FPT Software

JAVA SPRING FRAMEWORK

Lab Guides

Document Code	25e-BM/HR/HDCV/FSOFT
Version	1.0
Effective Date	01/09/2024

Hanoi, 08/2024

RECORD OF CHANGES

No	Effective Date	Change Description	Reason	Reviewer	Approver
1	06/08/2024	Create a new Lab	Create new		VinhNV

Contents

CODE:	JSFW_Lab_07_Opt1
TYPE:	SHORT
LOC:	200
DURATION:	120 MINUTES

Java Spring Framework Introduction

Objectives:

- Learn to use Spring Boot annotations such as @Controller, @Service, and @Repository.
- Implement the login and signup functionality using MVC architecture.
- Understand form handling in Spring Boot using Thymeleaf.

Lab Specifications:

- Trainees will implement login and signup functionality. The project will include controller methods, services, and repository interactions for user authentication.

Problem Description:

Trainees are required to:

- Implement user signup functionality.
- Implement user login.
- Use Thymeleaf for the user interface (UI).

Prerequisites:

- Completed **JSFW_Lab_06_Opt2**.

Guidelines:

Step 1: Update pom.xml with these denpendencies:

```
<dependencies>
  <dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-web</artifactId>
  </dependency>

  <!-- Spring Data JPA -->
  <dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-data-jpa</artifactId>
  </dependency>

  <!-- PostgreSQL Driver -->
  <dependency>
    <groupId>org.postgresql</groupId>
    <artifactId>postgresql</artifactId>
    <scope>runtime</scope>
```

```
    </dependency>

    <!-- Thymeleaf -->
    <dependency>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-thymeleaf</artifactId>
    </dependency>

    <!-- Spring Boot DevTools (optional for live reload) -->
    <dependency>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-devtools</artifactId>
        <scope>runtime</scope>
    </dependency>

    <!-- Test Dependencies (Optional) -->
    <dependency>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-test</artifactId>
        <scope>test</scope>
    </dependency>
</dependencies>
```

Step 2: Create Entity Class for User

Create User Entity: In com.example.model, create a User.java class:

```
package com.example.model;


import javax.persistence.*;

@Entity
@Table(name = "app_user")  // Renaming table to avoid using reserved keyword
public class User {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;

    @Column(nullable = false, unique = true)
    private String username;

    @Column(nullable = false)
    private String password;

    // Getters and Setters
    public Long getId() {
        return id;
    }

    public void setId(Long id) {
        this.id = id;
    }

    public String getUsername() {
        return username;
    }

    public void setUsername(String username) {
        this.username = username;
    }

    public String getPassword() {
        return password;
```

```
    }

    public void setPassword(String password) {
        this.password = password;
    }
}
```

Step 3: Set up Repository for User:

Create UserRepository Interface: In com.example.repository, create a repository interface:

```
package com.example.repository;

import com.example.model.User;
import org.springframework.data.jpa.repository.JpaRepository;

public interface UserRepository extends JpaRepository<User, Long> {
    User findByUsername(String username);
}
```

Step 4: Service Layer for User Authentication:

Create UserService: In com.example.service, create a UserService.java:

```
package com.example.service;

import com.example.model.User;
import com.example.repository.UserRepository;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;

@Service
public class UserService {
    @Autowired
    private UserRepository userRepository;

    public User save(User user) {
        return userRepository.save(user);
    }

    public User findByUsername(String username) {
        return userRepository.findByUsername(username);
    }
}
```

Step 5: Create Login and Signup Controllers

Create AuthController: In com.example.controller, create a AuthController.java:

```
package com.example.controller;

import com.example.model.User;
import com.example.service.UserService;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.RequestParam;
import org.springframework.web.servlet.mvc.support.RedirectAttributes;
```

```
@Controller
public class AuthController {

    @Autowired
    private UserService userService;

    @GetMapping("/signup")
    public String signupForm() {
        return "signup";
    }

    @PostMapping("/signup")
    public String signup(User user) {
        userService.save(user);
        return "redirect:/login";
    }

    @GetMapping("/login")
    public String loginForm() {
        return "login";
    }

    @PostMapping("/login")
    public String login(@RequestParam String username, @RequestParam String
password, Model model, RedirectAttributes redirectAttributes) {
        User user = userService.findByUsername(username);
        if (user != null && user.getPassword().equals(password)) {
            // Add user info to redirect attributes to pass to the dashboard
            redirectAttributes.addFlashAttribute("username",
user.getUsername());
            return "redirect:/dashboard";
        }
        model.addAttribute("error", "Invalid username or password");
        return "login";
    }

    @GetMapping("/dashboard")
    public String dashboard(Model model) {
        // Optional: Add any additional data you want to pass to the dashboard
view
        // Example: model.addAttribute("message", "Welcome to your dashboard");
        return "dashboard";
    }
}
```

Step 6: Create HTML Views Using Thymeleaf

Create Thymeleaf Templates: Create signup.html, login.html and dashboard.html in src/main/resources/templates:

a. signup.html file:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Signup</title>
    <link
href="https://stackpath.bootstrapcdn.com/bootstrap/4.5.2/css/bootstrap.min.css"
rel="stylesheet">
</head>
<body>
```

```
<div class="container mt-5">
    <h2 class="text-center">Sign Up</h2>
    <form action="/signup" method="post" class="mt-4">
        <div class="form-group">
            <label for="username">Username</label>
            <input type="text" id="username" name="username" class="form-
control" required>
        </div>
        <div class="form-group">
            <label for="password">Password</label>
            <input type="password" id="password" name="password" class="form-
control" required>
        </div>
        <button type="submit" class="btn btn-primary btn-block">Sign Up</button>
    </form>
</div>

<!-- Bootstrap JS and dependencies -->
<script src="https://code.jquery.com/jquery-3.5.1.slim.min.js"></script>
<script
src="https://cdn.jsdelivr.net/npm/@popperjs/core@2.0.7/dist/umd/popper.min.js"><
/script>
<script
src="https://stackpath.bootstrapcdn.com/bootstrap/4.5.2/js/bootstrap.min.js"></s
cript>
</body>
</html>
```

b. login.html:

```
<!DOCTYPE html>
<html xmlns:th="http://www.thymeleaf.org">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Login</title>
    <link
href="https://stackpath.bootstrapcdn.com/bootstrap/4.5.2/css/bootstrap.min.css"
rel="stylesheet">
</head>
<body>
<div class="container mt-5">
    <h2 class="text-center">Login</h2>
    <form action="/login" method="post" class="mt-4">
        <div class="form-group">
            <label for="username">Username</label>
            <input type="text" id="username" name="username" class="form-
control" required>
        </div>
        <div class="form-group">
            <label for="password">Password</label>
            <input type="password" id="password" name="password" class="form-
control" required>
        </div>
        <button type="submit" class="btn btn-primary btn-block">Login</button>

        <!-- Display error if any -->
        <p class="text-danger text-center mt-3" th:text="${error}"></p>
    </form>
</div>

<!-- Bootstrap JS and dependencies -->
```

```
<script src="https://code.jquery.com/jquery-3.5.1.slim.min.js"></script>
<script
src="https://cdn.jsdelivr.net/npm/@popperjs/core@2.0.7/dist/umd/popper.min.js"><
/script>
<script
src="https://stackpath.bootstrapcdn.com/bootstrap/4.5.2/js/bootstrap.min.js"></s
cript>
</body>
</html>
```

c. dashboard.html:

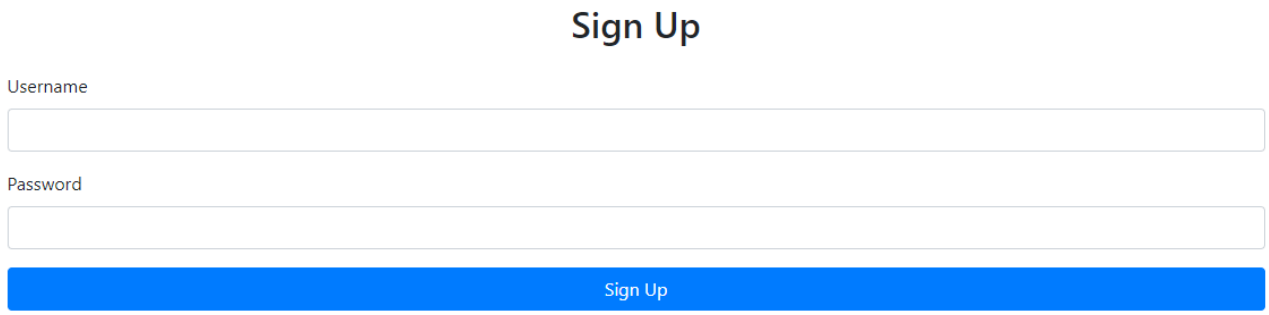
```
<!DOCTYPE html>
<html xmlns:th="http://www.thymeleaf.org">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Dashboard</title>
    <link rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/4.5.2/css/bootstrap.min.css">
</head>
<body>

<!-- Navigation Bar -->
<nav class="navbar navbar-expand-lg navbar-light bg-light">
    <a class="navbar-brand" href="#">Dashboard</a>
    <button class="navbar-toggler" type="button" data-toggle="collapse" data-
target="#navbarNav" aria-controls="navbarNav" aria-expanded="false" aria-
label="Toggle navigation">
        <span class="navbar-toggler-icon"></span>
    </button>
    <div class="collapse navbar-collapse" id="navbarNav">
        <ul class="navbar-nav ml-auto">
            <li class="nav-item active">
                <a class="nav-link" href="#">Home</a>
            </li>
            <li class="nav-item">
                <a class="nav-link" href="/profile">Profile</a>
            </li>
            <li class="nav-item">
                <a class="nav-link" href="/logout">Logout</a>
            </li>
        </ul>
    </div>
</nav>

<!-- Main Content -->
<div class="container mt-5">
    <div class="row">
        <div class="col-md-12 text-center">
            <h1>Welcome, <span th:text="${username}">User</span>!</h1>
            <p>This is your dashboard where you can manage your profile, view
modules, and more.</p>
        </div>
    </div>

    <!-- Cards Section -->
    <div class="row mt-4">
        <div class="col-md-4">
            <div class="card">
                <img src="https://via.placeholder.com/150" class="card-img-top"
alt="Profile">
                <div class="card-body">
                    <h5 class="card-title">Profile</h5>
                    <p class="card-text">Manage your profile settings.</p>
```

```
                    <a href="/profile" class="btn btn-primary">Go to Profile</a>
                </div>
            </div>
        </div>
        <div class="col-md-4">
            <div class="card">
                <img src="https://via.placeholder.com/150" class="card-img-top"
alt="Modules">
                <div class="card-body">
                    <h5 class="card-title">Modules</h5>
                    <p class="card-text">View and access your modules.</p>
                    <a href="/modules" class="btn btn-primary">Go to Modules</a>
                </div>
            </div>
        </div>
        <div class="col-md-4">
            <div class="card">
                <img src="https://via.placeholder.com/150" class="card-img-top"
alt="Settings">
                <div class="card-body">
                    <h5 class="card-title">Settings</h5>
                    <p class="card-text">Manage your account settings.</p>
                    <a href="/settings" class="btn btn-primary">Go to
Settings</a>
                </div>
            </div>
        </div>
    </div>
</div>

<!-- Footer -->
<footer class="footer mt-auto py-3 bg-light">
    <div class="container text-center">
        <span class="text-muted">&copy; 2024 Your Company. All Rights
Reserved.</span>
    </div>
</footer>

<!-- Bootstrap JS and dependencies -->
<script src="https://code.jquery.com/jquery-3.5.1.slim.min.js"></script>
<script
src="https://cdn.jsdelivr.net/npm/@popperjs/core@2.9.2/dist/umd/popper.min.js"><
/script>
<script
src="https://stackpath.bootstrapcdn.com/bootstrap/4.5.2/js/bootstrap.min.js"></s
cript>
</body>
</html>
```

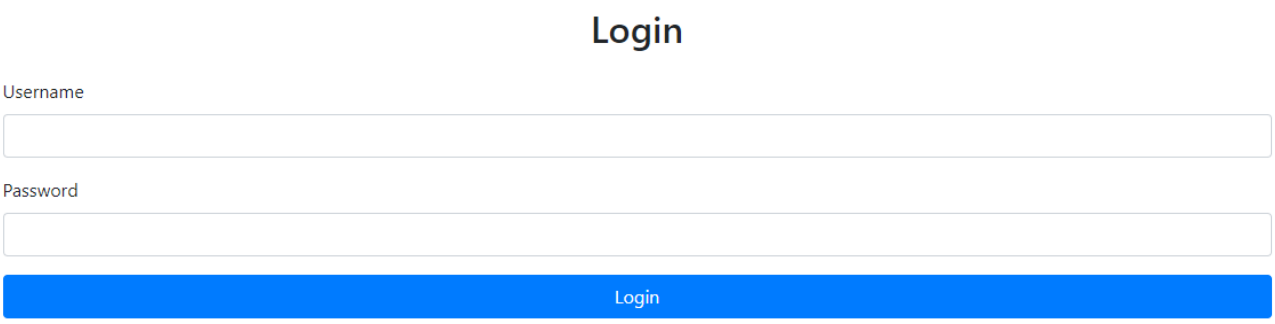
Step 7: Run and Test

Run the Spring Boot application, and here are some screenshots:

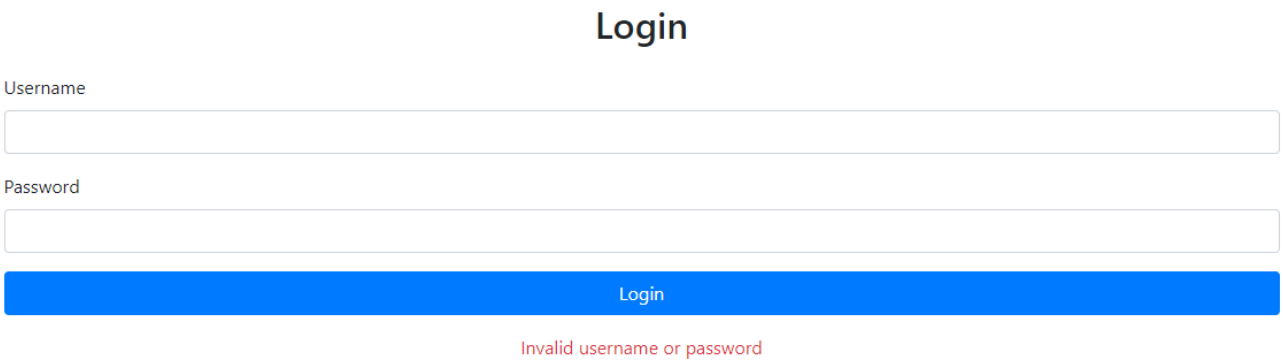
For the [**Sign up**] function: http://localhost:8080/signup



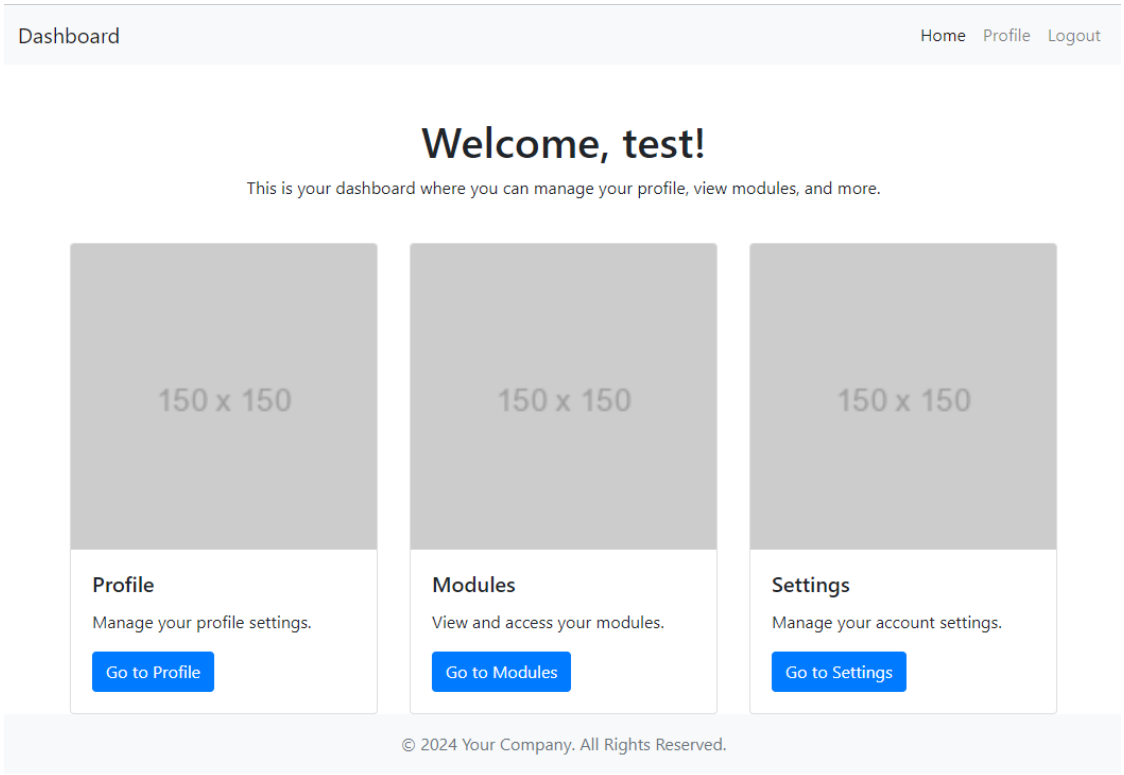
For the [**Sign in**] function: http://localhost:8080/login



If the login fails, this is the screen you will see:



After logging in successfully, this is the dashboard:



----oOo-----

THE END