**SPORTY SHOES - E.COMM STORE**

​Source Code

# Presented By : Ajunumasa Sadanandan

## Git hub link : <https://github.com/ajunu/Phase3FinalProject/tree/master/phase3-master>

# TABLE OF CONTENTS

|  |  |  |
| --- | --- | --- |
| S.No | Topic | Page No. |
| 1 | Source Code Access - GitHub URL | 1 |
| 2 | Source Code - Key Technologies used | 1 |
| 3 | Code/Algorithm Explanation | 2 |
| 4 | Front End Implementation | 8 |
| 5 | Backend Code Structure | 8 |
| 6 | Conclusion | 10 |

# 1. SOURCE CODE ACCESS

The source code is pushed to GitHub repository. Please find the URL below for source code access:

***https://github.com/haritaToboso/SimpliLearn\_P3\_EcommerceShoeStore***

More details on setup and installation and how to run code can be found under README.md document in the repository.​

# 2. SOURCE CODE - KEY TECHNOLOGIES USED

The application backend is built completely using ​**Spring ​**concepts along with the critical

**Core Java​** concepts.

Key concepts and Technologies implemented are:

1. JSP
2. Servlets
3. Hibernate

Along with this, following critical Java concepts are also used:

1. Collections
2. Exception Handling
3. Sorting
4. Looping & Control Flow - switch, if..else

The application front end is developed using:

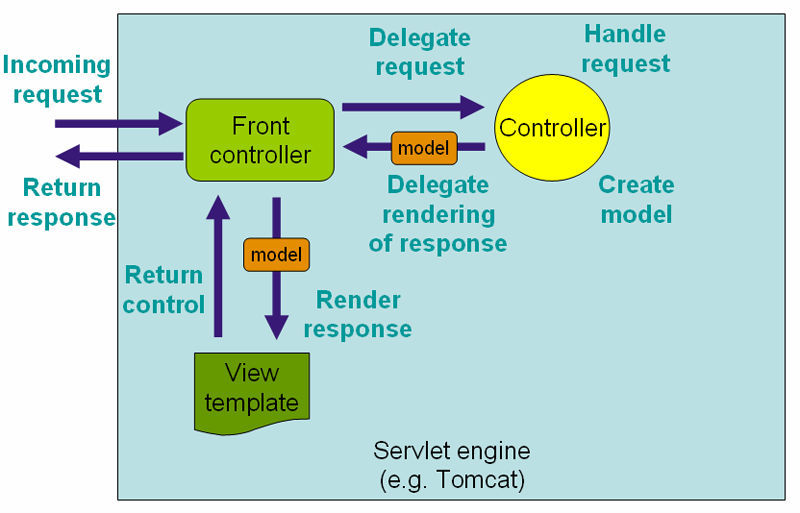
1. HTML 5
2. Bootstrap
3. Javascript

For Database, ​**MySQL ​**is selected. ​**Hibernate ​**is used as the ORM tool. The program is developed using ​**STS ​**as IDE along with ​**Maven ​**Spring MVC archetype.

The application is hosted using a local server provided by Apache ​**Tomcat​**. Detailed algorithm explanation is given in the following sections.

# 3. CODE/ALGORITHM EXPLANATION

The e-commerce application is developed adopting an MVC Architecture. The database connection and interactions are coded adopting a DAO pattern.



The overall application flow OR code structure is divided into phases like this:

1. The user interacts with application via HTML/JSP pages (​**VIEW​**)
2. The request along with parameters,session attributes are sent to Servlets acting as **Controllers​**.
3. The servlet redirects the request to appropriate methods in DBUtil classes containing DB Codes.
4. The result is send back to controller servlet
5. The response is given back to the user in the HTML/JSP Page.

Few other concepts that are implemented are:

* 1. **HTTP Sessions ​**- for tracking user login status, and logged in user details

* 1. **POJO ​**- for storing entities like user details, product Details and purchase Details etc.

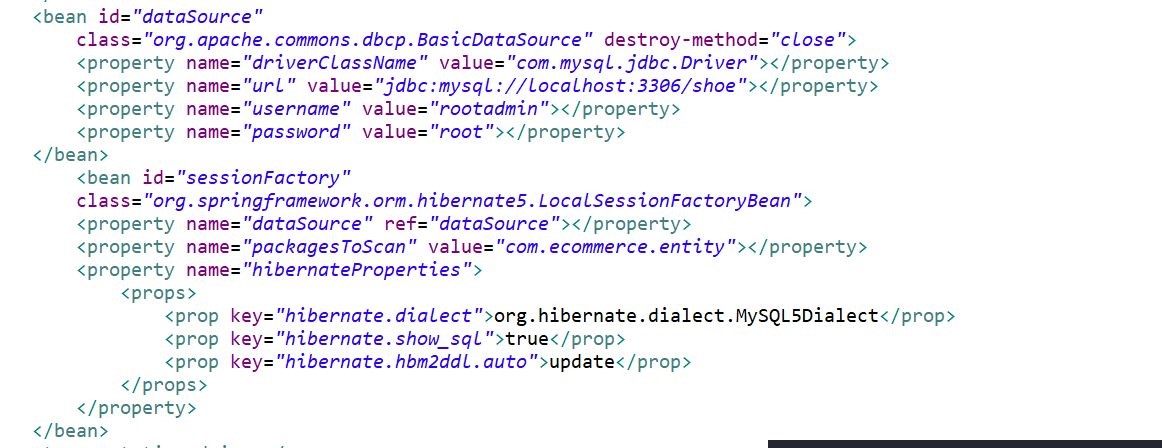


## DATABASE SETUP

In MySQL the DB and appropriate tables are created for storing and maintaining data about users,products and purchase details using Hibernate queries automatically.

The DB is connected to java codes using configurations is done using ​**spring-servlet.xml**

file.



# USER VIEW

Requirements:

Users interact with the service using this front end website. The demanded specifications are as follows:

1. Users should be able to login or register to the portal.
2. Users should be displayed with all the available products and make a purchase.
3. Order details should be accessible to the user.

User Entry Point:

The user can enter the application via HOME page as explained in the flowchart.

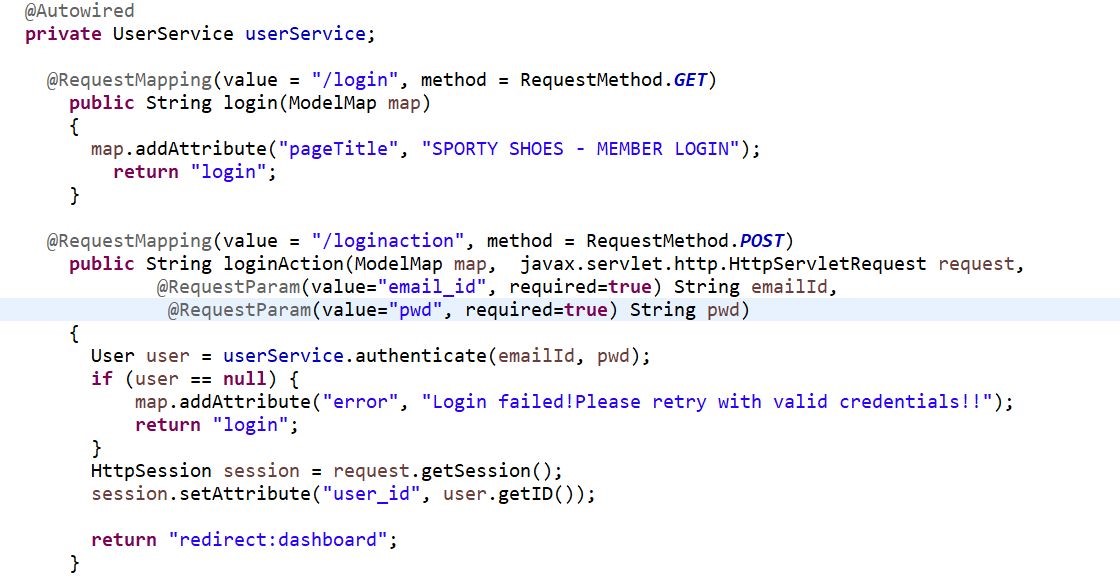
***a) Welcome Page:***

From this page, the user is directly redirected to Login Page. Users can choose any of the 3 methods to login.

* 1. Login as existing user
  2. Register as new user
  3. Login as admin

Based on the selection, the user can proceed with successful landing on the appropriate home page, after credential validations.

* + - If credentials are correct, the user is redirected to the home page using Page Redirection method.
    - Else, an appropriate error message is displayed by adding the message as attribute and reading in the JSP page.



***b) ​Home Page:***

Users are presented with all the available products. This is achieved by collecting all product details from DB as a ​**LIST ​**through hibernate queries.



## 2. Make Purchase

When a user tries to add a product to cart, session validation is done to check whether the user is logged in or not. If not, the user is prompted to login to continue. Then user can proceed with the checkout.

## 4. Confirmation Page

The purchase item details are stored in the session. And the user can proceed to make payment after verifying the details.

## 5. Payment Gateway

This is a purely dummy page, created using HTML and CSS to mimic payment gateways. **Users are displayed with the total price​** . After making payment, the user is displayed with a Payment Successful/Order Confirmartionpage. From here, user can either:

1. View his order details (redirecting to “View Orders” page) OR
2. Go back to Home page

## 6. My Profile

This page is accessible via Navigation Bar on top of every page in the application. From here, user can:

1. View his/her profile personal information details
2. View previous order history

All these are done using interacting with Controller Servlet and DB codes.

## 7. Logout

At this point, the user is prompted with an alert, when she/he clicks on LOGOUT option, displayed in Navigation Bar on top of every page in the application.

Once the confirmation is given, the user is redirected to the Logout page, and the ​**session is invalidated.**

The user needs to login again if he wishes to continue to use the application

**ADMIN BACKEND:**

Requirements:

**The admin should be able to change his password if he wants, he should be able to:**

* Manage the products in the store including categorizing them
* Browse the list of users who have signed up and be able to search users ● See purchase reports filtered by date and category

## 1. Login Page

The admin can login providing the credentials, which will be validated against the data stored in DB. After successful validation, admin can land on the Admin Dashboard home page.

## 2. Home Page

The user is presented with options to view the Master list of required data - product ,categories,user details, purchase report. When admin clicks on a tab, REQUEST is made, the DB code returns the data as ​**LIST​. ​**This is then displayed in the UI page.

## 3. Change Password

Admin is given a “ Change Password” tab in the navigation bar. After clicking on this, he is provided with the option to change password.

User is displayed with appropriate messages.User is navigated to HOME page and can verify the new password by trying to login again

## 4. Logout

The functionality is implemented similar to User logout. Session invalidation happens here and the user must login again to continue.

# 4. FRONT END IMPLEMENTATION

JSP pages are used for front end development. Bootstrap is used for styling.

The​ **JSTL tags​** are used for ease of coding.

Headers & Footers were kept in separate files and then included in every page.



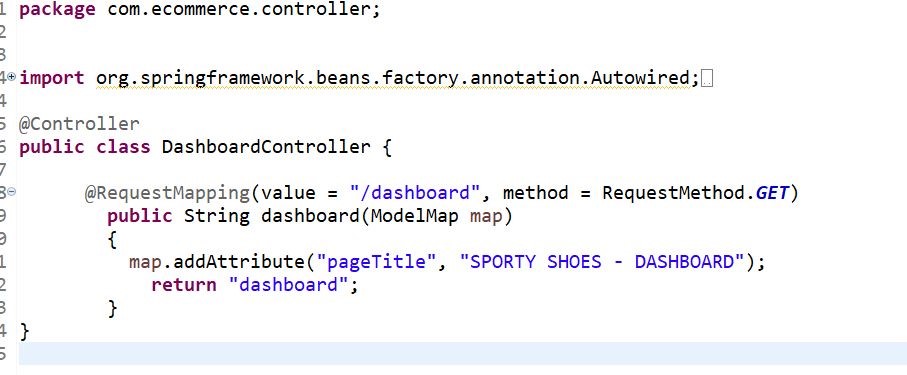
# 5. BACKEND CODE STRUCTURE

The ​**DispatcherServlet ​**delegates the request to the controllers to execute the functionality specific to it. The ​**@Controller​** annotation indicates that a particular class serves the role of a controller. The ​**@RequestMapping​** annotation is used to map a URL to either an entire class or a particular handler method.

Servlets:

There are 6 major controller servlets developed.

|  |  |
| --- | --- |
| Servlet Name | Purpose |
| AdminController | For admin related REQUESTS like viewing master lists and change password etc. |
| MemberController | For REQUESTS related to login,logout,sign up etc. |
| HomeController | For viewing home page with all associated Nav bar and links and product details table |
| DashboardController | For viewing dashboard. |
| CartController | For adding or deleting items from cart and perform checkout related REQUESTS |
| PurchaseController | For purchase and payment related REQUESTS |



*Eg: DashboardController*

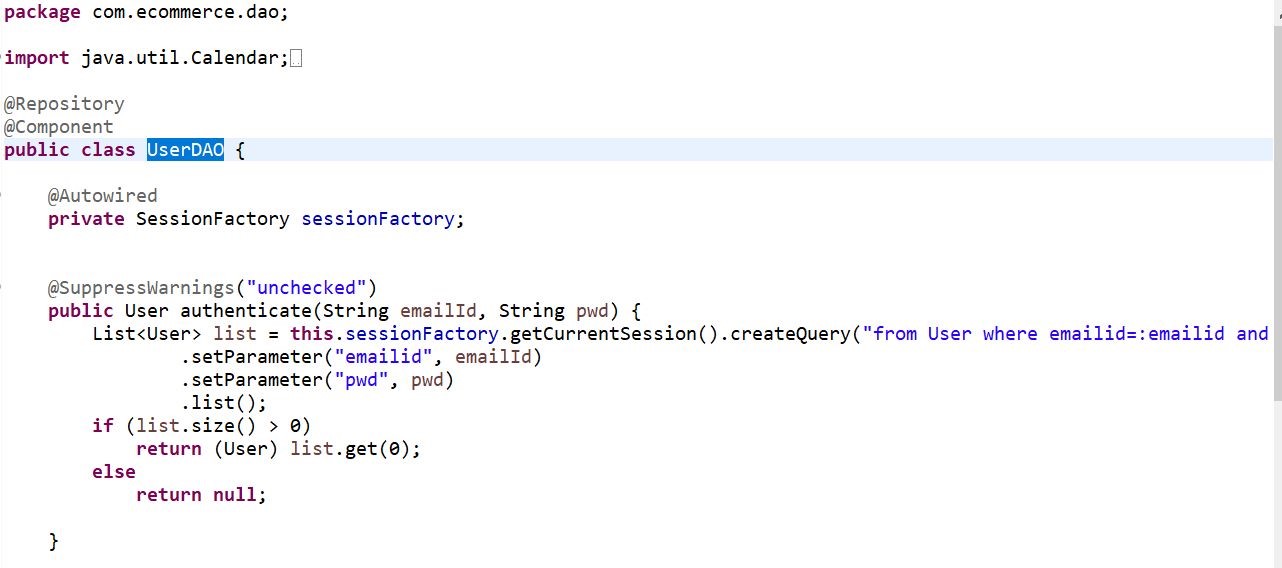
The controllers make use of ​**Services ​**to access ​**DAO​**. From the Service, the corresponding DAO java class is called and executes the query. The result is then passed back to the calling method and the user is redirected to an appropriate JSP page displaying REQUEST’s RESPONSE.

  *Eg: AdminService*

DAO:

There are 6 major DAO developed.

|  |  |
| --- | --- |
| **DAO Name** | **Purpose** |
| UserDAO | For queries related to user login validations |
| AdminDAO | For queries related to admin control panel |
| CategoryDAO | For queries related to category |
| ProductDAO | For queries related to product |
| PurchaseDAO | For queries related to purchase details |
| PurchaseItemDAO | For queries related to purchase item |



**​**

*​*

**​**

*Fig: A method inside DAO*

This is the overall code structure and algorithm used in the project.

# 6. CONCLUSION

The application is designed for the Sporty Shoes E-commerce Portal application using Java Spring Concepts. Along with Hibernate as an ORM tool,Data Structures and algorithms are used for retrieving data from Database to generate RESPONSE. A minimal UI is developed using HTML5 and Bootstrap. The program is stable and implements all functionalities as per requirement.