

**CLOTHING SHOP PROJECT REPORT**

**Software Design Document**

– Hanoi, Summer 2022 –

**Table of Contents**

[I. Overview](#_gjdgxs) **3**

[1. Code Packages/Namespaces](#_30j0zll) 3

[2. Coding Conventions](#_1fob9te) 4

[II. Code Designs](#_3znysh7) **4**

[1. <Feature/Function Name1>](#_2et92p0) 4

[a. Class Diagram](#_tyjcwt) 4

[b. Class Specifications](#_3dy6vkm) 4

[XYZ Class](#_tiif54cjisgr) 4

[ABC Class](#_jvylgokokbji) 5

[c. Sequence Diagram(s)](#_1t3h5sf) 5

[d. Database queries](#_4d34og8) 5

[2. Login](#_2s8eyo1) 6

[Class Diagram](#_7l39didldxl3) 6

[Class Specifications](#_x448n0z5y6v0) 6

[Sequence Diagram(s)](#_xvl8itot5ygb) 7

[Database queries](#_59zu3h7gxaiw) 7

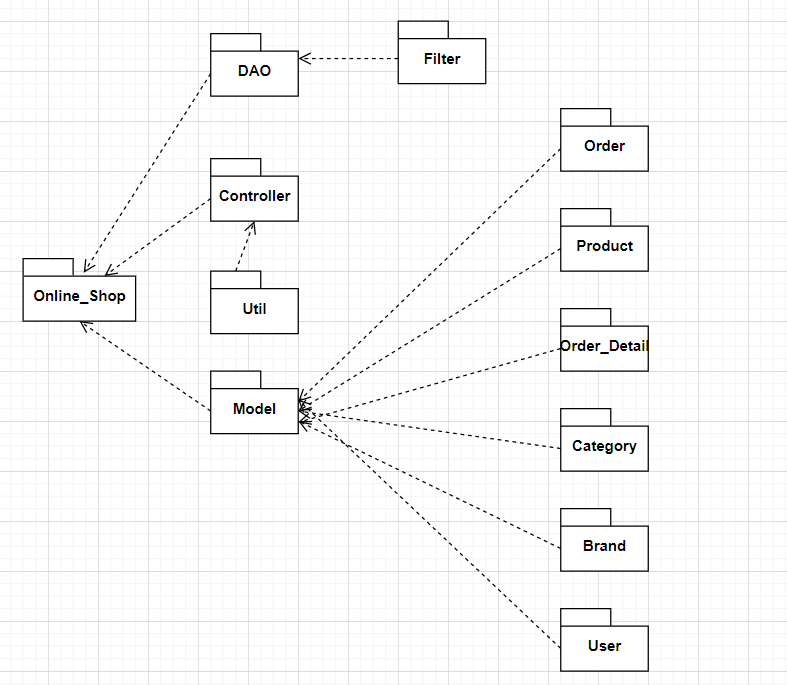
[III. Database Design](#_17dp8vu) **8**

[1. Database Schema](#_9mv5o6wjk5xw) 8

[2. Table Description](#_kom4gjfr82sg) 8

# 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>

### a. Class Diagram

### b. Class Specifications

#### XYZ Class

*[Provide the detailed description for the class methods]*

| **No** | **Method** | **Description** |
| --- | --- | --- |
| *01* | *<method name>* | *<Description of the method, including the inputs, outputs & internal method processing>* |
|  |  |  |

#### ABC Class

***Class Methods***

*[Provide the detailed description for the class methods]*

| **No** | **Method** | **Description** |
| --- | --- | --- |
| *01* | *<method name>* | *<Description of the method, including the inputs, outputs & internal method processing>* |
|  |  |  |

### c. Sequence Diagram(s)

*[Provide the sequence diagram(s) for the feature, see the sample below]*

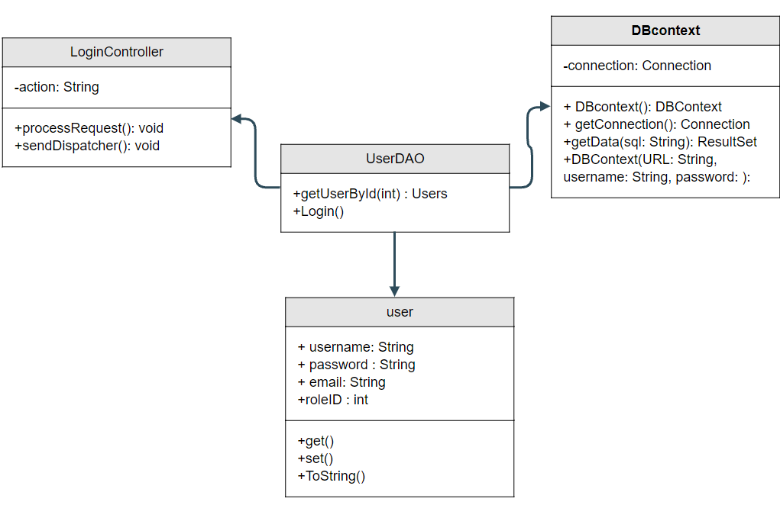


### d. Database queries

*[Provide the detailed SQL (select, insert, update...) which are used in implementing the function/screen]*

## 2. Login

### Class Diagram



### Class Specifications

UserController

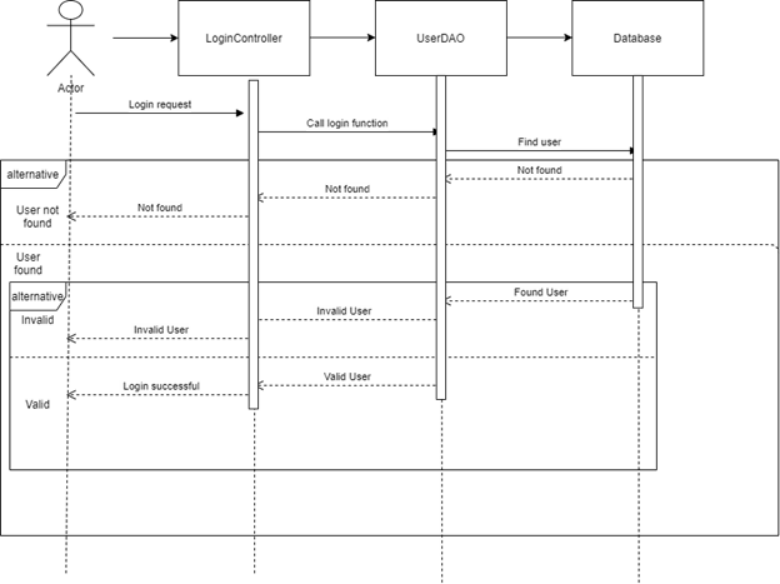
| No | Method | Description |
| --- | --- | --- |
| 01 | processRequest() | get all new information from user Form and update the database |

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. Homepage

### Class Diagram

### Class Specifications

UserController

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

UserDAO

[Provide the detailed description for the class methods]

| No | Method | Description |
| --- | --- | --- |
| 01 |  |  |
| 02 |  |  |

### 

### c. Sequence Diagram(s)

### 

### d. Database queries

## 4. Product detail

### Class Diagram

### Class Specifications

UserController

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

UserDAO

[Provide the detailed description for the class methods]

| No | Method | Description |
| --- | --- | --- |
| 01 |  |  |
| 02 |  |  |

### 

### c. Sequence Diagram(s)

### 

### d. Database queries

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

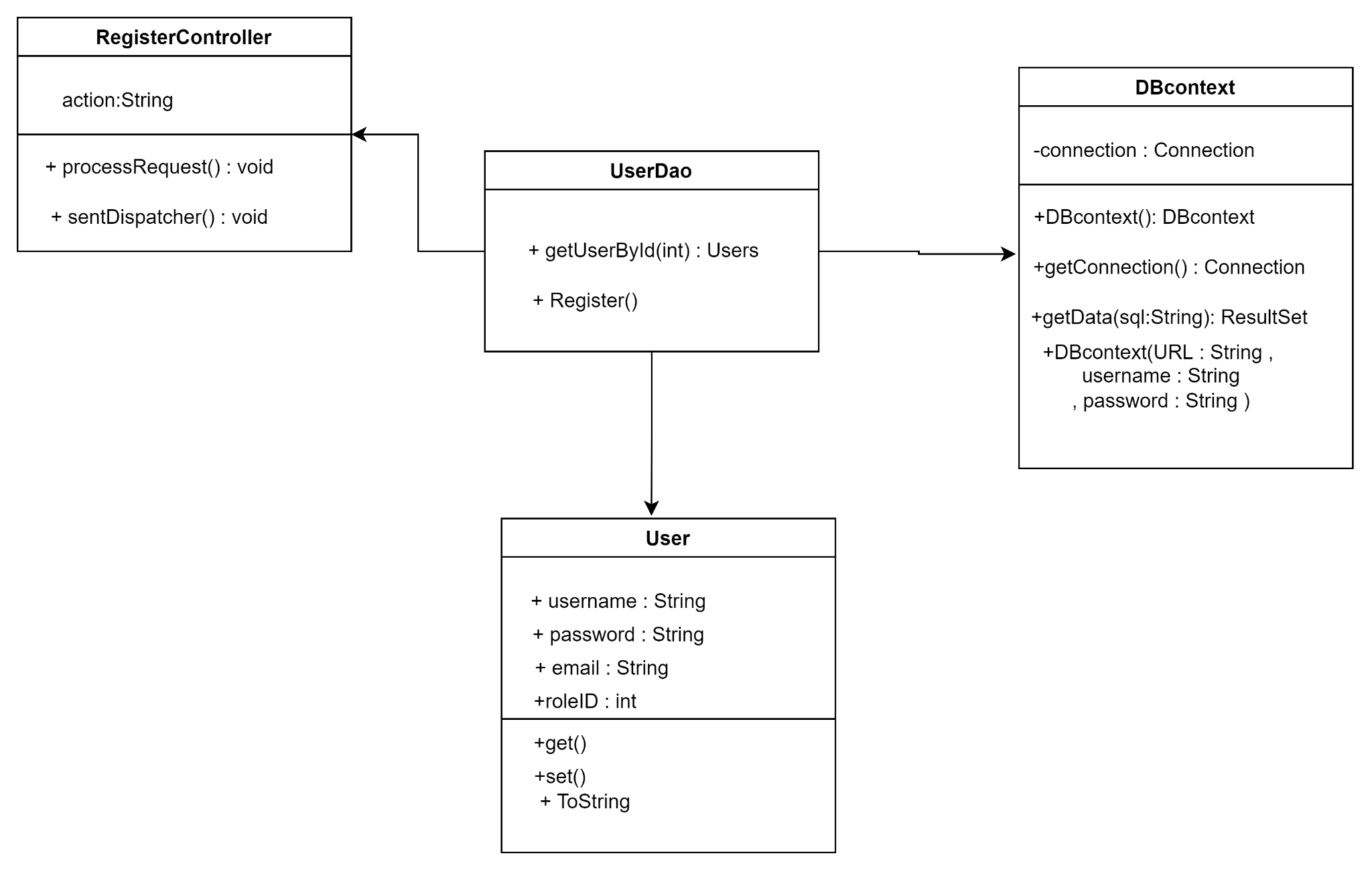
+ "on p.SubCategoryID = s.SubCategoryID\n"

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

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

## 5. Register

**a. Class Diagram**



### Class Specifications

UserController

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

UserDAO

[Provide the detailed description for the class methods]

| No | Method | Description |
| --- | --- | --- |
| 01 |  |  |
| 02 |  |  |

### 

### c. Sequence Diagram(s)

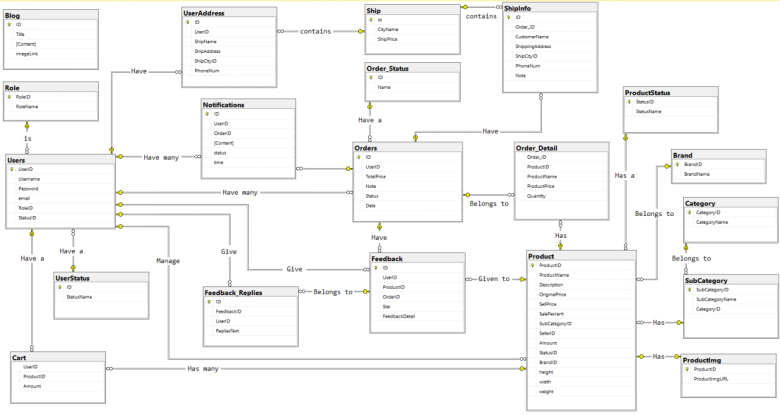
### 

### d. Database queries

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

# 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* |