

**ONLINE SHOPPING SYSTEM**

**Software Design Document**

**(Version 0.2 – 19/09/2022)**

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 3](#_Toc114517413)

[1. Code Packages/Namespaces 3](#_Toc114517414)

[2. Coding Conventions 4](#_Toc114517415)

[II. Code Designs 4](#_Toc114517416)

[1. Login 4](#_Toc114517417)

[a. Class Diagram 4](#_Toc114517418)

[b. Class Specifications 4](#_Toc114517419)

[c. Sequence Diagram(s) 4](#_Toc114517420)

[d. Database queries 5](#_Toc114517421)

[2. Register 6](#_Toc114517422)

[a. Class Diagram 6](#_Toc114517423)

[b. Class Specifications 6](#_Toc114517424)

[c. Sequence Diagram(s) 7](#_Toc114517425)

[d. Database queries 7](#_Toc114517426)

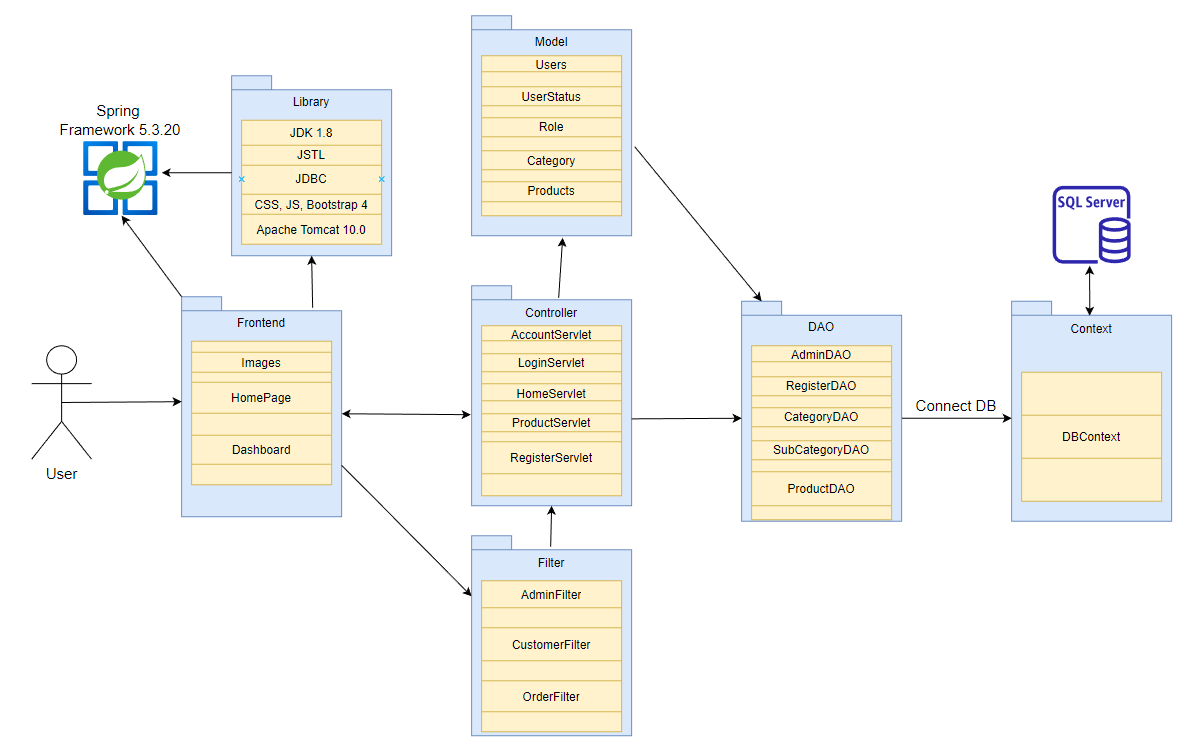
[III. Database Design 8](#_Toc114517427)

[1. Database Schema 8](#_Toc114517428)

[2. Table Description 8](#_Toc114517429)

# I. Overview

## 1. Code Packages/Namespaces



*Figure 1.1. Software Architecture Design*

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

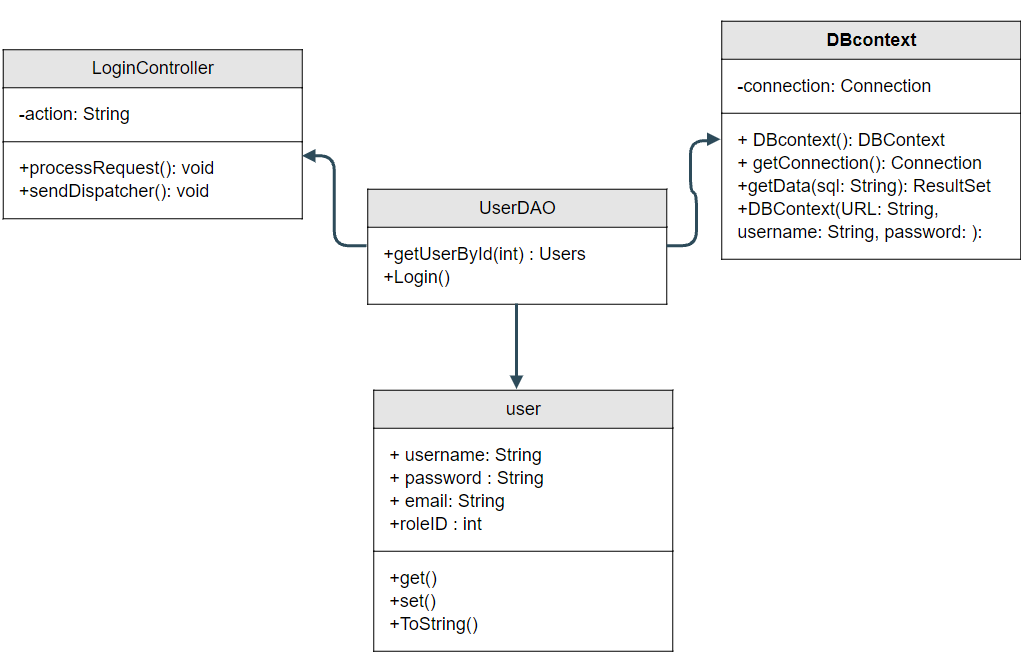
|  |  |  |
| --- | --- | --- |
| **No** | **Package** | **Description** |
| *1* | *View(Web Pages)* | *Provide interaction with user* |
| *2* | *Controller* | *Control the data flow between View (Frontend) and Model (DAO), control the logic of the system, Servlet naming convention* |
| *3* | *DAO* | *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

***Class Methods***

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *01* | *getAllAccount()* | *Get all user accounts list* |
| *02* | *getAccount(String username)* | *Get user account with username* |

### c. Sequence Diagram(s)

****

*Figure 2.1.2. Sequence Diagram*

### d. Database queries

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

- **Select \* from Users;**

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

## 2. Register

### a. Class Diagram



*Figure 2.1.1. Class Register Diagram*

### b. Class Specifications

#### LoginServlet

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

#### RegisterDAO

***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)

****

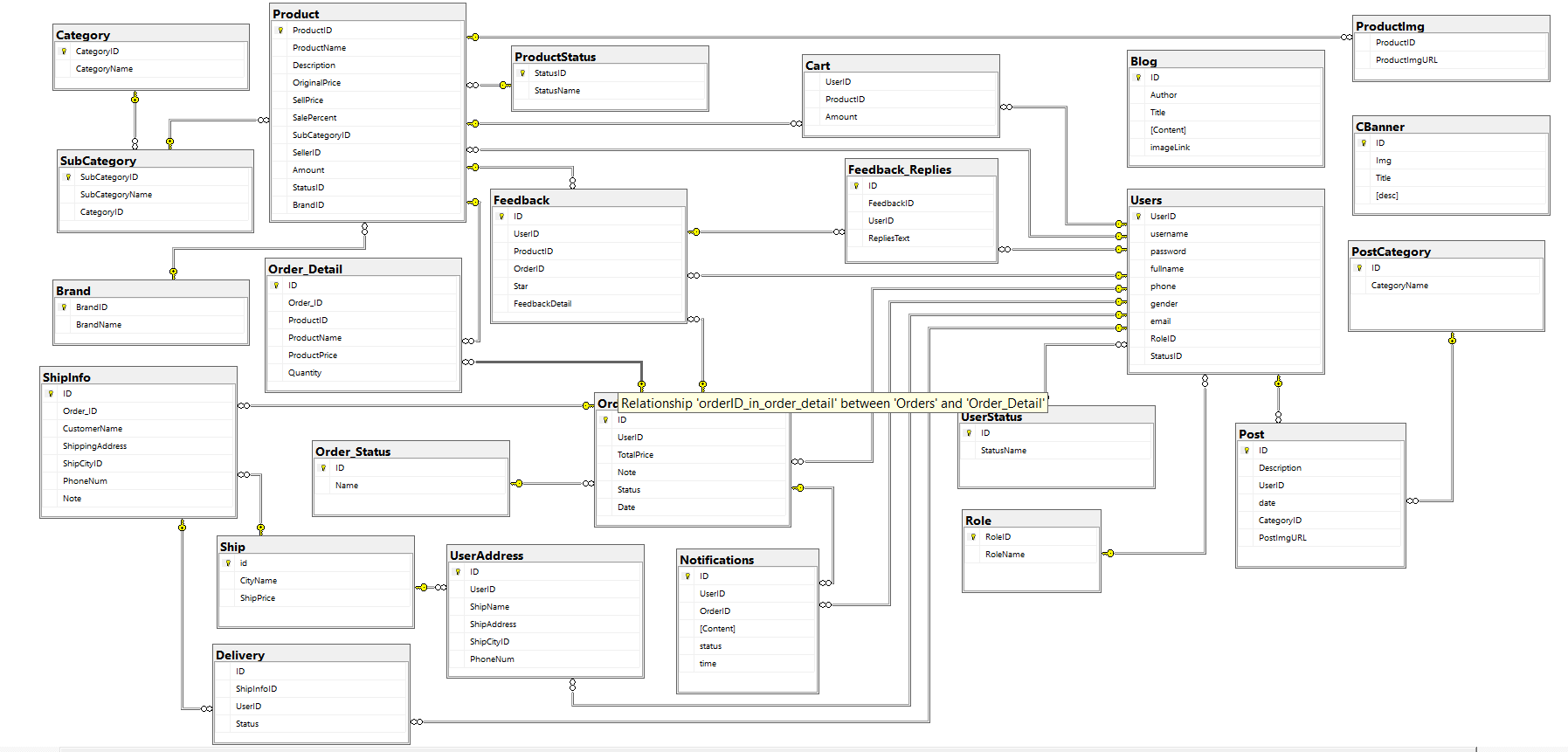
*Figure 2.1.2. Sequence Diagram*

### d. Database queries

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

# 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 | PostCategory | <Description of the table>  - Primary keys: ID  - Foreign keys: none |
| 22 | Post | <Description of the table>  - Primary keys: ID  - Foreign keys: UserID, CategoryID |
| 23 | CBanner | Carousel Banner  - Primary keys: BannerID  - Foreign keys: none |
| 24 | Delivery | Delevery methos  - Primary keys: ID  - Foreign keys: ShipInfo, UserID |