

**ONLINE SHOPPING SYSTEM**

**Software Design Document**

**(Version 0.4 – 03/10/2022)**

Member:

Trần Trung Việt – HE153481

Nguyễn Trung Hiếu – HE160326

Trần Công Minh – HE161354

Lê Sỹ Thái – HE160614

Vũ Hoàng Anh – HE163109

– Hanoi, August 2019 –

**Table of Contents**

[I. Overview 4](#_Toc115656872)

[1. Code Packages/Namespaces 4](#_Toc115656873)

[2. Coding Conventions 5](#_Toc115656874)

[II. Code Designs 5](#_Toc115656875)

[1. Login 5](#_Toc115656876)

[a. Class Diagram 5](#_Toc115656877)

[b. Class Specifications 5](#_Toc115656878)

[c. Sequence Diagram(s) 6](#_Toc115656879)

[d. Database queries 6](#_Toc115656880)

[2. Register 7](#_Toc115656881)

[a. Class Diagram 7](#_Toc115656882)

[b. Class Specifications 7](#_Toc115656883)

[c. Sequence Diagram(s) 8](#_Toc115656884)

[d. Database queries 8](#_Toc115656885)

[3. Feature Product 9](#_Toc115656886)

[a. Class Diagram 9](#_Toc115656887)

[b. Class Specification 10](#_Toc115656888)

[c. Sequence Diagram(s) 10](#_Toc115656889)

[d. Database queries 10](#_Toc115656890)

[4. View Product Detail 11](#_Toc115656891)

[a. Class Diragram 11](#_Toc115656892)

[b. Class Specification 11](#_Toc115656893)

[c. Sequence Diagram(s) 11](#_Toc115656894)

[d. Database queries 11](#_Toc115656895)

[5. Blog List 11](#_Toc115656896)

[a. Class Diagram 11](#_Toc115656897)

[b. Class Specification 12](#_Toc115656898)

[c. Sequence Diagram(s) 12](#_Toc115656899)

[d. Database queries 12](#_Toc115656900)

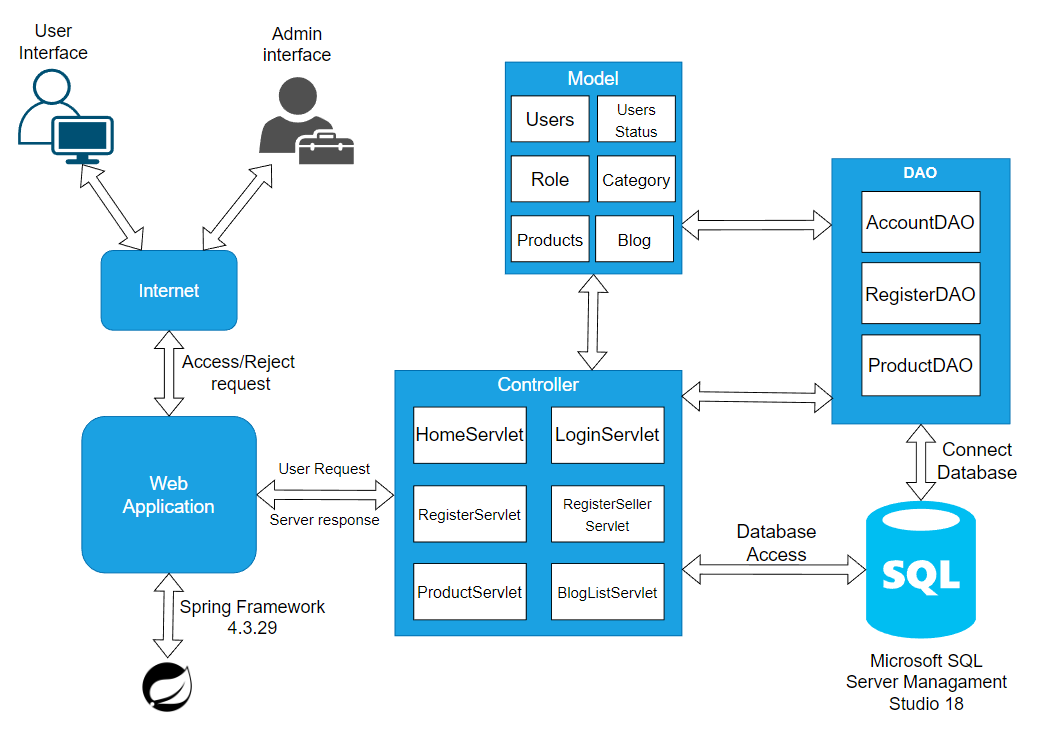
[III. Database Design 12](#_Toc115656901)

[1. Database Schema 12](#_Toc115656902)

[2. Table Description 13](#_Toc115656903)

# I. Overview

## 1. Code Packages/Namespaces



*Figure 1.1. Software Architecture Design*

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

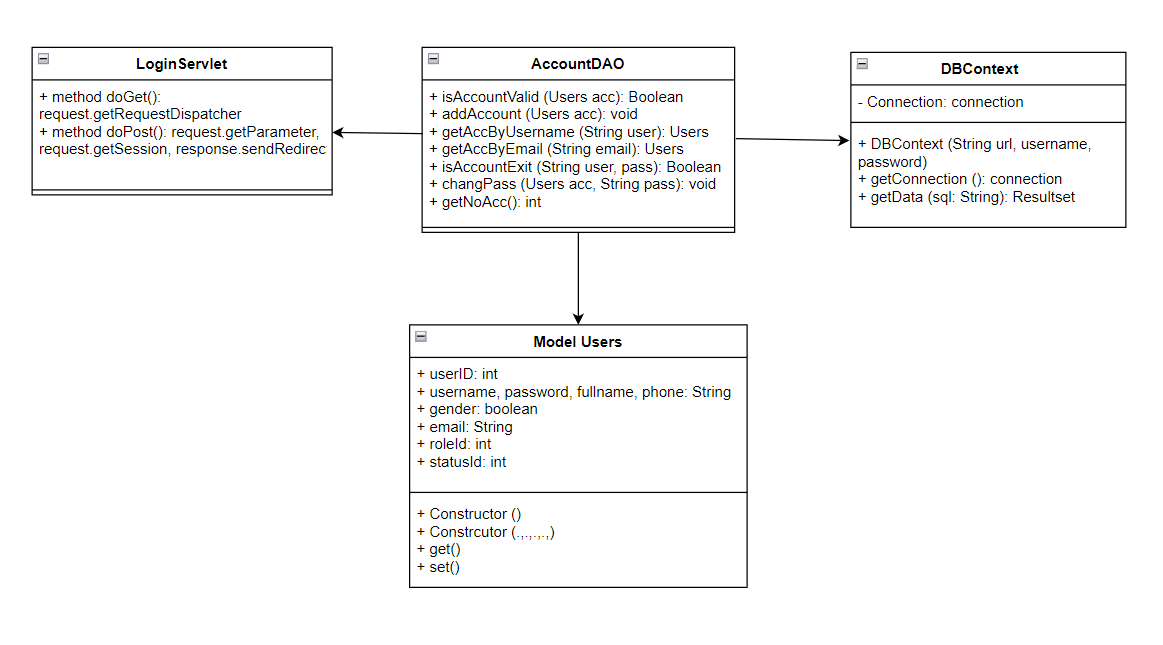
|  |  |  |
| --- | --- | --- |
| **No** | **Package** | **Description** |
| *1* | *View(Web Pages)* | *Provide interaction with user* |
| *2* | *Controller* | *Control the data flow between View (Frontend) and Model (DAO), control the logic of the system, Servlet naming convention* |
| *3* | *DAO* | *Provide the connection to the Database, Extract data for the system* |
| *4* | *Filter* | *Control the redirection between Pages in View (Frontend) and from View (Frontend) to Controller, Java Class naming convention* |
| *5* | *Model* | *Define the logic of data extracted from Database, Java Class naming convention* |
| *6* | *Context* | *Connect DBContext* |
| *7* | *Libraries* | *Contains libraries used in the project* |

## 2. Coding Conventions

# II. Code Designs

## 1. Login

### a. Class Diagram



*Figure 2.1.1. Class Login Diagram*

### b. Class Specifications

#### LoginServlet

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *processRequest()* | *Get all new information from user Form anf update the database* |
|  |  |  |

#### AdminDAO, Users

***Class Methods***

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *getAllAccount()* | *Get all user accounts list* |
| *02* | *getAccount(String username)* | *Get user account with username* |
| *03* | *Check(String username, String password)* | *Check if the account already exists* |
| *04* | *Public class User {},*  *Public user()*  *Getter(), setter(),* | *Create a class that contains the properties of the Users table* |

### c. Sequence Diagram(s)

****

*Figure 2.1.2. Sequence Diagram*

### d. Database queries

- **Select \* from Users;**

- **Select \* from Users where username = ?;**

**- select \* from Users where Username=? and Password=?";**

## 2. Register

### a. Class Diagram



*Figure 2.2.1. Class Register Diagram*

### b. Class Specifications

#### RegisterServlet

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *processRequest()* | *Get all new information from register Form anf update the database* |
|  |  |  |

#### AccountDAO, Users

***Class Methods***

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *checkAccountExits(String username)* | *Check if the username is already exits*  *Return null* |
| *02* | *void create(String username, String password, String fullname, String phone, String sex, String email)* | *Insert a new user with username, password, fullname, phone, gender, email.* |

### c. Sequence Diagram(s)

****

*Figure 2.2.2. Sequence Diagram*

### d. Database queries

**- Insert into Users values(?, ?, ?, ?, ?, ?, 3, 1);**

## 3. Feature Product

### a. Class Diagram



*Figure 2.3.1. Class Diagram*

### b. Class Specification

### c. Sequence Diagram(s)

**

*Figure 2.3.2 Sequence Diagram*

### d. Database queries

**- CategoryDAO: select \* from Category;**

**- SubCategoryDAO: SELECT [SubCategoryID], [SubCategoryName], [CategoryID] FROM [dbo].[SubCategory]**

**- ProductDAO:**

**+ getNewProducts:** SELECT TOP(15) \* FROM (SELECT p.ProductID,MIN(p.ProductName) AS ProductName,

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

MIN(p.SellPrice) AS SellPrice, MIN(p.SalePercent) AS SalePercent,

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

MIN(p.Amount) AS Amount, MIN(p.StatusID) AS StatusID, MIN(p.BrandID) AS BrandID,

MIN(ProI.ProductImgURL) AS ProductImgURL, MIN(Sub.CategoryID) AS CategoryID

FROM dbo.Product p JOIN dbo.ProductImg ProI ON ProI.ProductID = p.ProductID

JOIN dbo.SubCategory Sub ON Sub.SubCategoryID = p.SubCategoryID

GROUP BY p.ProductID ) t

ORDER BY t.ProductID DESC

**+ getProductsByCid(int cid, int sid, int sortType, int sortMode):** SELECT \* FROM (SELECT p.ProductID,MIN(p.ProductName) AS ProductName,

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

MIN(p.SellPrice) AS SellPrice, MIN(p.SalePercent) AS SalePercent,

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

MIN(p.Amount) AS Amount, MIN(p.StatusID) AS StatusID, MIN(p.BrandID) AS BrandID,

MIN(ProI.ProductImgURL) AS ProductImgURL, MIN(Sub.CategoryID) AS CategoryID

FROM dbo.Product p JOIN dbo.ProductImg ProI ON ProI.ProductID = p.ProductID

JOIN dbo.SubCategory Sub ON Sub.SubCategoryID = p.SubCategoryID

WHERE 1=1 AND p.StatusID!= 2 AND p.Amount>0 AND Sub.CategoryID= ?

AND Sub.SubCategoryID= ?

GROUP BY p.ProductID) t

ORDER BY t.SellPrice, t.SalePercent, t.ProductName ASC

**+ getProductById(int id):** SELECT \* FROM (SELECT p.ProductID,MIN(p.ProductName) AS ProductName,

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

MIN(p.SellPrice) AS SellPrice, MIN(p.SalePercent) AS SalePercent,

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

MIN(p.Amount) AS Amount, MIN(p.StatusID) AS StatusID, MIN(p.BrandID) AS BrandID,

MIN(ProI.ProductImgURL) AS ProductImgURL, MIN(Sub.CategoryID) AS CategoryID

FROM dbo.Product p JOIN dbo.ProductImg ProI ON ProI.ProductID = p.ProductID

JOIN dbo.SubCategory Sub ON Sub.SubCategoryID = p.SubCategoryID

GROUP BY p.ProductID ) t where t.ProductID = ?

## 4. View Product Detail

### a. Class Diragram



*Figure 2.4.1 Class Diagram*

### b. Class Specification

### c. Sequence Diagram(s)



*Figure 2.4.2 Sequence Specification*

### d. Database queries

## 5. Blog List

### a. Class Diagram



*Figure 2.5.1 Class Diagram*

### b. Class Specification

### c. Sequence Diagram(s)



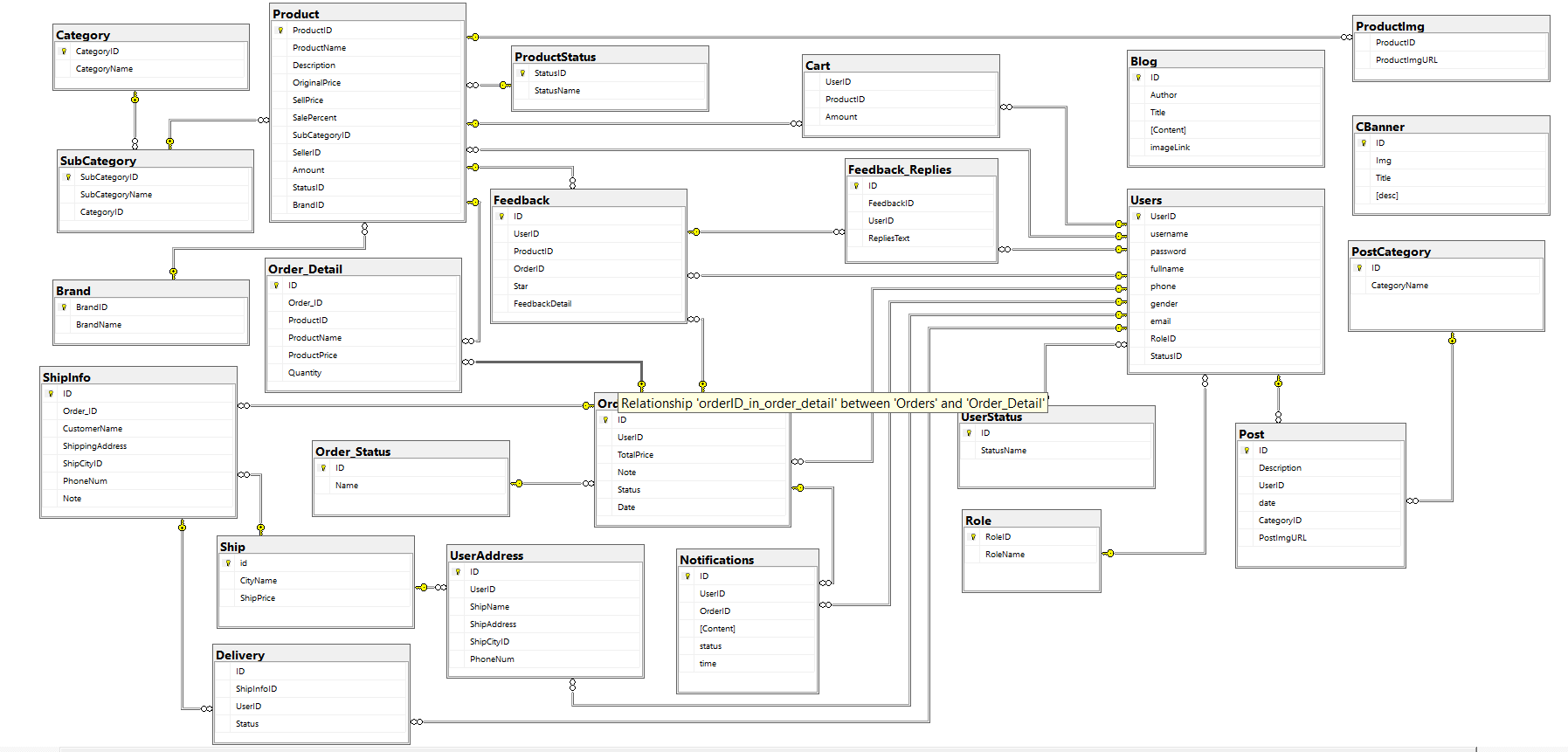
*Figure 2.5.2 Sequence Diagram*

### d. Database queries

**+ getAllBlogs(): select \* from Blogs;**

# III. Database Design

## 1. Database Schema



*Figure 3.1. Database diagram*

## 2. Table Description

|  |  |  |
| --- | --- | --- |
| **No** | **Table** | **Description** |
| 1 | UserStatus | Account activation status  - Primary keys: ID  - Foreign keys: none |
| 2 | Role | Role of the user (Admin, Seller, Customer, Shipper)  - Primary keys: RoleID  - Foreign keys: none |
| 3 | User | User  - Primary keys: UserID  - Foreign keys: RoleID, StatusID |
| 4 | Ship | <Description of the table>  - Primary keys: ID  - Foreign keys: none |
| 5 | UserAddress | Address for each user  - Primary keys: ID  - Foreign keys: UserID, ShipCityID |
| 6 | Category | <Description of the table>  - Primary keys: CategoryID  - Foreign keys: none |
| 7 | SubCategory | Sub Categories  - Primary keys: SubCategoryID  - Foreign keys: CategoryID |
| 8 | ProductStatus | Status of a product  - Primary keys: StatusID  - Foreign keys: none |
| 9 | Brand | Brand data  - Primary keys: BrandID  - Foreign keys: none |
| 10 | Product | Product data  - Primary keys: ProductID  - Foreign keys: SubCategoryID, SellerID, StatusID, BrandID |
| 11 | ProductImg | Image of products  - Primary keys: none  - Foreign keys: ProductID |
| 12 | Cart | Cart info for each user  - Primary keys: UserID  - Foreign keys: ProductID |
| 13 | Order\_Status | Status of a order  - Primary keys: ID  - Foreign keys: none |
| 14 | Order | Order Data  - Primary keys: ID  - Foreign keys: UserID,Status |
| 15 | Order\_Detail | <Description of the table>  - Primary keys: none  - Foreign keys: OrderID, ProductID |
| 16 | ShipInfo | Shipping info for each order  - Primary keys: ID  - Foreign keys: Order\_ID, ShipCityID |
| 17 | Notifications | Notification data  - Primary keys: ID  - Foreign keys: OrderID, UserID |
| 18 | Feedback | Feedback of a product order  - Primary keys: ID  - Foreign keys: UserID,ProductID, OrderID |
| 19 | Feedback\_Replies | Replies of feedbacks  - Primary keys: id  - Foreign keys: FeedbackID,UserID |
| 20 | Blog | Blog content + image  - Primary keys: ID  - Foreign keys: |
| 21 | BlogDetail | Detailed content of the blog + image  -Primary key: BlogDetailID  -Foreign key: BlogID |
| 22 | PostCategory | <Description of the table>  - Primary keys: ID  - Foreign keys: none |
| 23 | Post | <Description of the table>  - Primary keys: ID  - Foreign keys: UserID, CategoryID |
| 24 | CBanner | Carousel Banner  - Primary keys: BannerID  - Foreign keys: none |
| 25 | Delivery | Delevery methos  - Primary keys: ID  - Foreign keys: ShipInfo, UserID |