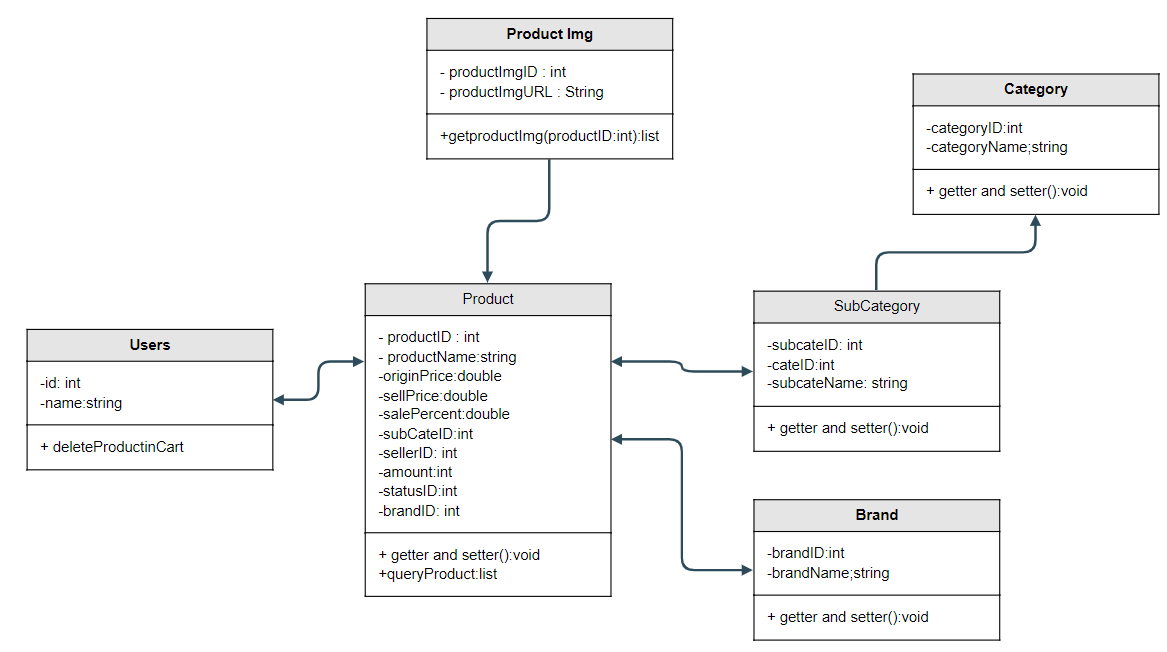
,[RoleID]

,[isBlock])

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

## 6. Add to cart

### Class Diagram



### Class Specifications

CartDetailController

| No | Method | Description |
| --- | --- | --- |
| 01 | processRequest() | If Products are already in cart then plus number quantity else add products to cart |

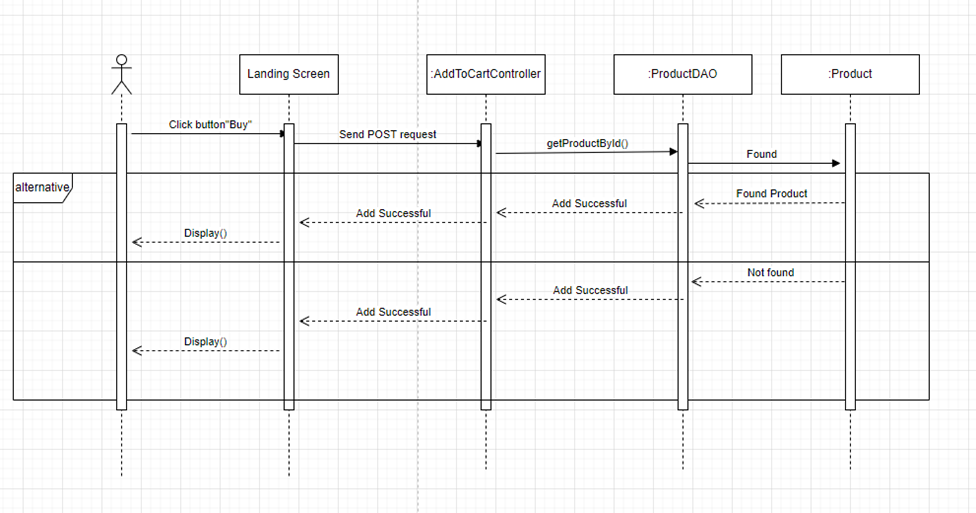
ProductDAO

[Provide the detailed description for the class methods]

| No | Method | Description |
| --- | --- | --- |
| *01* | *getProductById()* | *Get Product you want to add to cart* |

### 

### c. Sequence Diagram(s)



### d. Database queries

SELECT \* FROM (SELECT p.ProductID,MIN(p.ProductName) AS ProductName,MIN(p.Description) AS Description,

MIN(p.OriginalPrice) AS OriginalPrice,MIN(p.SalePrice) AS SalePrice,

MIN(p.SubCategoryID) AS SubCategoryID,MIN(p.Amount) AS Amount,

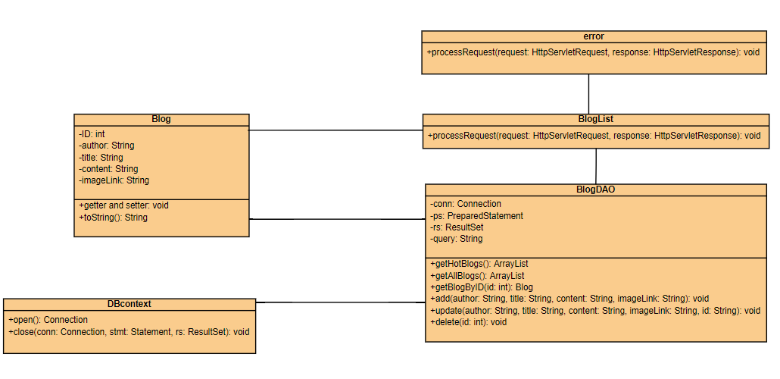
MIN(p.BrandID) AS BrandID,MIN(p.sell\_id) AS sell\_id,

MIN(ProI.ProductImgURL) AS ProductImgURL FROM

dbo.Product p JOIN dbo.ProductImg ProI ON ProI.ProductID = p.ProductID GROUP BY p.ProductID ) t where t.ProductID = ?

## Blog List

### Class Diagram



### Class Specifications

- BlogList class

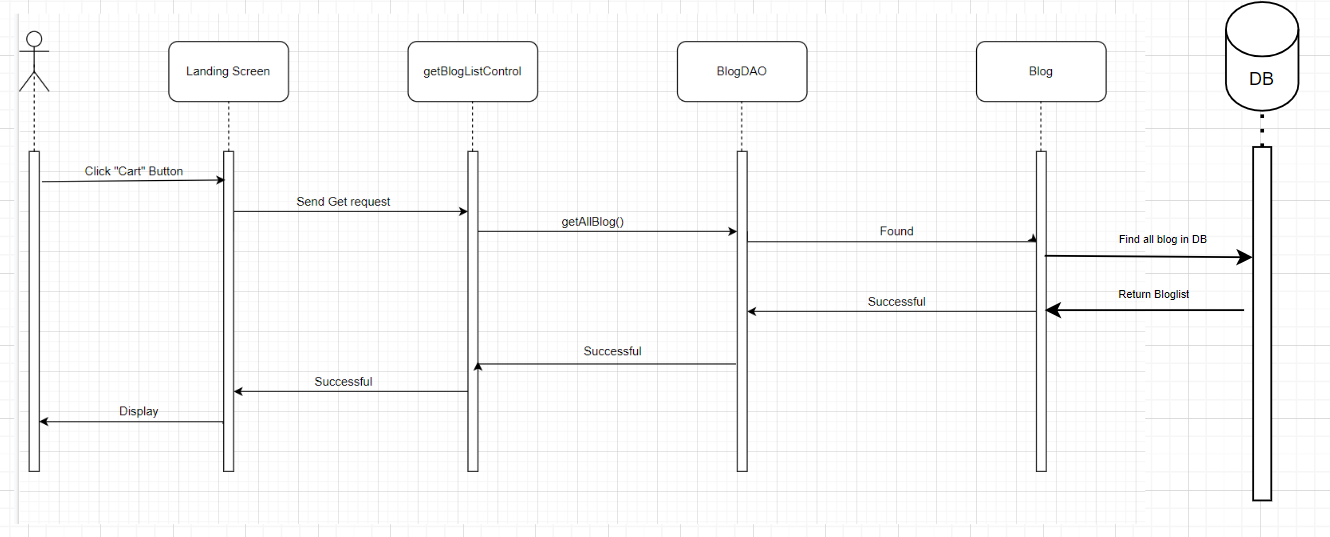
| **No** | **Method** | **Description** |
| --- | --- | --- |
| 01 | processRequest() | get blog list from the database to display on BlogList.jsp |

- BlogDAO class

| **No** | **Method** | **Description** |
| --- | --- | --- |
| 01 | getBlogByID(int id) | Return: Blog Detail where blog id contain parameter  Input: ID of blog |
| 02 | getAllBlog() | Return: All blog in database |
| 03 | 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 |
| 04 | 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 |
| 05 | delete(int id) | Delete blog with id entered in blogManage.jsp  Return: null  Input: id |

### 

### c. Sequence Diagram(s)



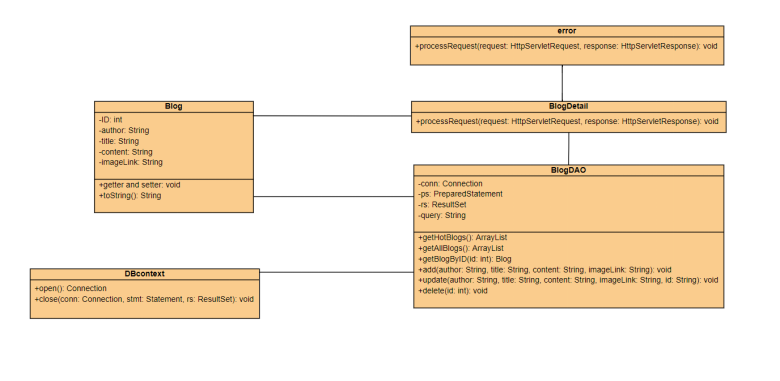
### 

### d. Database queries

**Select \* from Blog desc**

## 8. Blog Detail

### Class Diagram



### Class Specifications

- BlogDetail class

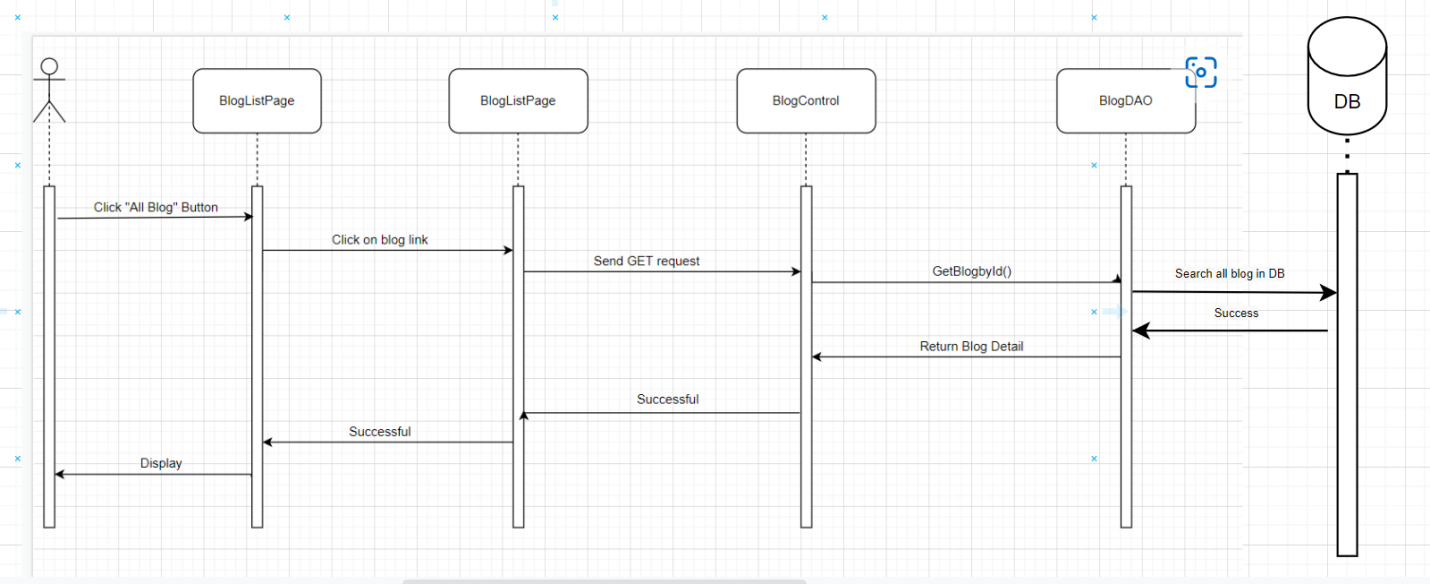
| **No** | **Method** | **Description** |
| --- | --- | --- |
| 01 | processRequest() | get blog detail from the database to display on BlogDetail.jsp |

- BlogDAO class

| **No** | **Method** | **Description** |
| --- | --- | --- |
| 01 | getBlogByID(int id) | Return: Blog Detail where blog id contain parameter  Input: ID of blog |
| 02 | getAllBlog() | Return: All blog in database |
| 03 | 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 |
| 04 | 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 |
| 05 | delete(int id) | Delete 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

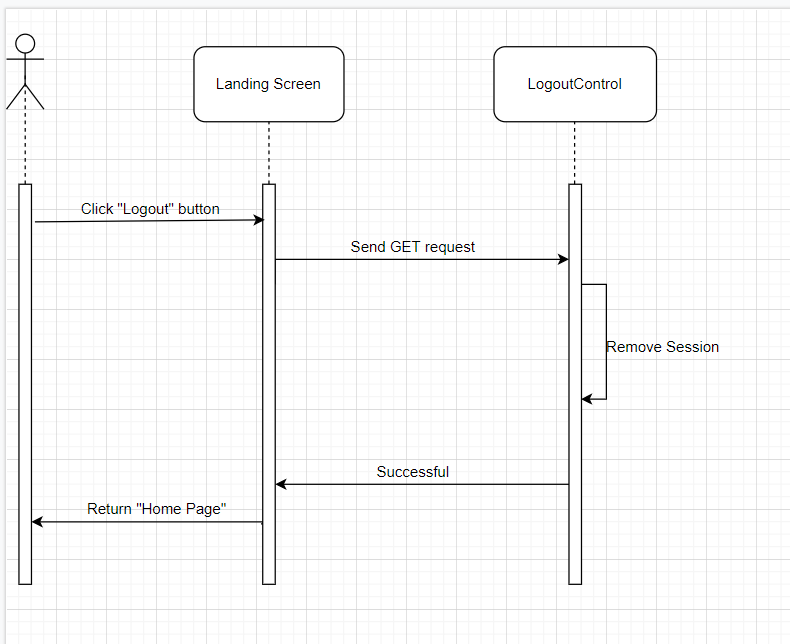
### b.Class Specifications

LogoutController

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

### 

### c. Sequence Diagram(s)

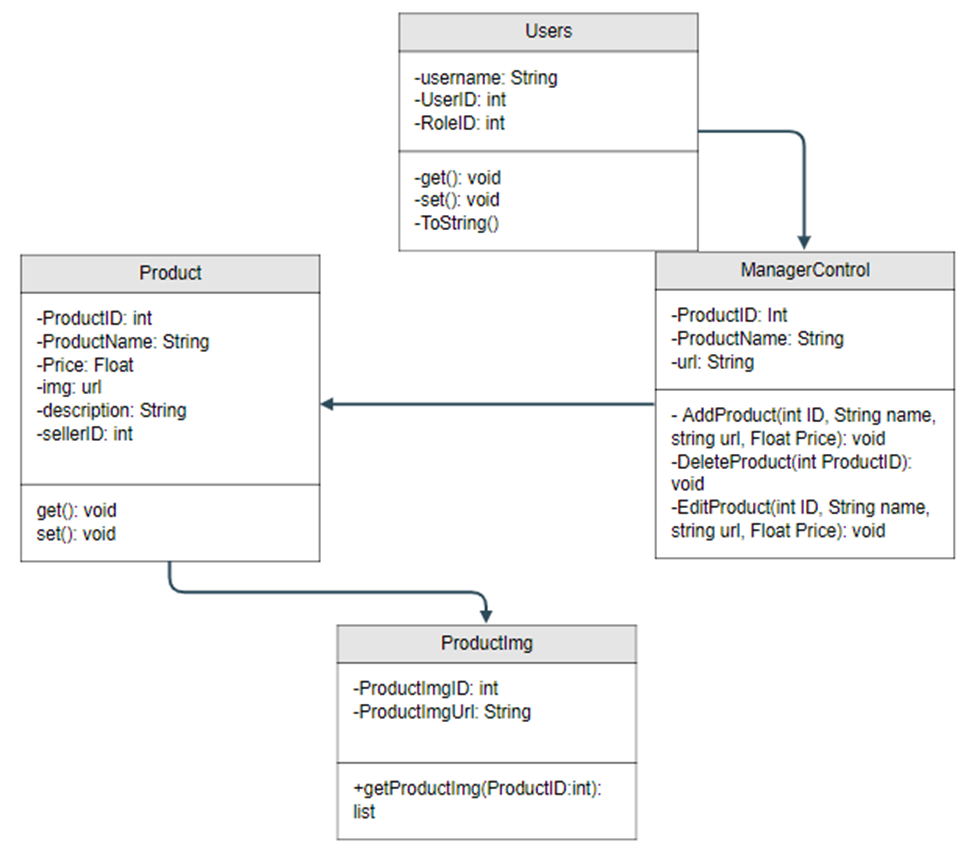


### 

### d. Database queries

## 10. Add Product

### a. Class Diagram

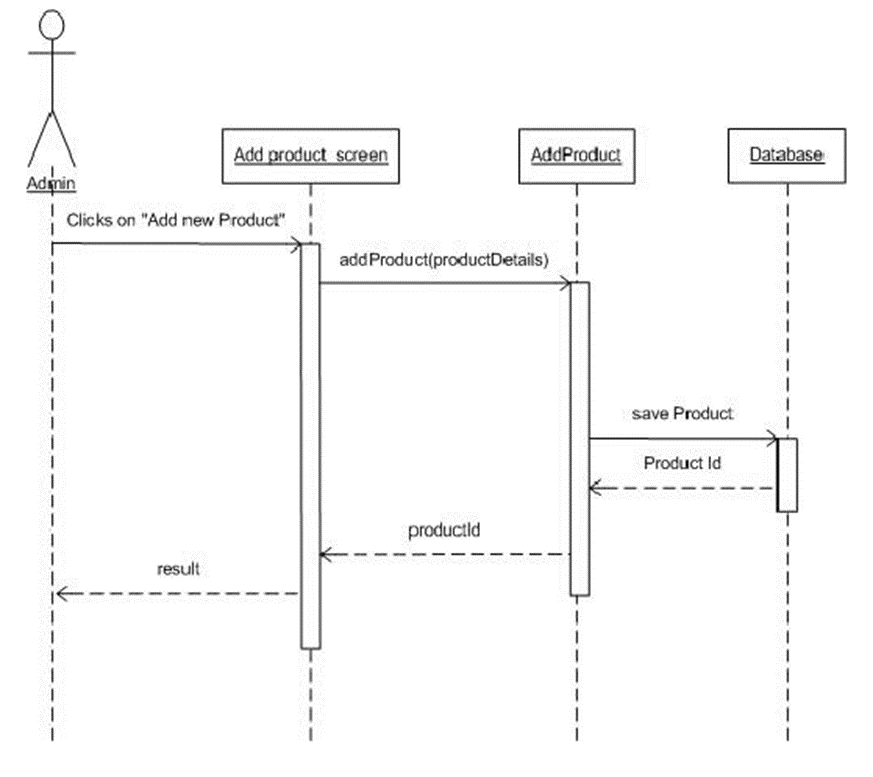


### b. Class Specifications

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

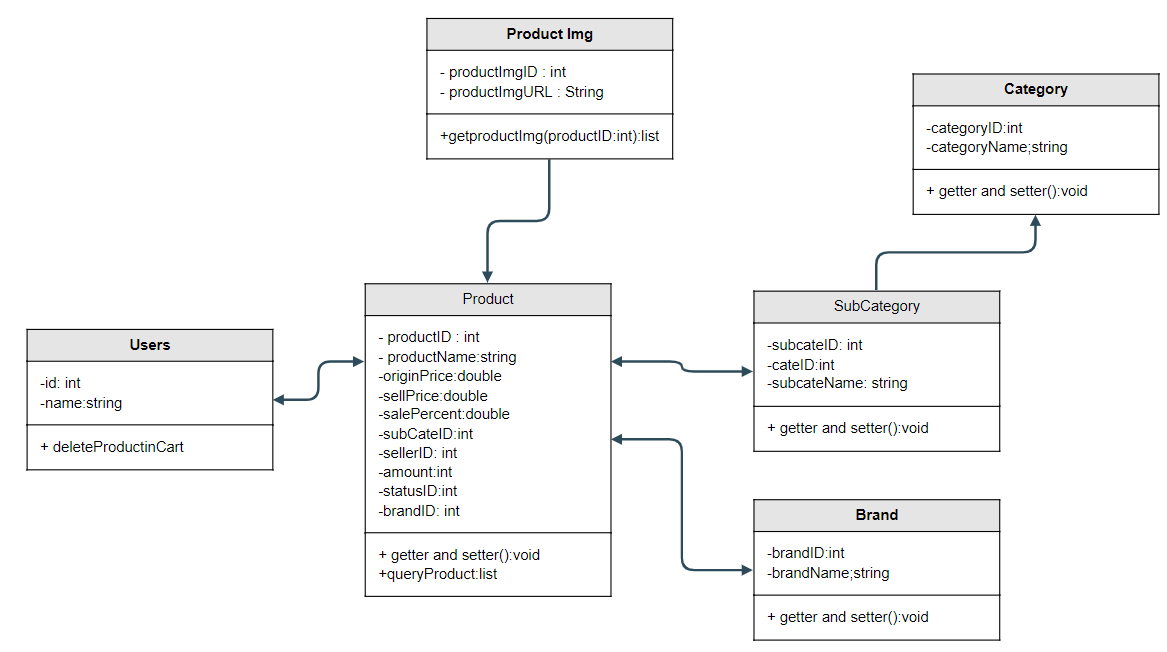
### 

### c. Sequence Diagram(s)

 **d. Database queries**

## 11. Change amount

### a. Class Diagram

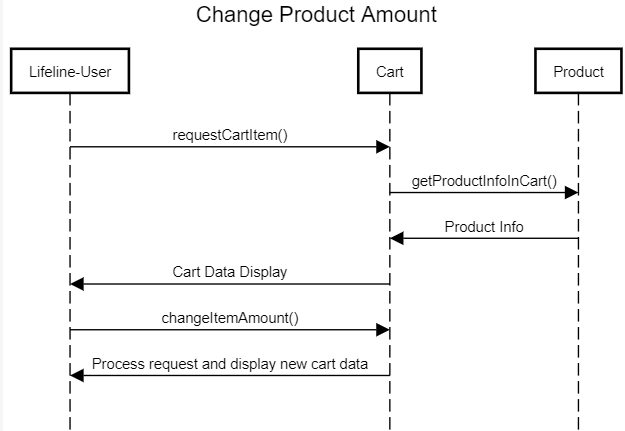


### b. Class Specifications

Customer

| **No** | **Method** | **Description** |
| --- | --- | --- |
| *01* | *changeProductAmount()* | *Add or reduce amount of a product from cart* |

### c. Sequence Diagram(s)



### d. Database queries

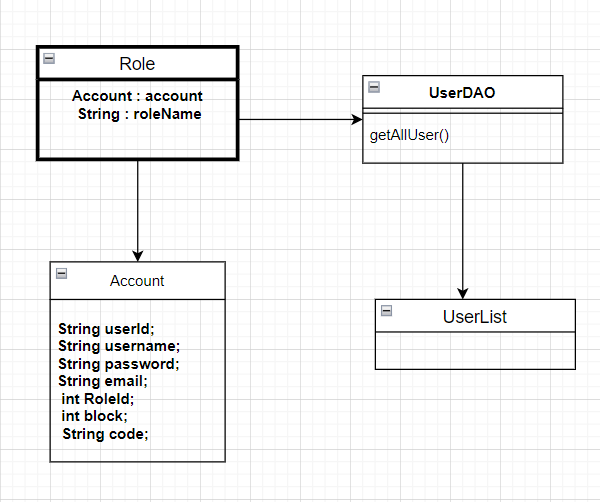
UPDATE dbo.cart

SET amount = ?

WHERE userID = ? and productID = ?

## 12. View List User

### a. Class Diagram



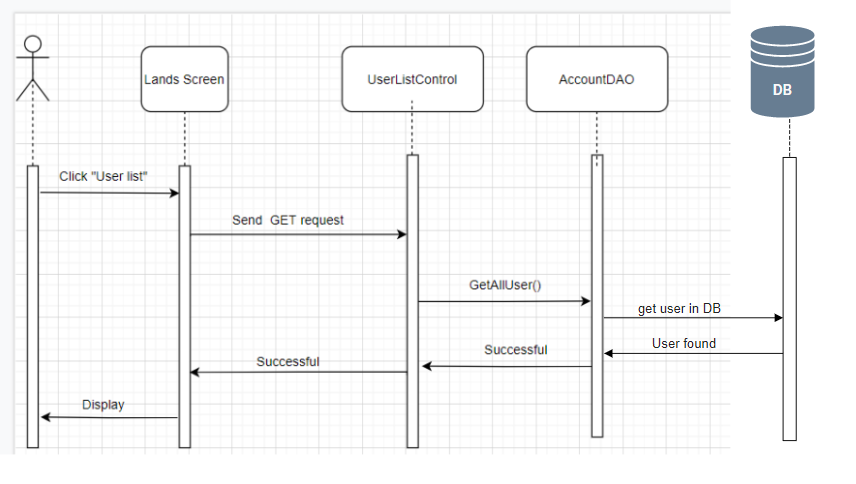
### b. Class Specifications

UserDAO

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

### 

### c. Sequence Diagram(s)



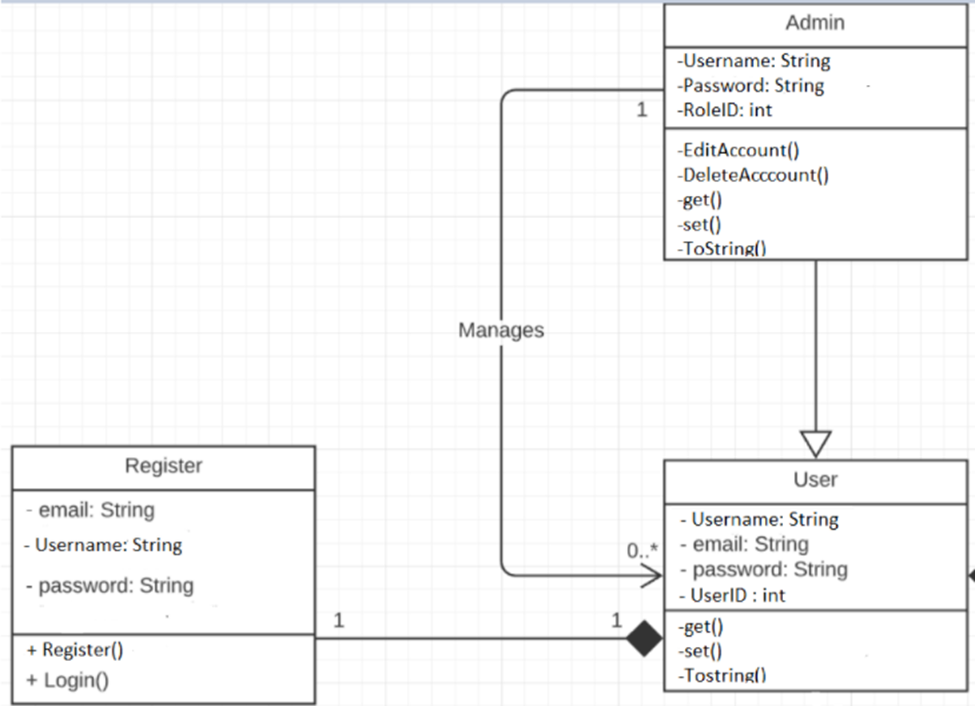
### 

### 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



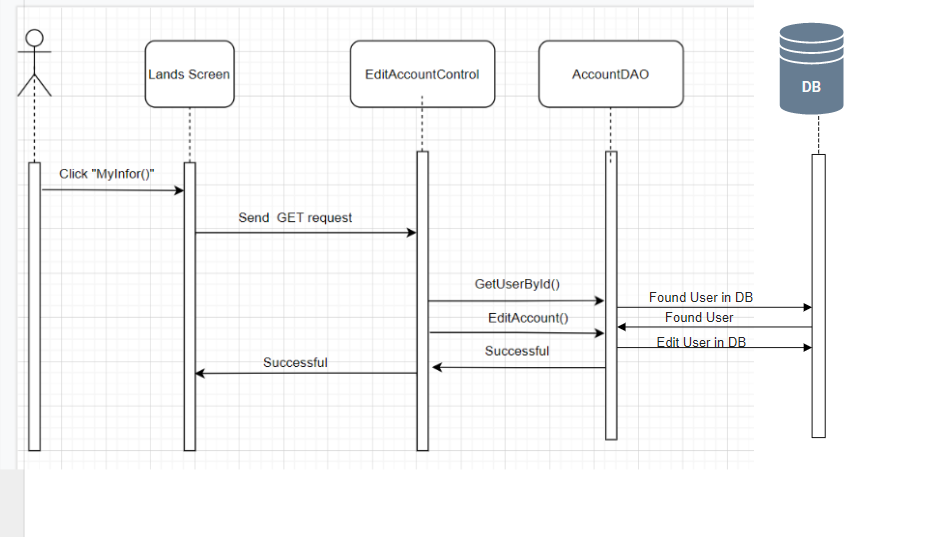
### b. Class Specifications

UserDAO

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

### 

### c. Sequence Diagram(s)

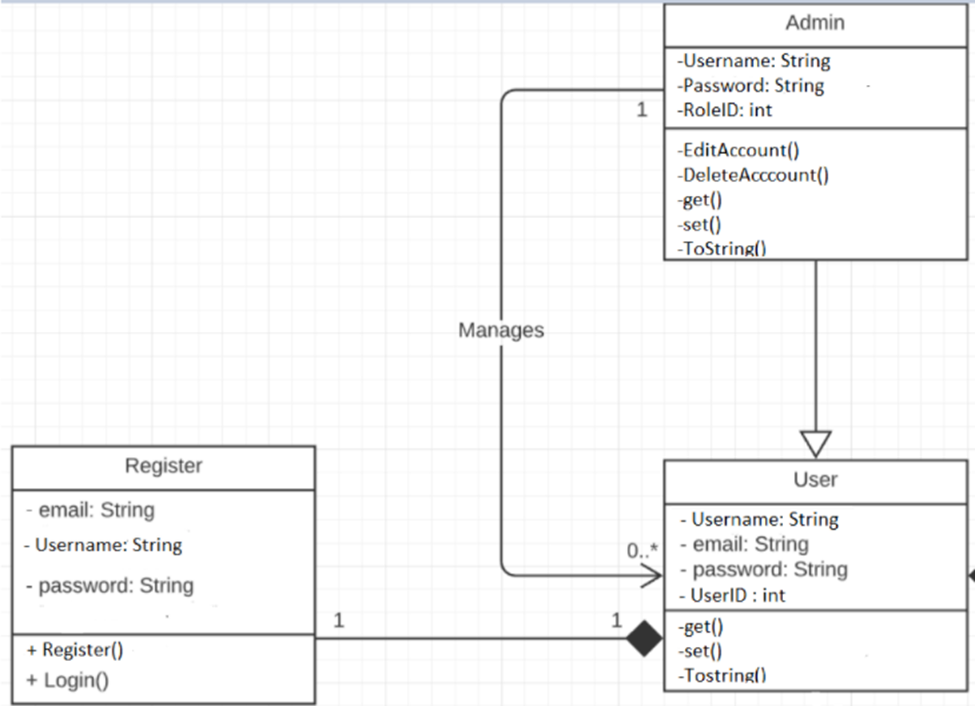


### d. Database queries

Select \* from Users

## 15.Set Role User

### a. Class Diagram



### b. Class Specifications

UserController

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

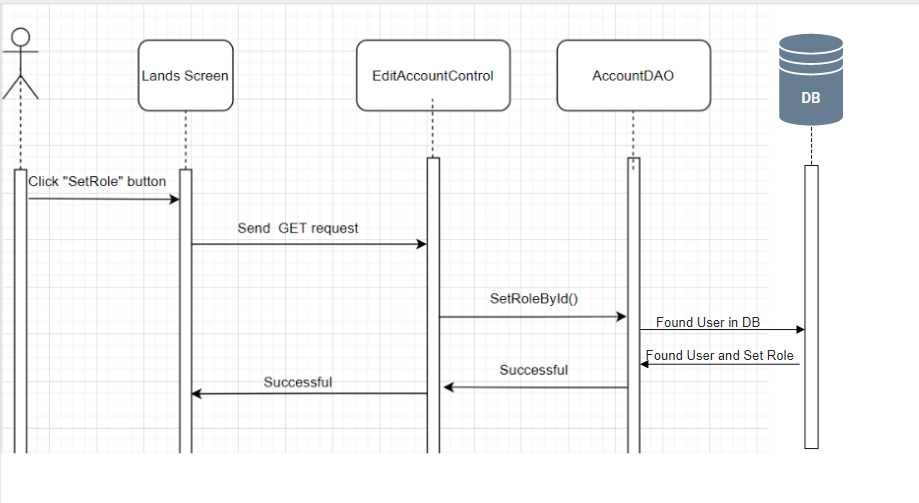
BlogDAO

[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)



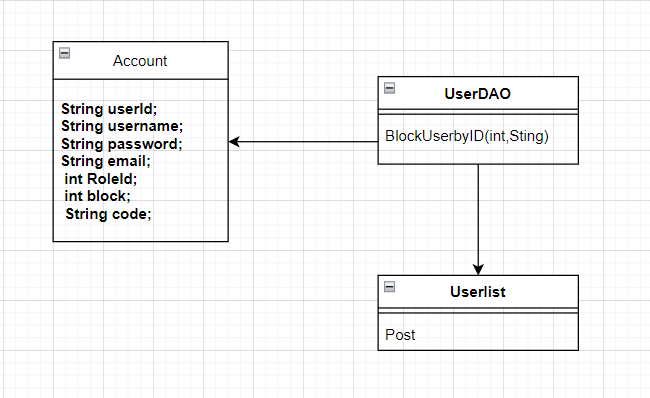
### 

### 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 |

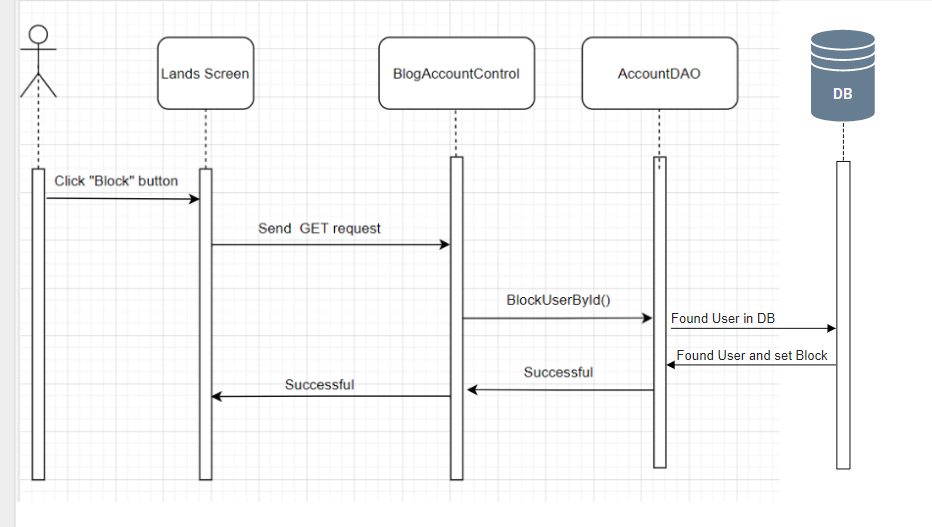
UserDAO

[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

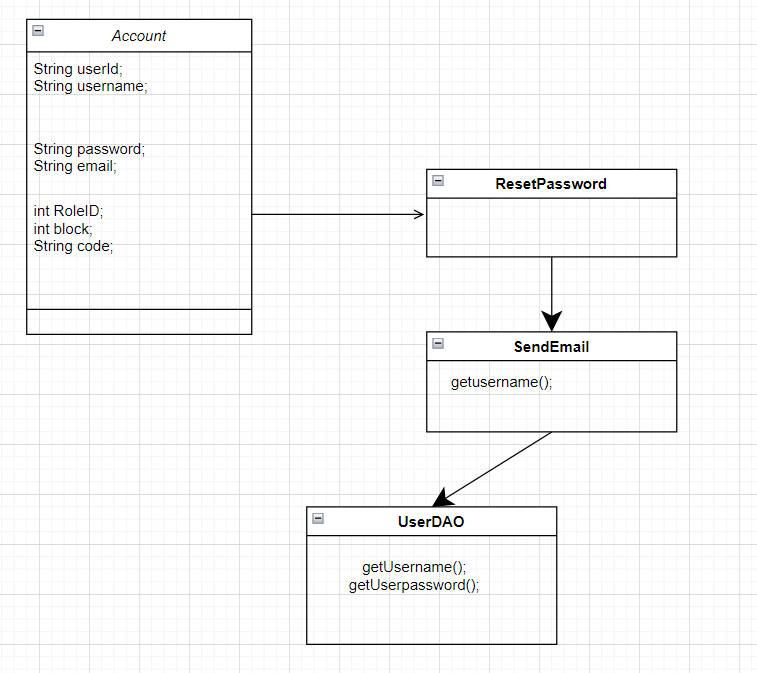
Update users

set isBlock = ?

where UserID = ?

## 17. Reset Password

### Class Diagram



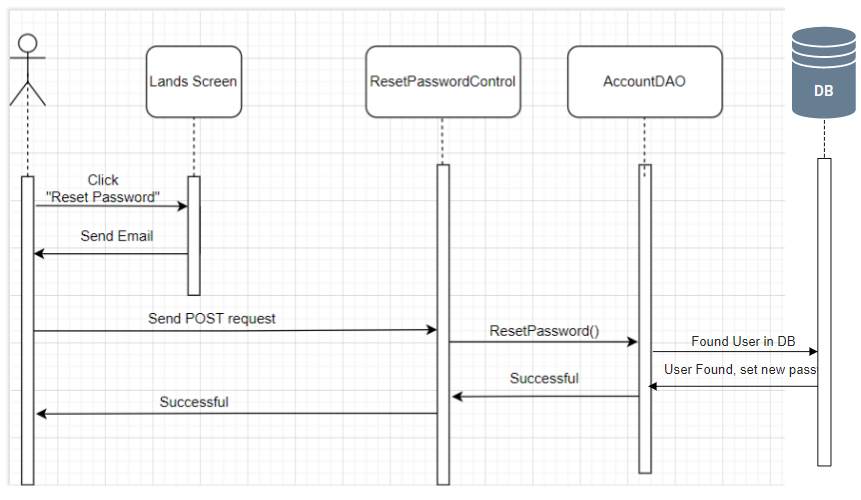
### Class Specifications

| No | Method | Description |
| --- | --- | --- |
| 01 | processRequest() | After user enters their email, system will check that email exists in DB or not. If email exists, system will send new password to that email |

| No | Method | Description |
| --- | --- | --- |
| 01 | updatePassword | Update password after send mail |

### 

### c. Sequence Diagram(s)



### 

### d. Database queries

UPDATE Users

SET Password = ?

WHERE UserID = ?

## 18. Change Password

### a. Class Diagram

### b. Class Specifications

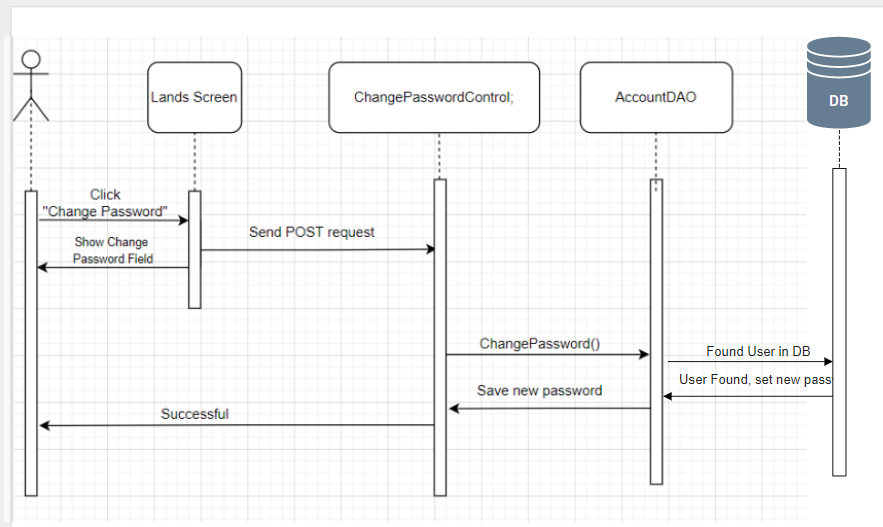
UserDAO

| 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

p

### b. Class Specifications

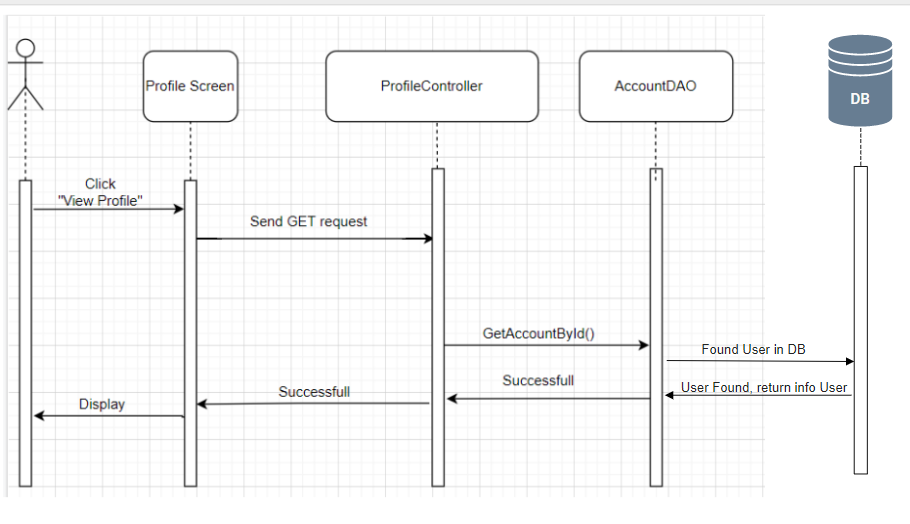
UserController

| 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)



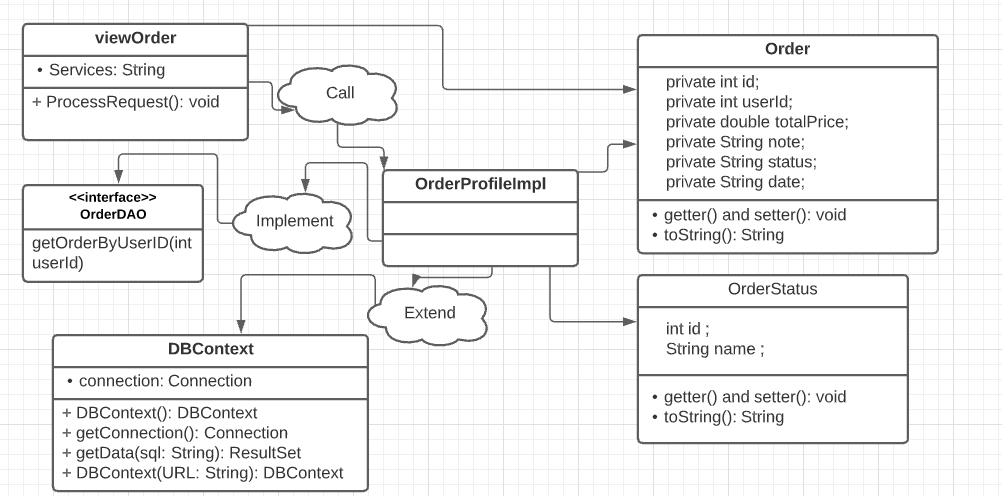
### 

### d. Database queries

Select \* from Users

## 20. Order History

### a. Class Diagram



### b. Class Specifications

UserController

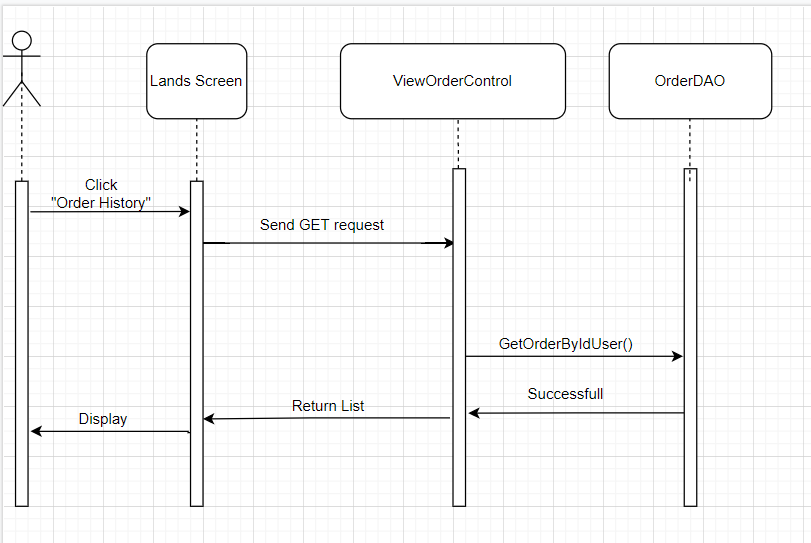
| No | Method | Description |
| --- | --- | --- |
| 01 | processRequest() |  |

OrderDAO

| No | Method | Description |
| --- | --- | --- |
| 01 | viewOrderWait(String userId) | See the product that userId ordered is in the waiting state |
| 02 |  | Check user if exists |
|  |  |  |

### 

### c. Sequence Diagram(s)



### 

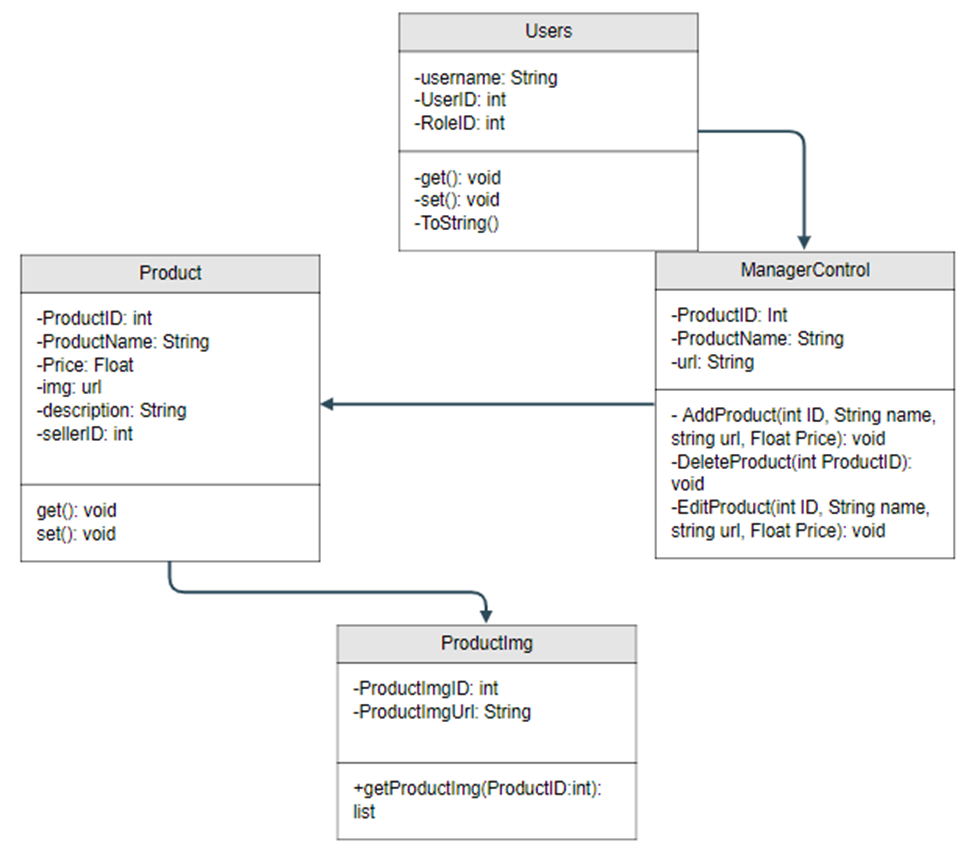
### d. Database queries

Select \* from Users

,

## 21. Delete Product

### a. Class Diagram



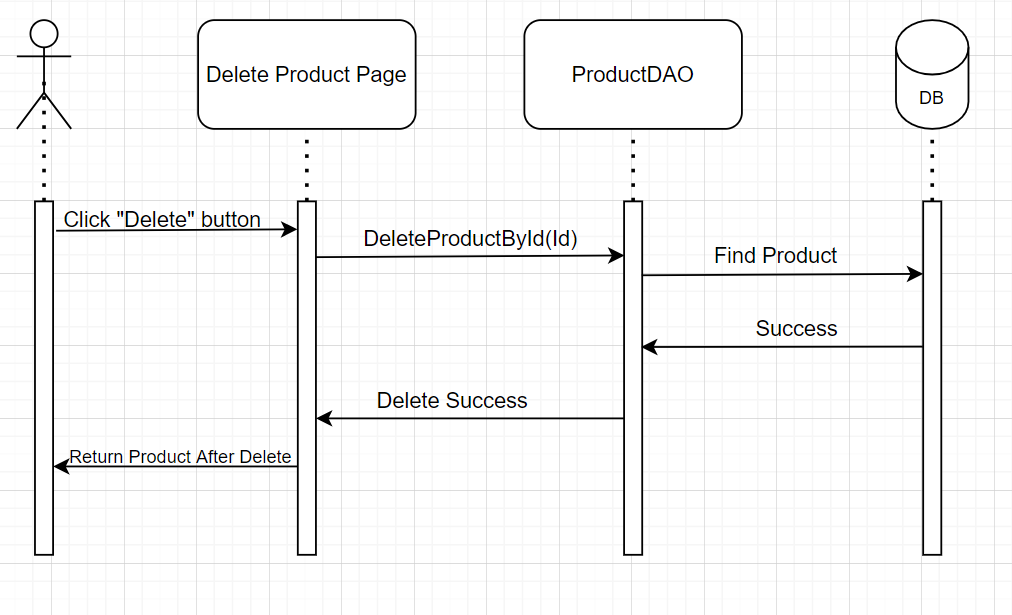
### b. Class Specifications

ProductDAO

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

### 

### c. Sequence Diagram(s)

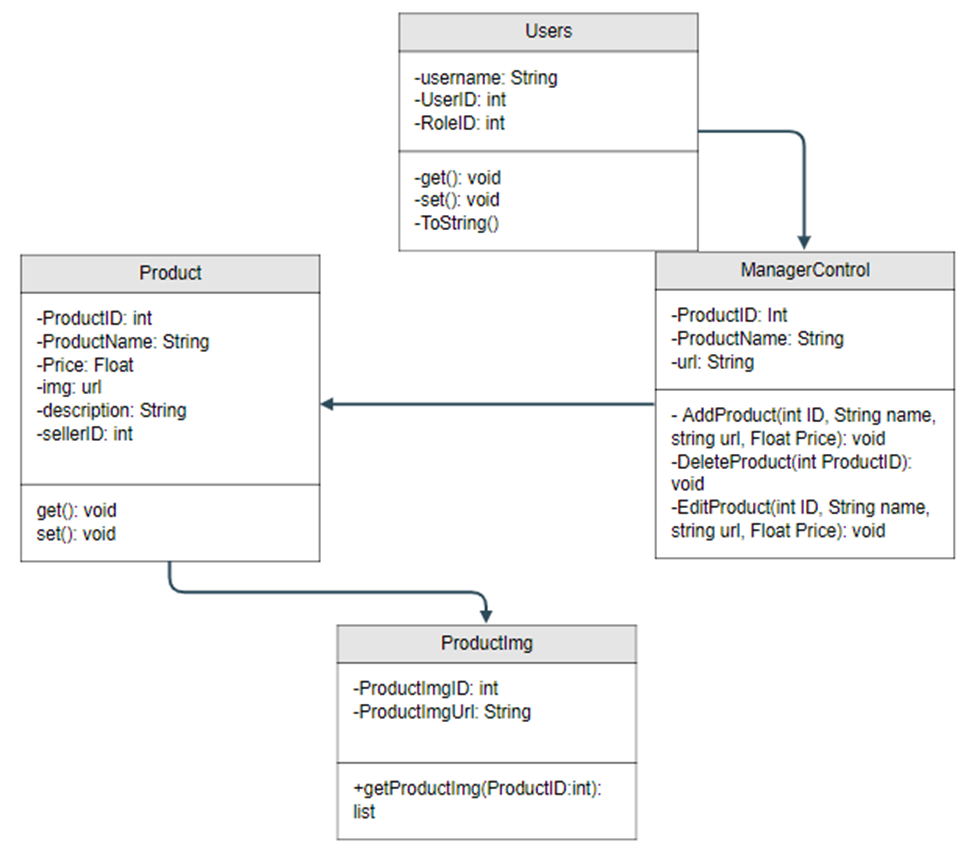


### d. Database queries

delete from Product where ProductID = ?

## 22. Edit Product

### a. Class Diagram



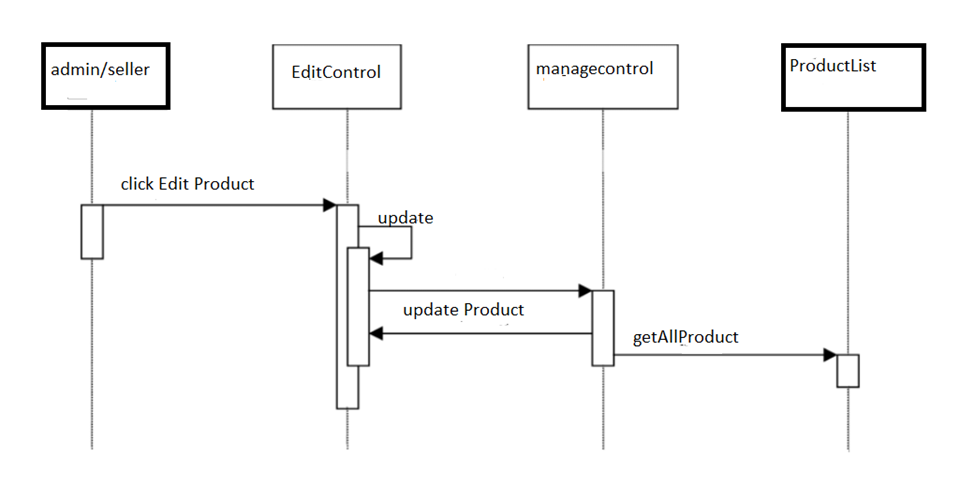
### b. Class Specifications

ProductDAO

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

### 

### c. Sequence Diagram(s)



### 

### d. Database queries

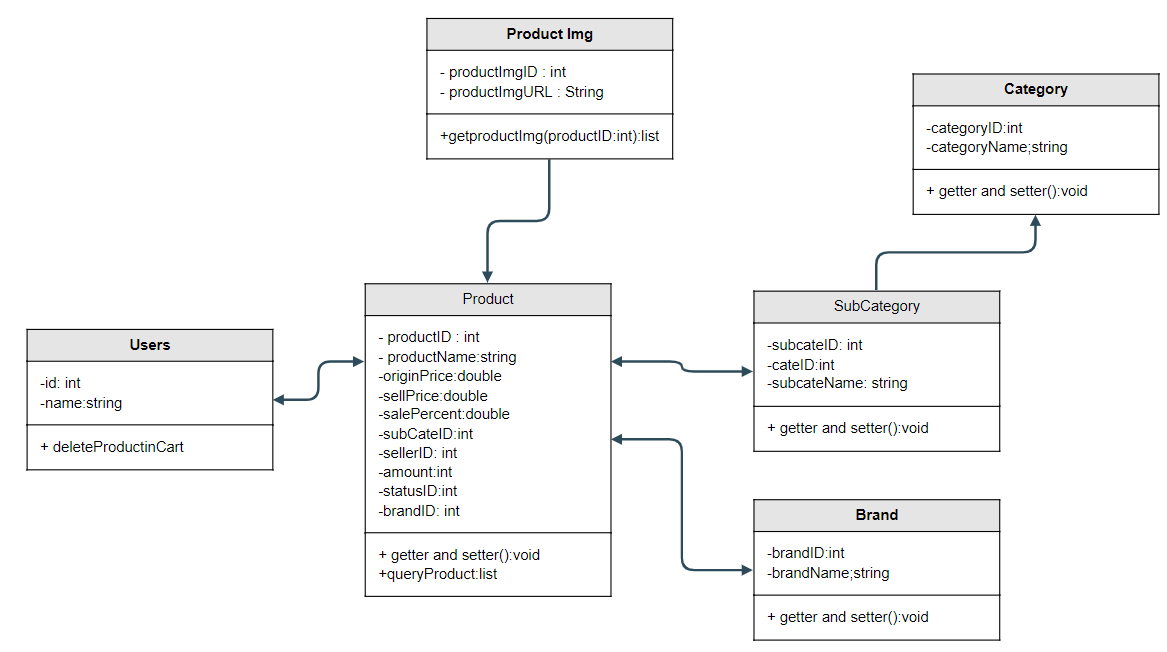
UPDATE datatable

SET ... = ?

WHERE …... = ?

## 23. Checkout

### a. Class Diagram

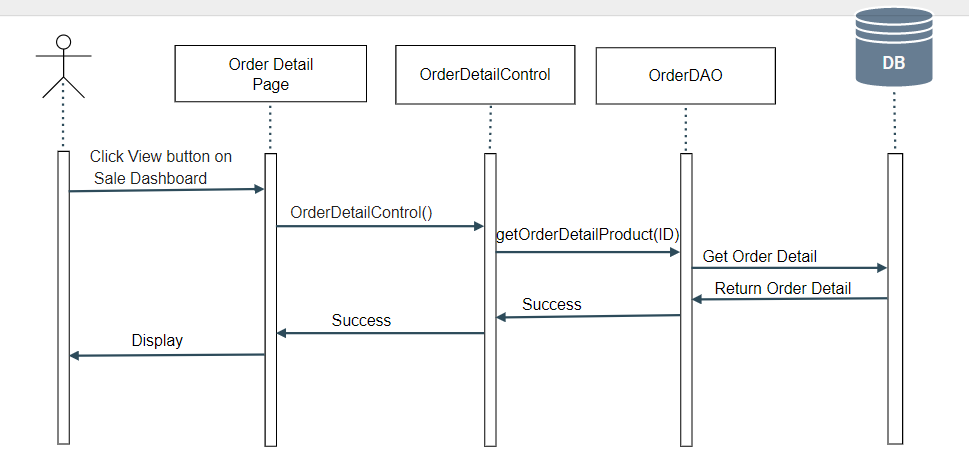


### b. Class Specifications

| 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(),?)

INSERT INTO [dbo].[Order\_Detail]\n"

+ " ([OrderID]\n"

+ " ,[ProductName]\n"

+ " ,[ProductPrice]\n"

+ " ,[Quantity]\n"

+ " ,[ProductID])\n"

+ " VALUES\n"

+ " (?,?,?,?,?)

## 24.Manager Product

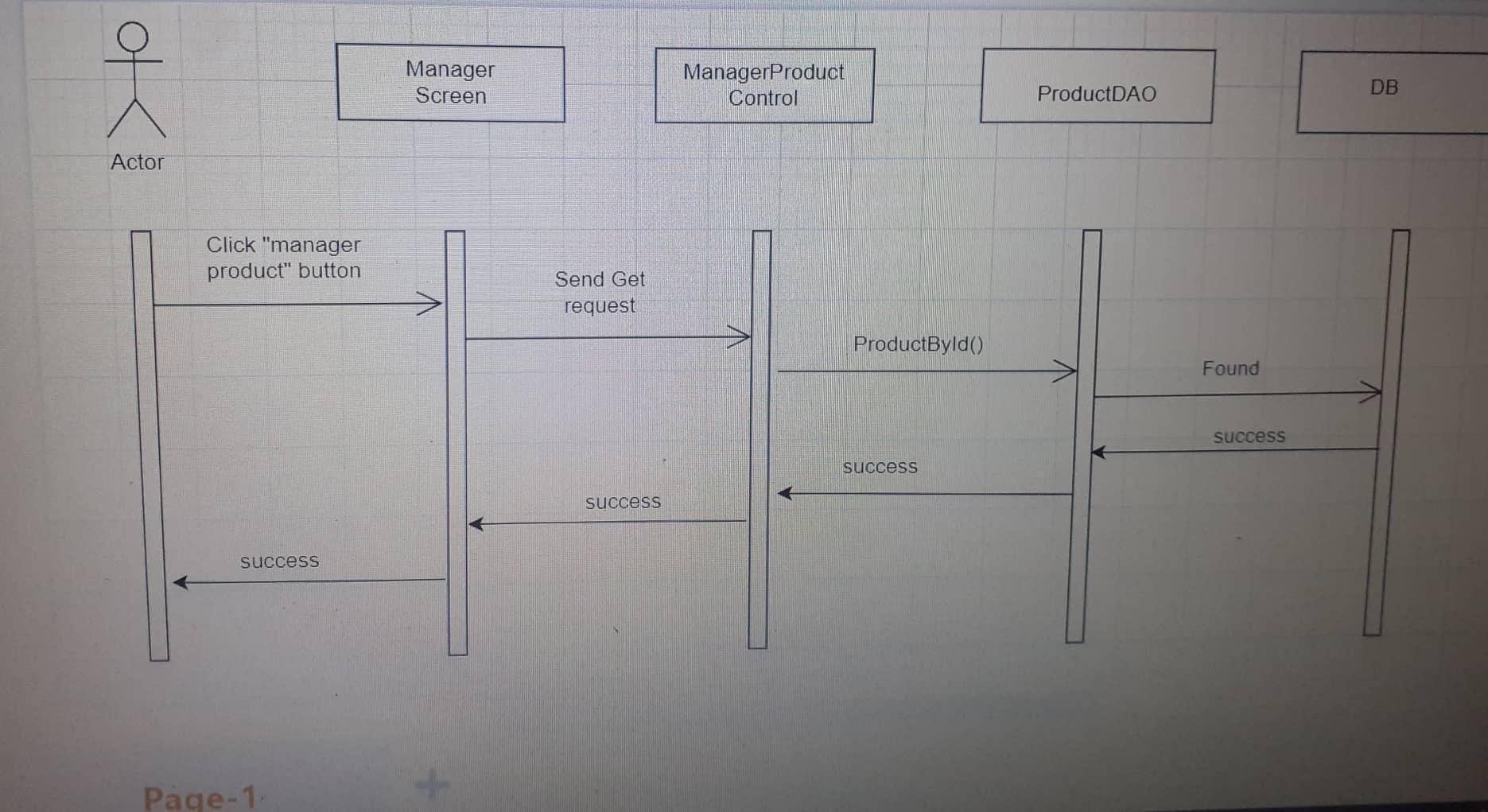
### Class diagram

### 

### Class specifications

| No | Method | Description |
| --- | --- | --- |
| 01 | processRequest() | View and manage products |

### Sequence Diagram(s)



### Database queries

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

+ "on p.ProductID = pr.ProductID

## 25.Home Page

### Class Diagram

### 

### 

### 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* |

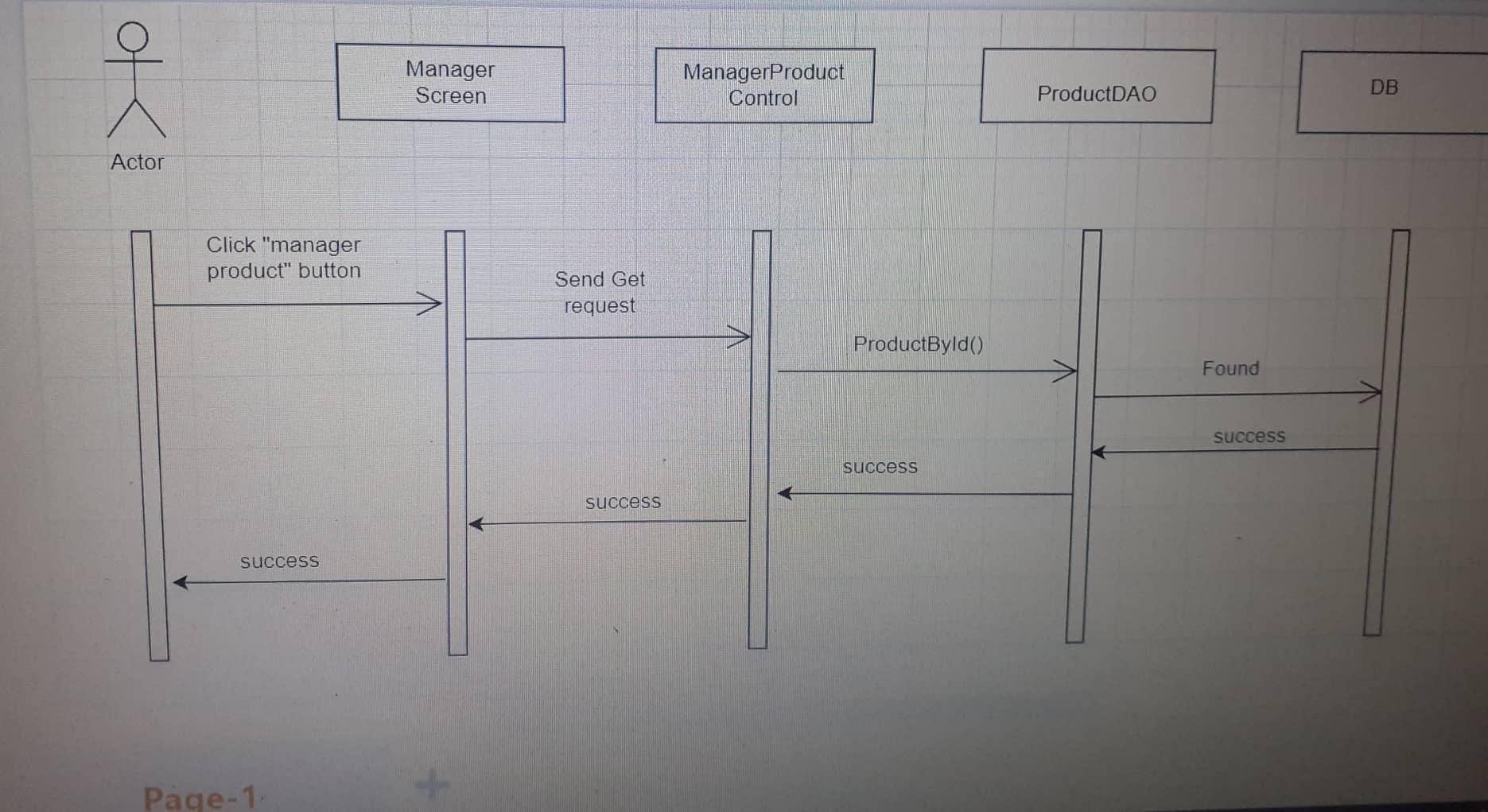
CategoryDAO

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

BrandDAO

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

### Sequence Diagram



1. Database queries

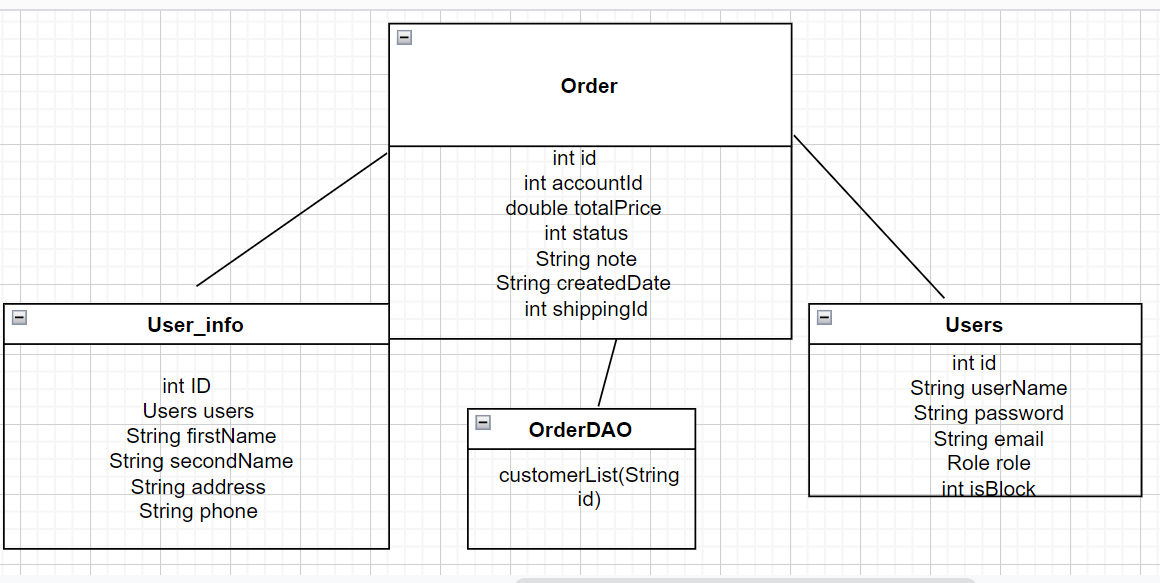
### d. Database queries

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

+ "on p.ProductID = pr.ProductID

## 26. List Customer

### Class Diagram



### Class Specification

| No | Method | Description |
| --- | --- | --- |
| 01 | processRequest() | Review a list of past customers. |

### Sequence Diagram(s)

### 

### Database queries

select \* from Orders join Users

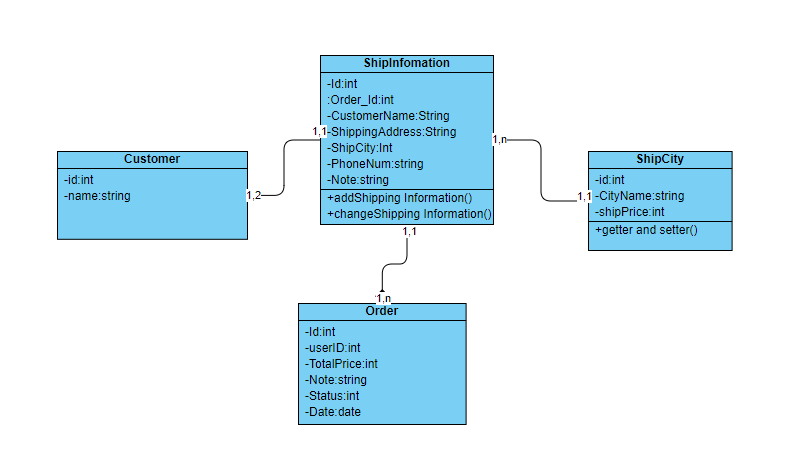
on Orders.UserID = Users.UserID

join User\_info as c

on Users.UserID = c.UserID

## 26. Add Ship Information

### a. Class Diagram



### b. Class Specifications

Ship Information Class

| **No** | **Method** | **Description** |
| --- | --- | --- |
| *01* | *addShipping information()* | *Add New Shipping information for account* |

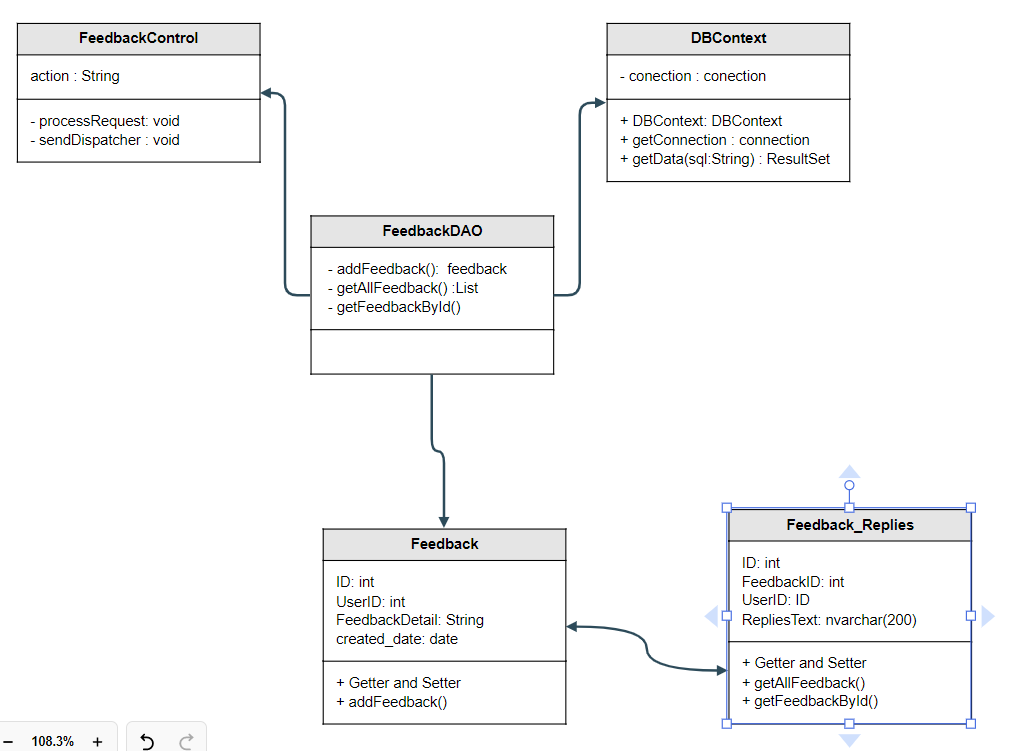
### c. Sequence Diagram(s)

### d. Database queries

INSERT INTO dbo.ShipInfo(order\_id,customerName,shippingAddress,ShipCity,PhoneNum,Note) VALUES(?,?,?,?,?,?)

## 27. Feedback

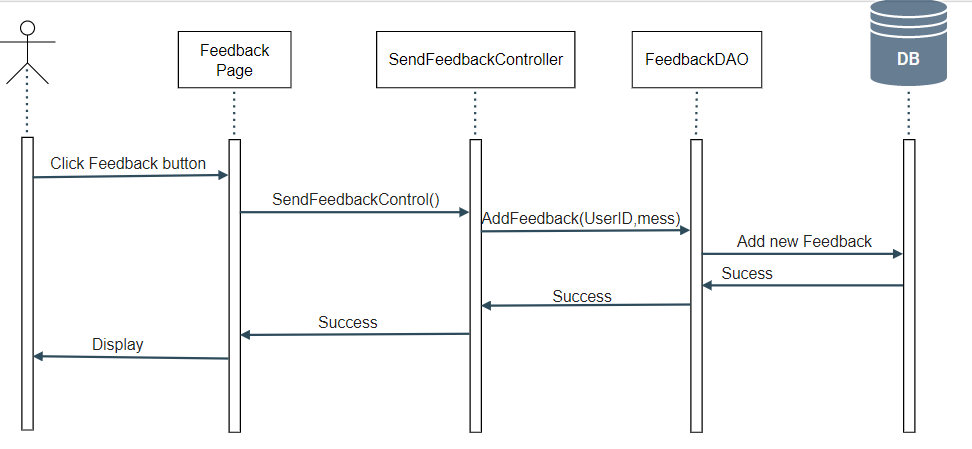
### a. Class Diagram



### b. Class Specifications

| **No** | **Method** | **Description** |
| --- | --- | --- |
| *01* | *addFeedback()* | *Add new Feedback* |
| *02* | *getAllFeedback()* | *Get all Feedback to display* |

### c. Sequence Diagram(s)



### d. Database queries

INSERT INTO [dbo].[Feedback]\n"

+ " ([UserID]\n"

+ " ,[FeedbackDetail]\n"

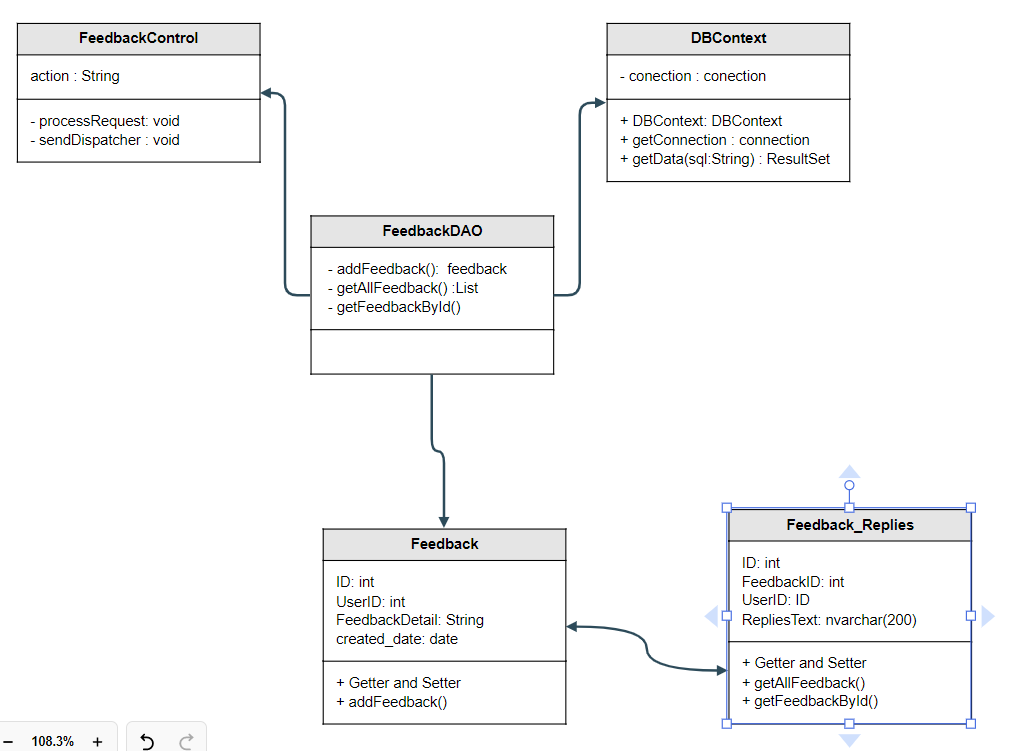
+ " ,[created\_date])\n"

+ " VALUES\n"

+ " (?,?,GETDATE())

## 28. Feedback Replies

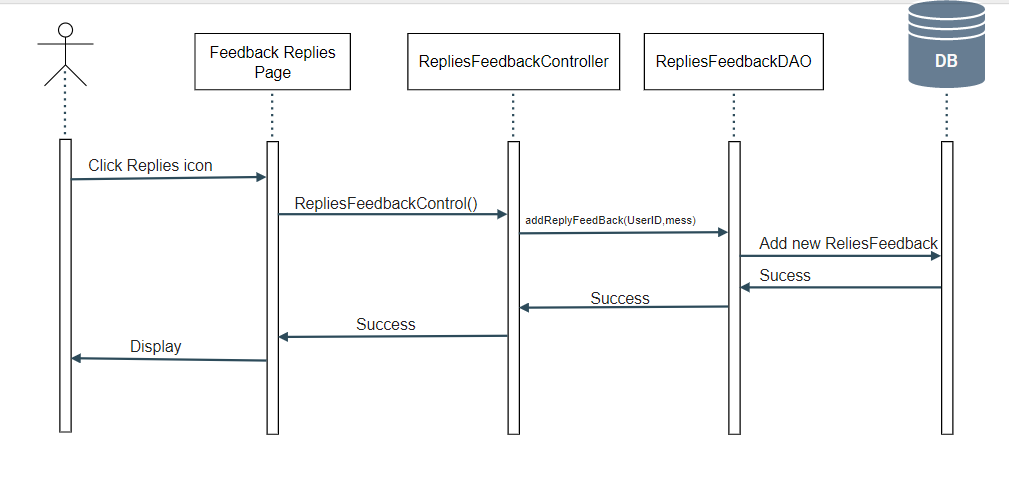
### a. Class Diagram



### b. Class Specifications

| **No** | **Method** | **Description** |
| --- | --- | --- |
| *01* | *GetFeedbackByID* | *Get Feedback by UserID* |
| *02* | *addRepliesFeedback()* | *Replies Feedback* |
|  |  |  |

### c. Sequence Diagram(s)



### d. Database queries

INSERT INTO [dbo].[Feedback\_Replies]\n"

+ " ([FeedbackID]\n"

+ " ,[UserID]\n"

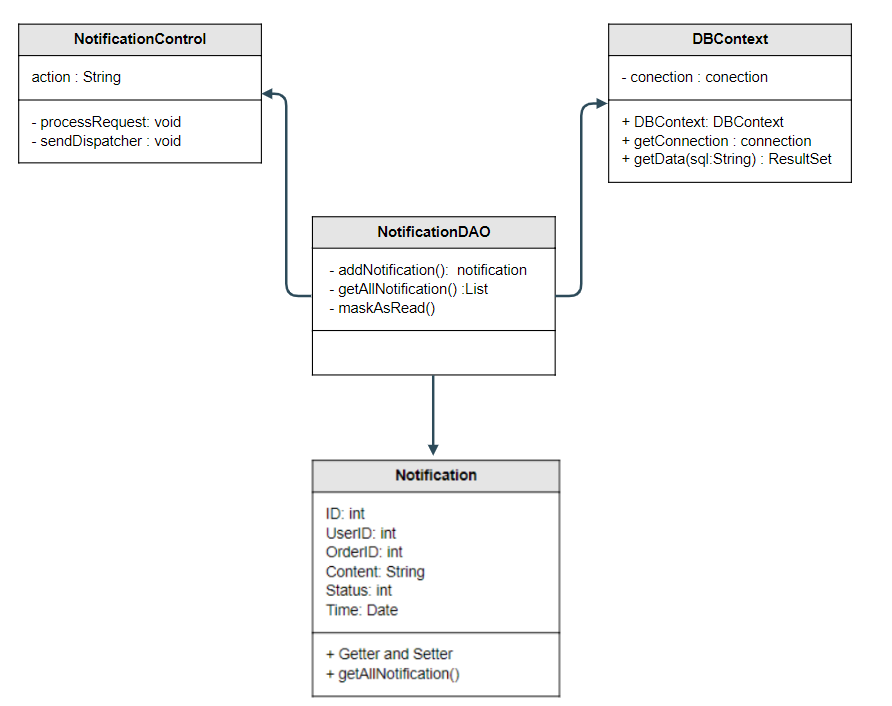
+ " ,[RepliesText])\n"

+ " VALUES\n"

+ " (?,?,?)"

## 29. Notification

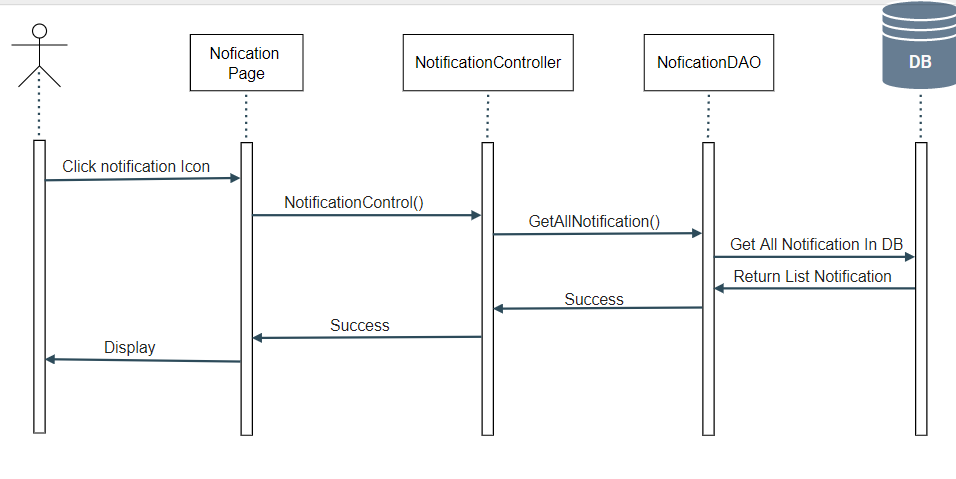
### a. Class Diagram



### b. Class Specifications

| **No** | **Method** | **Description** |
| --- | --- | --- |
| *01* | *addNotificatioṇ()* | *Add new Notification* |
| *02* | *GetAllNotification()* | *Get all notification to display* |

### c. Sequence Diagram(s)



### d. Prod. Database queries

SELECT \* FROM dbo.[Notifications] WHERE UserID = ? ORDER BY [status] ASC, [time] DESC

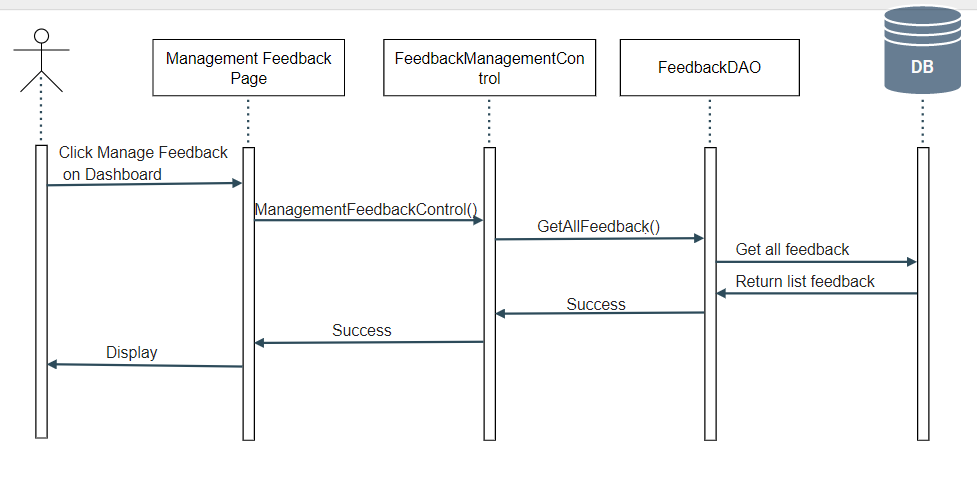
## 30. Manage Feedback

### a. Class Diagram

### b. Class Specifications

| **No** | **Method** | **Description** |
| --- | --- | --- |
| *01* | *getAllFeedback()* | *Get all feedback and display* |
|  |  |  |

### c. Sequence Diagram(s)

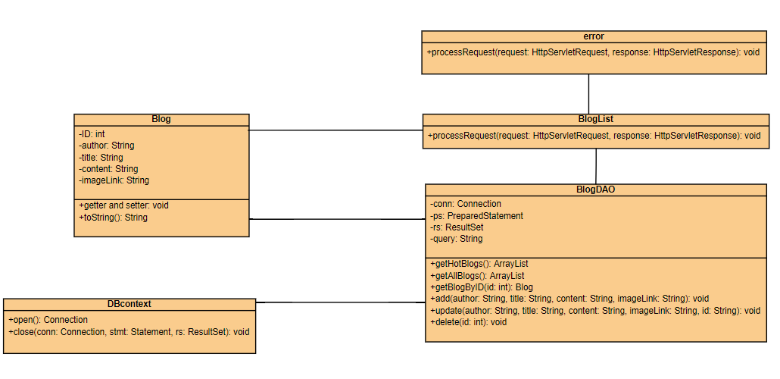


### d. Database queries

SELECT Feedback.\*, Users.Username from Feedback inner join Users on Feedback.UserID = Users.UserID

## 31. Management Blog

### a. Class Diagram



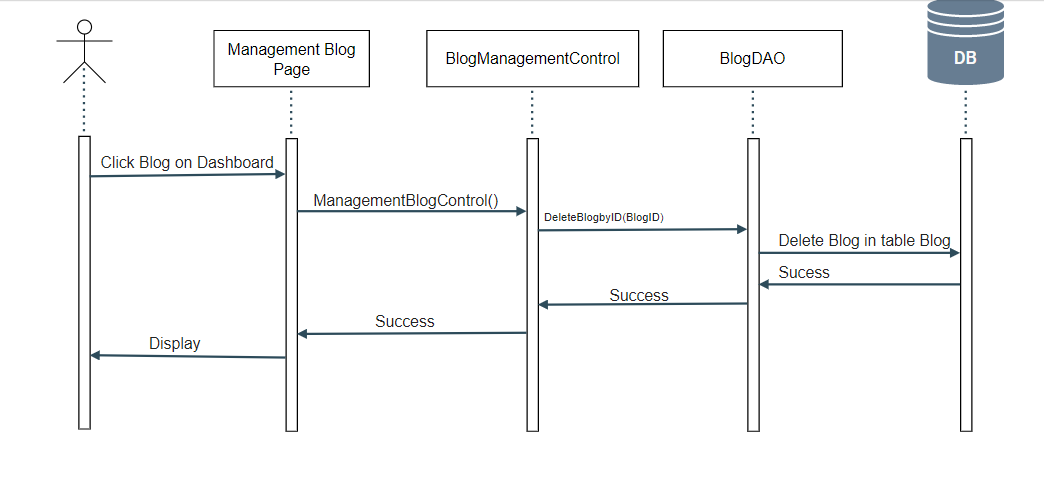
### b. Class Specifications

| **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 | getAllBlog() | Return: All blog in database |
| 03 | 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 |
| 04 | 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 |
| 05 | delete(int id) | Delete blog with id entered in blogManage.jsp  Return: null  Input: id |

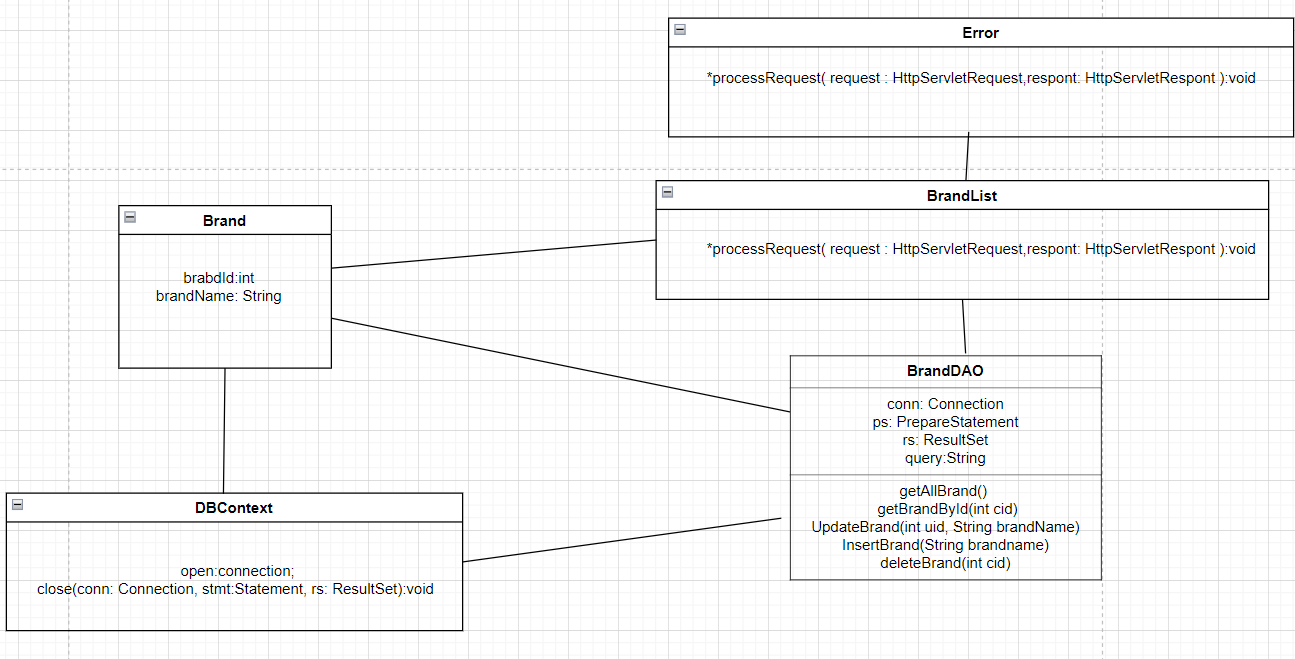
### c. Sequence Diagram(s)



### d. Database queries

## 32. Manager Brand

### a. Class Diagram



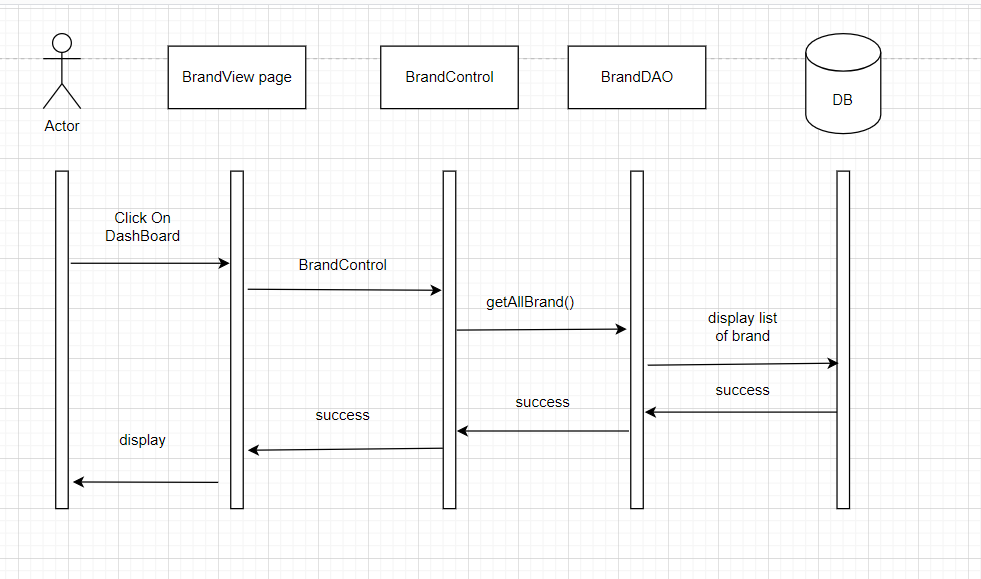
### b. Class Specifications

| **No** | **Method** | **Description** |
| --- | --- | --- |
| 01 | processRequest() | get brand list from database to display on BrandView.jsp |

- BrandDAO class

| **No** | **Method** | **Description** |
| --- | --- | --- |
| 01 | getBrandById(int cid) | Return: Brand Detail where brand id contain parameter  Input: ID of brand |
| 02 | getAllBrand() | Return: All brand in database |
| 03 | InsertBrand(String brandname) | Add brand with detail entered in AddBrand.jsp page  Return: null  Input: brand name |
| 04 | UpdateBrand(int uid, String brandName) | Update brand with detail entered in EditBrand.jsp page  Return: null  Input: brand name |
| 05 | deleteBrand(int cid) | Delete brand with id entered in BrandView.jsp  Return: null  Input: id |

### c. Sequence Diagram(s)

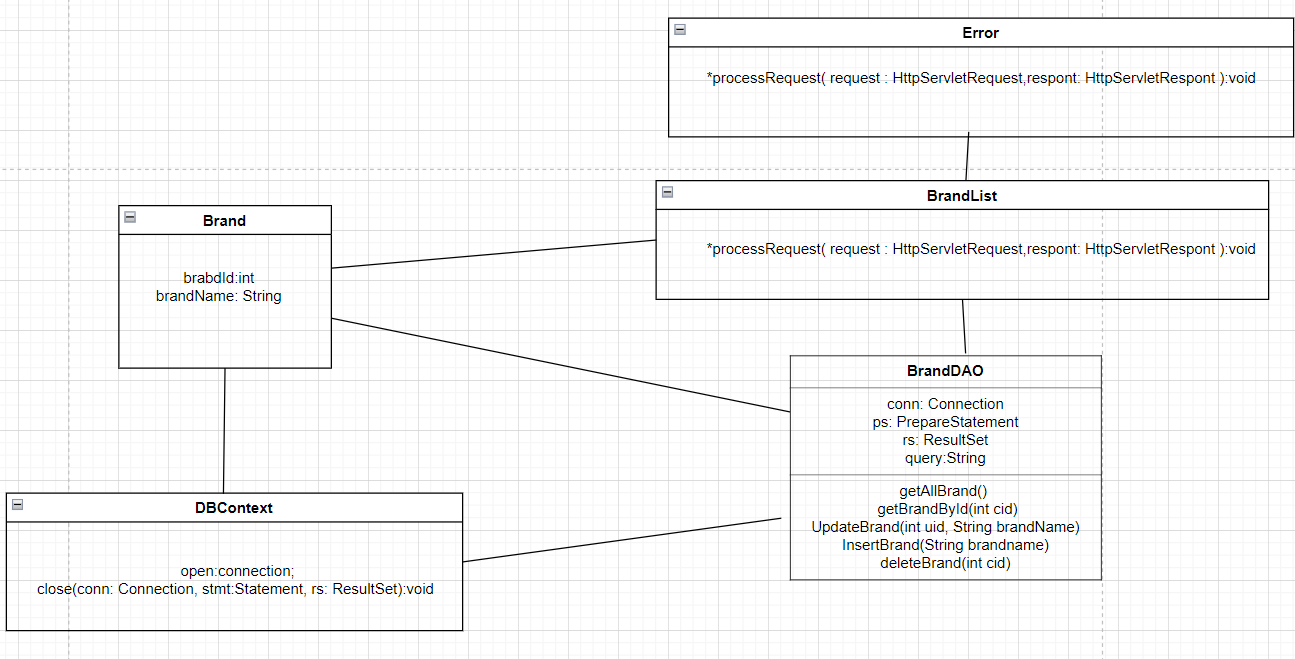


### d. Database queries

SELECT \* FROM dbo.Brand

## 33.Edit Brand

### a. Class Diagram



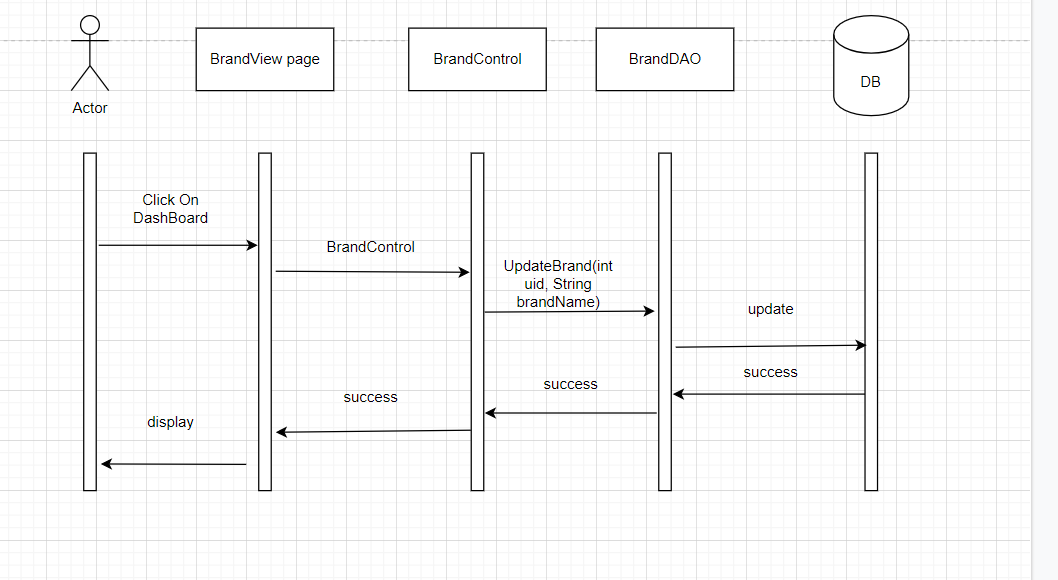
### b. Class Specifications

| **No** | **Method** | **Description** |
| --- | --- | --- |
| 01 | processRequest() | get brand list from database to display on BrandView.jsp |

- BrandDAO class

| **No** | **Method** | **Description** |
| --- | --- | --- |
| 01 | getBrandById(int cid) | Return: Brand Detail where brand id contain parameter  Input: ID of brand |
| 02 | getAllBrand() | Return: All brand in database |
| 03 | InsertBrand(String brandname) | Add brand with detail entered in AddBrand.jsp page  Return: null  Input: brand name |
| 04 | UpdateBrand(int uid, String brandName) | Update brand with detail entered in EditBrand.jsp page  Return: null  Input: brand name |
| 05 | deleteBrand(int cid) | Delete brand with id entered in BrandView.jsp  Return: null  Input: id |

### c. Sequence Diagram(s)



### d. Database queries

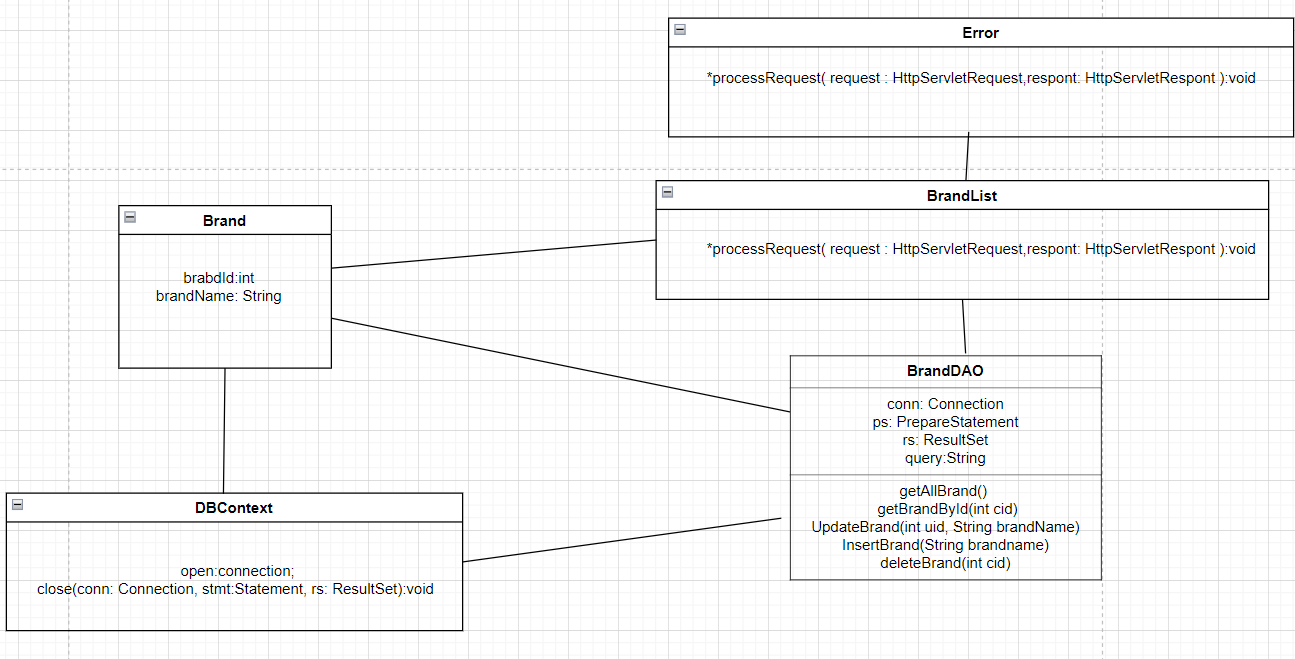
"UPDATE [dbo].[Brand]\n"

+ " SET [BrandName] = ?\n"

+ " WHERE BrandID = ?"

## 34. Add Brand

### a. Class Diagram



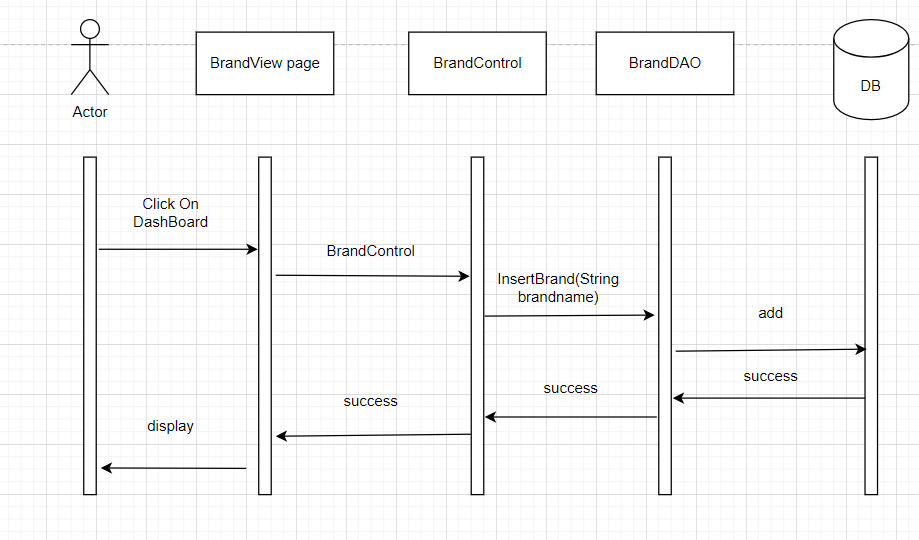
### b. Class Specifications

| **No** | **Method** | **Description** |
| --- | --- | --- |
| 01 | processRequest() | get brand list from database to display on BrandView.jsp |

- BrandDAO class

| **No** | **Method** | **Description** |
| --- | --- | --- |
| 01 | getBrandById(int cid) | Return: Brand Detail where brand id contain parameter  Input: ID of brand |
| 02 | getAllBrand() | Return: All brand in database |
| 03 | InsertBrand(String brandname) | Add brand with detail entered in AddBrand.jsp page  Return: null  Input: brand name |
| 04 | UpdateBrand(int uid, String brandName) | Update brand with detail entered in EditBrand.jsp page  Return: null  Input: brand name |
| 05 | deleteBrand(int cid) | Delete brand with id entered in BrandView.jsp  Return: null  Input: id |

### c. Sequence Diagram(s)



### d. Database queries

"INSERT INTO [dbo].[Brand]\n"

+ " ([BrandName])\n"

+ " VALUES\n"

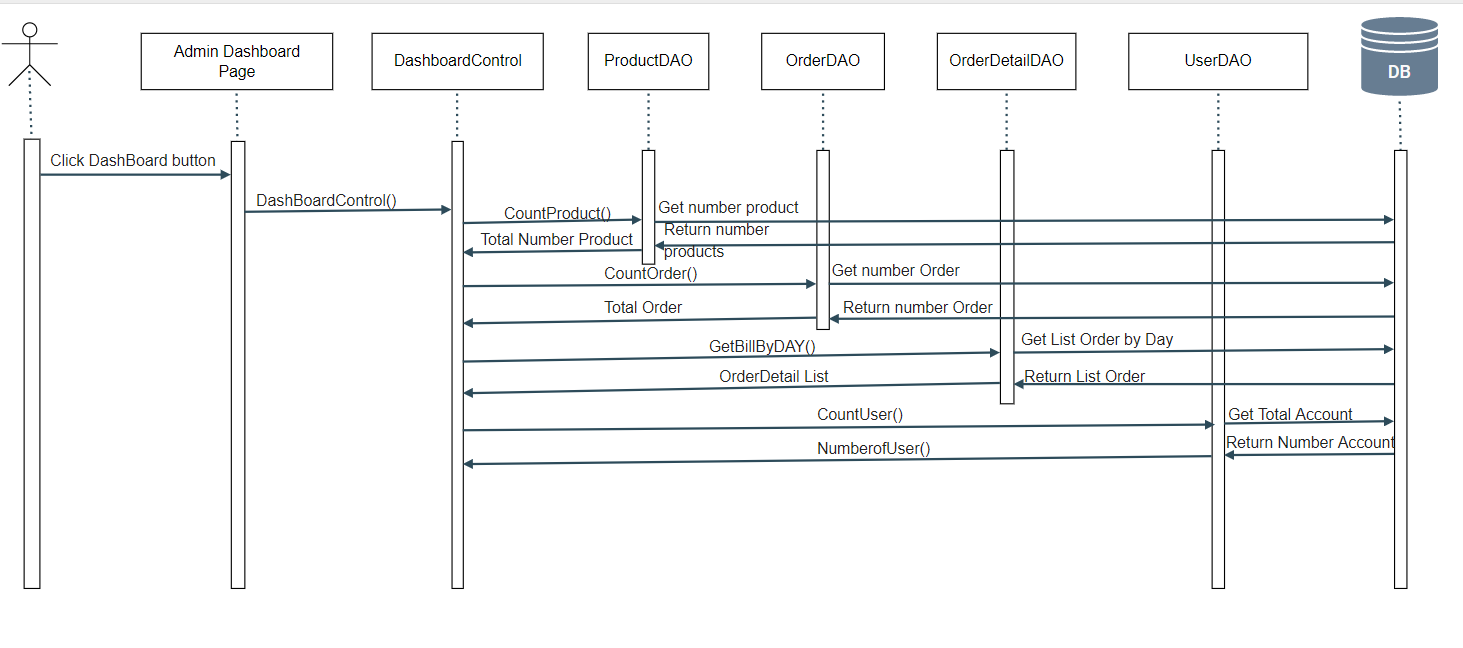
+ " (?)"

## 35. Admin DashBoard

### a. Class Diagram

### b. Class Specifications

### c. Sequence Diagram(s)



### d. Database queries

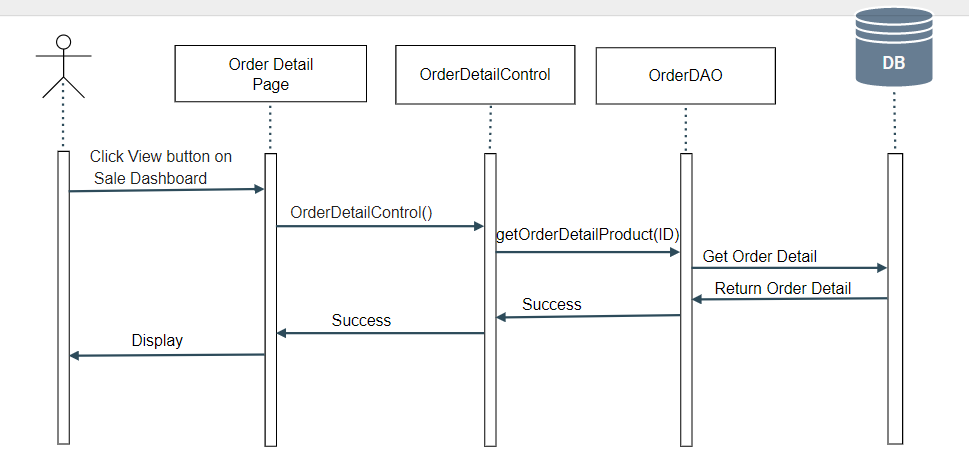
## 36. Order Detail[Sale]

### a. Class Diagram

### b. Class Specifications

| **No** | **Method** | **Description** |
| --- | --- | --- |
| 1 | getOrderDetailProduct | Get Order Detail by ID |

### c. Sequence Diagram(s)



### d. Database queries

SELECT od.OrderID,p.ProductName,od.ProductPrice,od.Quantity,proi.ProductImgURL from Order\_Detail Cod\n"

+ "inner join Product p on od.ProductID = p.ProductID\n"

+ "inner join ProductImg proi on proi.ProductID = p.ProductID where od.OrderID = ?

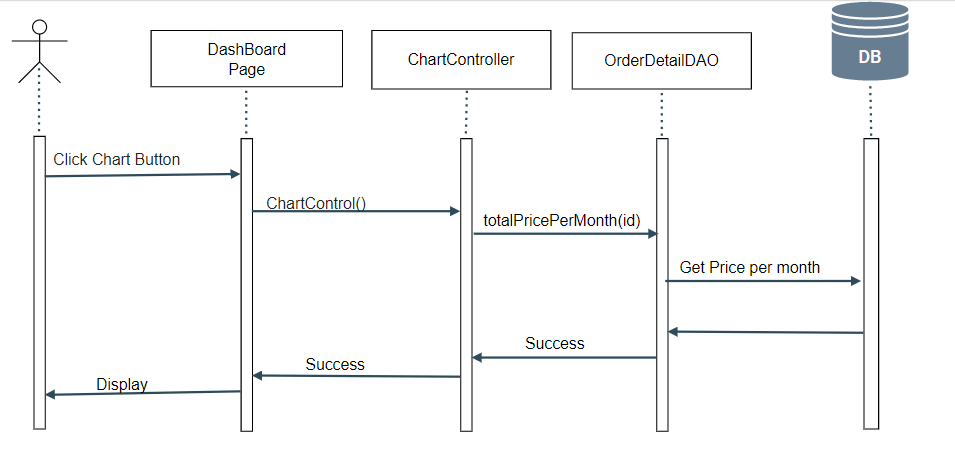
## 38. Chart Management

### a. Class Diagram

### b. Class Specifications

| **No** | **Method** | **Description** |
| --- | --- | --- |
| 1 | totalPricePerMonth() | Get price per month |

### c. Sequence Diagram(s)



### d. Database queries

select totalPrice from \n"

+ " (select MONTH(date) as SalesMonth,\n"

+ " SUM(od.ProductPrice) as totalPrice\n"

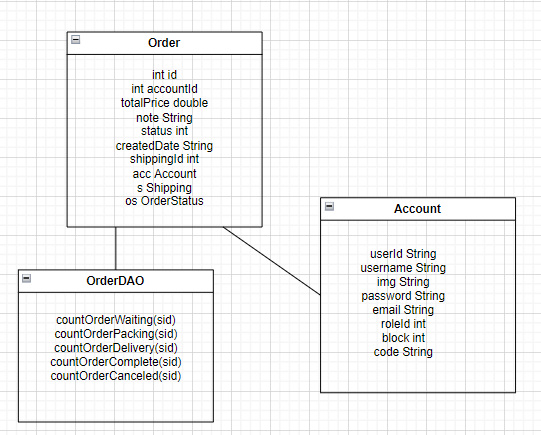
+ " from [Orders] o , Order\_Detail od\n"

+ " where o.ID = od.orderID\n"

+ " GROUP BY MONTH(date)) as t where t.SalesMonth = ?

## 39. View Order History

## a. Class Diagram

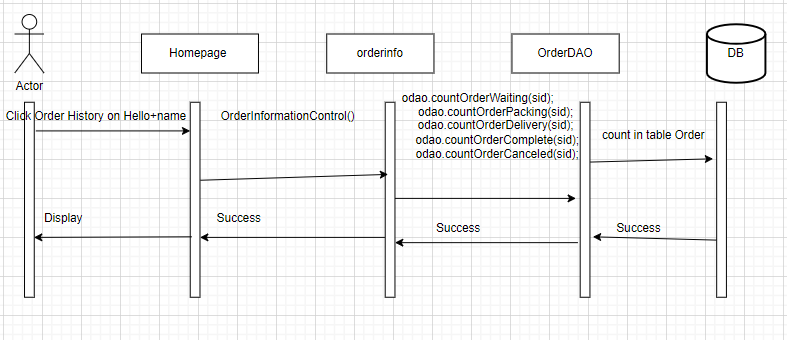


### b. Class Specifications

OrderDAO

| **No** | **Method** | **Description** |
| --- | --- | --- |
| 1  2  3  4  5 | countOrderWaiting(sid)  countOrderPacking(sid)  countOrderDelivery(sid)  countOrderComplete(sid)  countOrderCanceled(sid) | the number of product are waiting confirmation according by userId  the number of product are packing according by userId  the number of product are delivering according by userId  the number of product are completed according by userId  the number of product are canceled according by userId |

### c. Sequence Diagram(s)



### d. Database queries

select count(\*) from Orders o join Order\_Status os\n"

+ "on o.Status = os.ID\n"

+ "where o.UserID = ? and os.ID = 1

select count(\*) from Orders o join Order\_Status os\n"

+ "on o.Status = os.ID\n"

+ "where o.UserID = ? and os.ID = 2

select count(\*) from Orders o join Order\_Status os\n"

+ "on o.Status = os.ID\n"

+ "where o.UserID = ? and os.ID = 3

select count(\*) from Orders o join Order\_Status os\n"

+ "on o.Status = os.ID\n"

+ "where o.UserID = ? and os.ID = 4

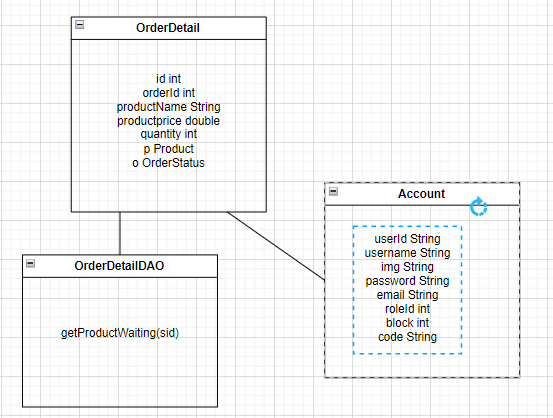
select count(\*) from Orders o join Order\_Status os\n"

+ "on o.Status = os.ID\n"

+ "where o.UserID = ? and os.ID = 5

## 40. View the product are waiting confirmation

## a. Class Diagram

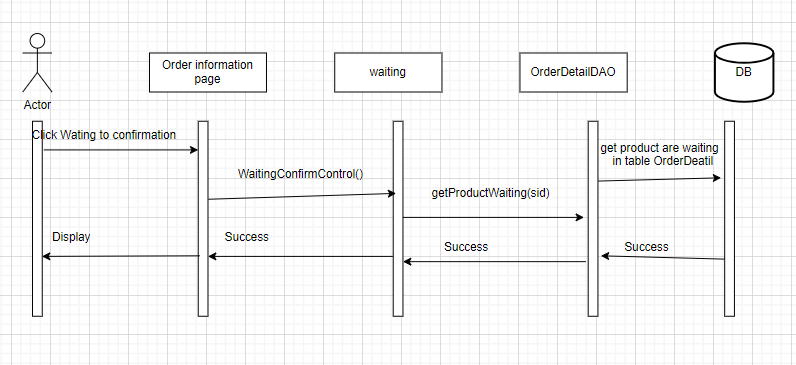


### b. Class Specifications

OrderDetailDAO

| **No** | **Method** | **Description** |
| --- | --- | --- |
| 1 | getProductWaiting(sid) | get product are Waiting confirmation according by userId |

### c. Sequence Diagram(s)



### d. Database queries

select p.ProductName,pr.ProductImgURL,p.OriginalPrice,p.SalePrice,o.Status from Order\_Detail d join Orders o\n"

+ "on d.OrderID = o.ID\n"

+ "join Order\_Status s on s.ID = o.status\n"

+ "join Users u on u.UserID = o.UserID\n"

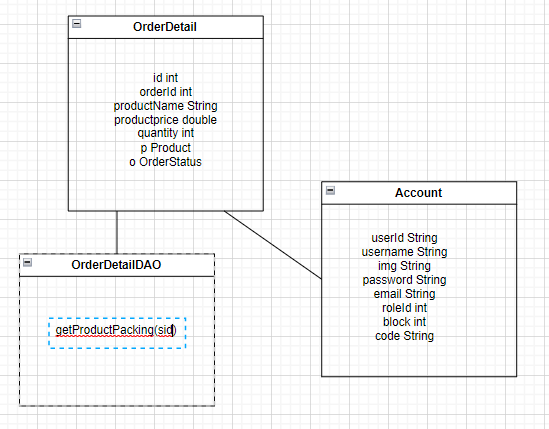
+ "join Product p on p.ProductID = d.ProductID\n"

+ "join ProductImg pr on pr.ProductID = p.ProductID\n"

+ "where u.UserID = ? and s.ID = 1

## 41. View the product are packing

## a. Class Diagram



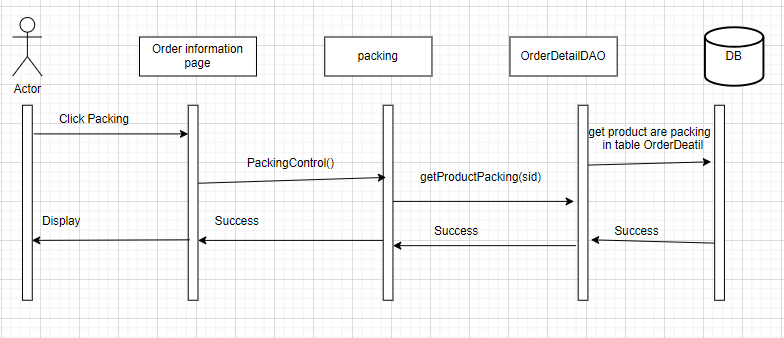
### b. Class Specifications

OrderDetailDAO

| **No** | **Method** | **Description** |
| --- | --- | --- |
| 1 | getProductPacking(sid) | get product are packing according by userId |

### 

### c. Sequence Diagram(s)



### d. Database queries

select p.ProductName,pr.ProductImgURL,p.OriginalPrice,p.SalePrice from Order\_Detail d join Orders o\n"

+ "on d.OrderID = o.ID\n"

+ "join Order\_Status s on s.ID = o.status\n"

+ "join Users u on u.UserID = o.UserID\n"

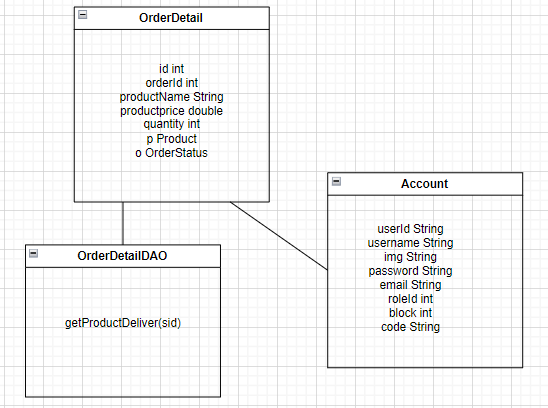
+ "join Product p on p.ProductID = d.ProductID\n"

+ "join ProductImg pr on pr.ProductID = p.ProductID\n"

+ "where u.UserID = ? and s.ID = 2

## 42. View the product are delivering

## a. Class Diagram



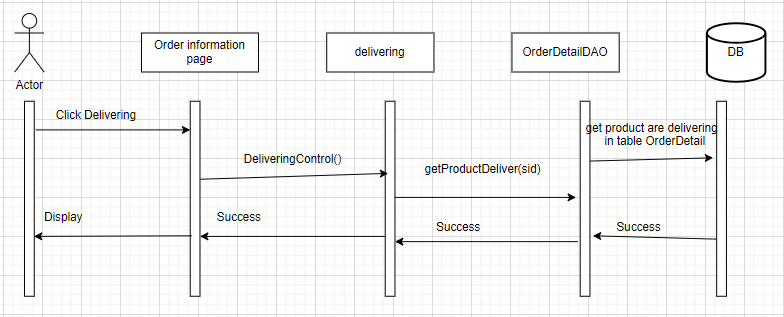
### b. Class Specifications

OrderDetailDAO

| **No** | **Method** | **Description** |
| --- | --- | --- |
| 1 | getProductDeliver(sid) | get product are packing according by userId |

### 

### c. Sequence Diagram(s)



### d. Database queries

select p.ProductName,pr.ProductImgURL,p.OriginalPrice,p.SalePrice from Order\_Detail d join Orders o\n"

+ "on d.OrderID = o.ID\n"

+ "join Order\_Status s on s.ID = o.status\n"

+ "join Users u on u.UserID = o.UserID\n"

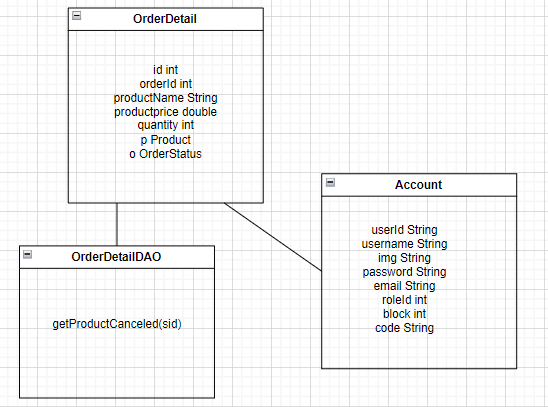
+ "join Product p on p.ProductID = d.ProductID\n"

+ "join ProductImg pr on pr.ProductID = p.ProductID\n"

+ "where u.UserID = ? and s.ID = 3

## 43. View the product are canceled

## a. Class Diagram



### b. Class Specifications

OrderDetailDAO

| **No** | **Method** | **Description** |
| --- | --- | --- |
| 1 | getProductCanceled(sid) | get product are canceled according by userId |

### c. Sequence Diagram(s)

### 

### d. Database queries

select p.ProductName,pr.ProductImgURL,p.OriginalPrice,p.SalePrice from Order\_Detail d join Orders o\n"

+ "on d.OrderID = o.ID\n"

+ "join Order\_Status s on s.ID = o.status\n"

+ "join Users u on u.UserID = o.UserID\n"

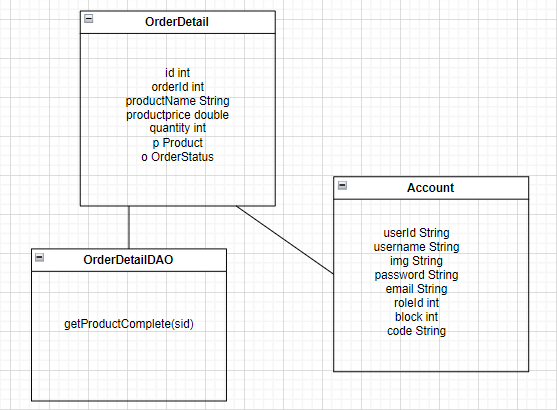
+ "join Product p on p.ProductID = d.ProductID\n"

+ "join ProductImg pr on pr.ProductID = p.ProductID\n"

+ "where u.UserID = ? and s.ID = 4

## 44. View the product are completed

## a. Class Diagram



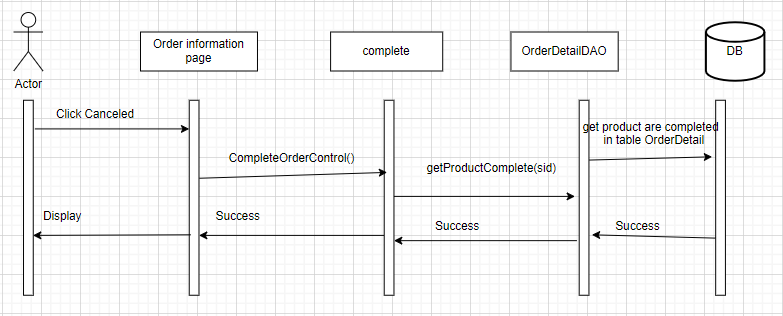
### b. Class Specifications

OrderDetailDAO

| **No** | **Method** | **Description** |
| --- | --- | --- |
| 1 | getProductComplete(sid) | get product are completed according by userId |

### 

### c. Sequence Diagram(s)



### d. Database queries

select p.ProductName,pr.ProductImgURL,p.OriginalPrice,p.SalePrice from Order\_Detail d join Orders o\n"

+ "on d.OrderID = o.ID\n"

+ "join Order\_Status s on s.ID = o.status\n"

+ "join Users u on u.UserID = o.UserID\n"

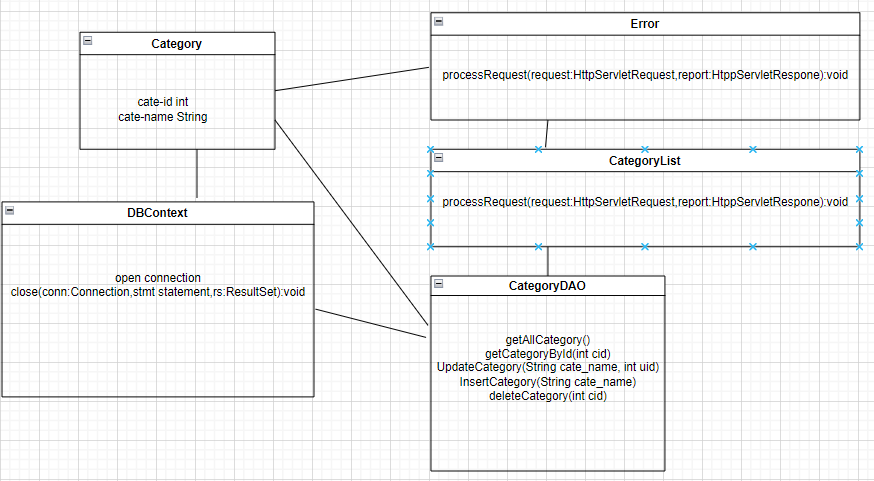
+ "join Product p on p.ProductID = d.ProductID\n"

+ "join ProductImg pr on pr.ProductID = p.ProductID\n"

+ "where u.UserID = ? and s.ID = 5

## 45. Manager Category

## a. Class Diagram



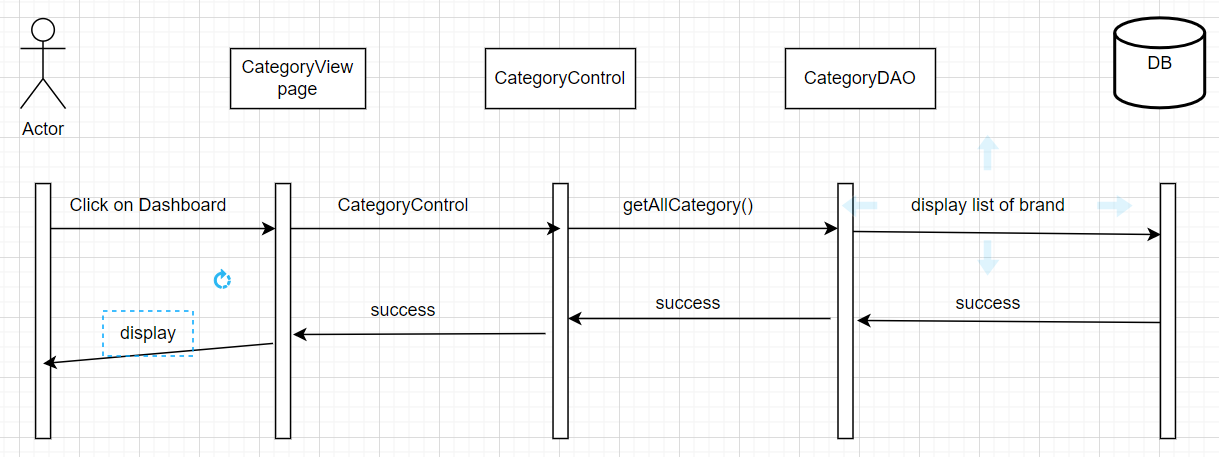
### b. Class Specifications

| **No** | **Method** | **Description** |
| --- | --- | --- |
| 01 | processRequest() | get brand list from database to display on BrandView.jsp |

- CategoryDAO class

| **No** | **Method** | **Description** |
| --- | --- | --- |
| 01 | getCategoryById(int cid) | Return: Category Detail where categoryid contain parameter  Input: ID of category |
| 02 | getAllCategory() | Return: All category in database |
| 03 | InsertCategory(String cate\_name) | Add category with detail entered in AddCategory.jsp page  Return: null  Input: category name |
| 04 | UpdateCategory(String cate\_name, int uid) | Update category with detail entered in EditCategory.jsp page  Return: null  Input: category name |
| 05 | deleteCategory(int cid) | Deletecategory with id entered in CategoryView.jsp  Return: null  Input: id |

### c. Sequence Diagram(s)

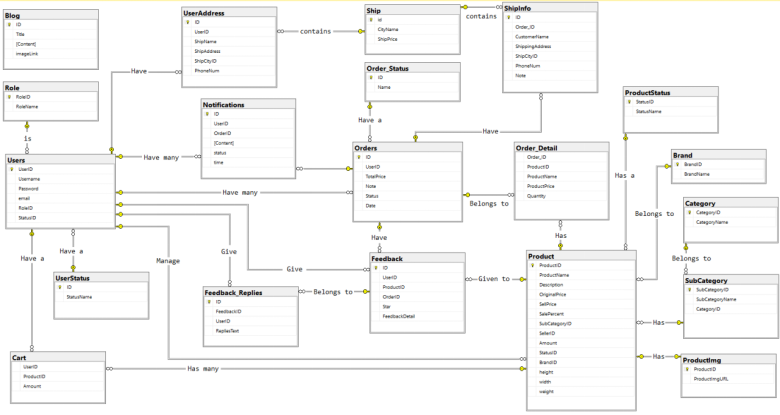


### d. Database queries

SELECT \* FROM dbo.Category

# 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* | *Comment* | *Status of a product*  *- Primary keys: CommentID*  *- Foreign keys: ProductID,* |
| *19* | *Brand* | *Brand data*  *- Primary keys: BrandID*  *- Foreign keys:* |
| *20* | *Cart* | *Cart info for each user*  *- Primary keys: UserID*  *- Foreign keys: ProductID* |